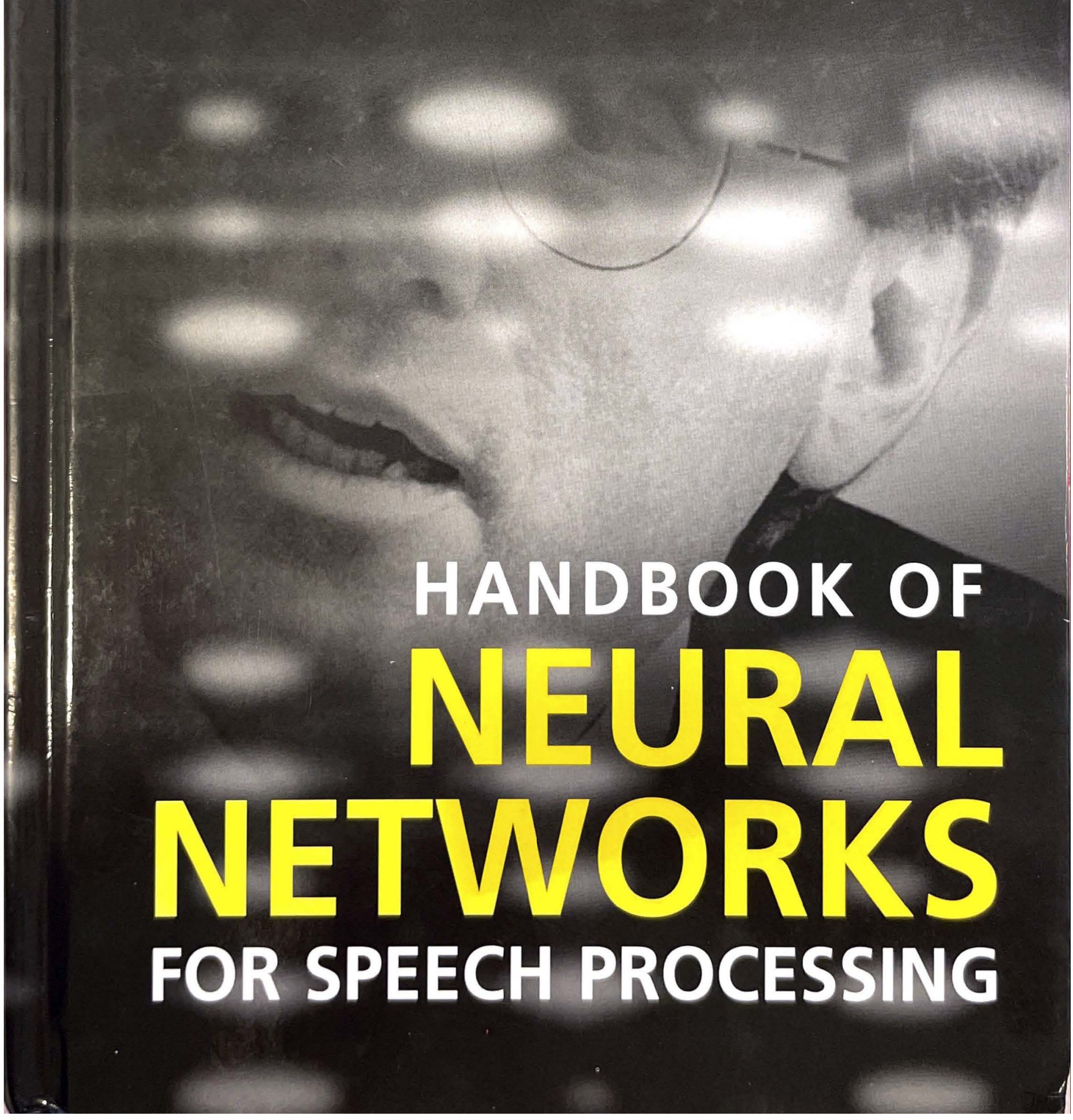
SHIGERU KATAGIRI
EDITOR



HANDBOOK OF NEURAL NETWORKS FOR SPEECH PROCESSING

Amazon / Zentian Limited
Exhibit 1021
Page 1

Handbook of Neural Networks for Speech Processing

Shigeru Katagiri

Editor



Artech House
Boston • London
www.artechhouse.com

Amazon / Zentian Limited
Exhibit 1021
Page 2

Library of Congress Cataloging-in-Publication Data
Handbook of neural networks for speech processing / Shigeru Katagiri, editor.
p. cm. — (Artech House signal processing library)
Includes bibliographical references and index.
ISBN 0-89006-954-9 (alk. paper)
1. Automatic speech recognition—Handbooks, manuals, etc. 2. Neural networks (Computer science)—Handbooks, manuals, etc. 3. Speech processing systems—Handbooks, manuals, etc. I. Katagiri, Shigeru. II. Series.

TK7895.S65 H36 2000 00-055883
006.4'54—dc21 CIP

British Library Cataloguing in Publication Data
Handbook of neural networks for speech processing
(Artech House signal processing library)
1. Neural networks (Computer science) 2. Speech processing systems 3. Speech perception
I. Katagiri, Shigeru
006.5'4

ISBN 0-89006-954-9

Cover design by Igor Valdman

© 2000 ARTECH HOUSE, INC.
685 Canton Street
Norwood, MA 02062

All rights reserved. Printed and bound in the United States of America. No part of this book may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without permission in writing from the publisher.

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Artech House cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

International Standard Book Number: 0-89006-954-9
Library of Congress Catalog Card Number: 00-055883

10 9 8 7 6 5 4 3 2 1

Amazon / Zentian Limited
Exhibit 1021
Page 3

Contents

	Preface	**xxi**
	Part I Fundamentals	
1	**Introduction**	**3**
1.1	Speech Processing	3
1.2	Neural Networks	6
1.2.1	Fundamentals	6
1.2.2	Taxonomy of Neural Networks	8
1.2.2.1	Overview	8
1.2.2.2	Structure	9
1.2.2.3	Measurement	11
1.2.2.4	Objective Function	13
1.2.2.5	Optimization	14
1.3	Neural Networks for Speech Processing	15
1.4	Handbook Overview	15
1.4.1	Part I: Fundamentals	15
1.4.2	Part II: Current Issues in Speech Recognition	16
1.4.3	Part III: Current Issues in Speech Signal Processing	17
	References	17

Amazon / Zentian Limited
Exhibit 1021
Page 4

2	**The Speech Signal and Its Production Model**	**19**
2.1	Introduction	19
2.2	Information Conveyed by Speech	21
2.2.1	Linguistic Information	21
2.2.1.1	Segmental Features	21
2.2.1.2	Suprasegmental Features	22
2.2.2	Paralinguistic Information	23
2.2.3	Nonlinguistic Information	27
2.2.3.1	Idiosyncratic Factors	27
2.2.3.2	Emotional Factors	29
2.2.4	Hierarchical Speech Production Processes	30
2.2.5	Summary	31
2.3	Physical and Physiological Processes in Speech Production	32
2.3.1	Respiration System	32
2.3.1.1	Normal Breathing Without Speech Production	32
2.3.1.2	Expiration in Speech Production	34
2.3.2	Phonatory System	34
2.3.2.1	Framework of the Larynx	34
2.3.2.2	Abduction versus Adduction	34
2.3.2.3	F0 Control During Speech	37
2.3.3	Articulatory System	39
2.3.3.1	Morphology of Articulators	39
2.3.3.2	Vowel Production	40
2.3.3.3	Consonant Production	41
2.3.4	Summary	46
2.4	Models and Theories of Speech Production	49
2.4.1	Laryngeal System	49
2.4.1.1	Vocal Fold Vibration	49
2.4.1.2	F0 Control in Running Speech Production	50
2.4.1.3	Vertical Movements of the Larynx	51
2.4.2	Dynamic Characteristic of Articulators	52

Amazon / Zentian Limited
Exhibit 1021
Page 5

2.4.2.1	Models of Individual Articulators	52
2.4.2.2	Articulatory Models of Speech Production	54
2.4.3	Summary	56
2.5	Conclusion	57
	Acknowledgment	58
	References	58
3	**Speech Recognition**	**63**
3.1	Introduction	63
3.2	Overview	64
3.2.1	Hearing and Machine Recognition	64
3.2.2	Recognition-Oriented Speech Feature Representation	66
3.2.2.1	Sound Spectrogram: Time-Frequency-Energy Representation	66
3.2.2.2	Acoustic Feature Vector	69
3.2.2.3	Static Versus Dynamic Nature	72
3.2.3	Variety of Recognition Tasks	73
3.2.4	Recognition Mechanism	73
3.2.4.1	Example Task Setting	73
3.2.4.2	Distance-Based Recognition	76
3.2.4.3	Distance Computation Based on Dynamic Time Warping	77
3.2.4.4	Remarks	80
3.3	Bayes Decision Theory	80
3.3.1	Overview	80
3.3.2	Maximum Likelihood Estimation Approach	83
3.3.3	Bayesian Approach	85
3.3.4	Discriminant Function Approach	86
3.3.4.1	Example Task and Decision Rule	86
3.3.4.2	Loss	88
3.3.4.3	Optimization	88
3.3.4.4	Design of Linear Discriminant Function Classifier	90

Amazon / Zentian Limited
Exhibit 1021
Page 6

3.3.4.5	Remarks	91
3.4	Acoustic Feature Extraction	91
3.4.1	Filter-Bank	91
3.4.1.1	Artificial Cochlea Filter	93
3.4.1.2	Fourier-Transform-Based Filter	94
3.4.2	Autoregressive Modeling	94
3.4.3	Cepstrum Modeling	98
3.4.4	Dynamic Feature Modeling	101
3.5	Probabilistic Acoustic Modeling Based on Hidden Markov Model	103
3.5.1	Principles of Hidden Markov Model	103
3.5.2	Selection of Output Probability Function	106
3.5.2.1	Discrete Model	106
3.5.2.2	Continuous Model	106
3.5.3	MLE-Based Design Method	109
3.5.3.1	Forward-Backward Method	109
3.5.4	Trellis Algorithm and Viterbi Algorithm	111
3.5.5	Discriminative Design Methods	112
3.6	Language Modeling	112
3.6.1	Role of Language Modeling	112
3.6.2	*N*-Gram Language Modeling	114
3.7	Concluding Remarks	115
3.7.1	Overview	115
3.7.2	Selection of Model Units	115
3.7.3	Open-Vocabulary Recognition	116
3.7.4	Bibliographical Remarks	117
	References	117
4	**Speech Coding**	**121**
4.1	Introduction	121
4.2	Attributes of Speech Coders	122
4.3	Basic Principles of Speech Coders	123
4.4	Quantization	126

Amazon / Zentian Limited
Exhibit 1021
Page 7

4.4.1	Scalar Quantization	126
4.4.2	Vector Quantization	126
4.5	Linear Prediction	128
4.5.1	Linear Prediction Principles	128
4.5.2	Speech Coding Based on Linear Prediction	129
4.5.3	The Analysis-by-Synthesis Principle	131
4.5.4	Perceptual Filtering	135
4.5.5	Quantization of the Linear Prediction Coefficients	136
4.6	Sinusoidal Coding	141
4.7	Waveform Interpolation Methods	142
4.8	Subband Coding	143
4.9	Variable-Rate Coding	144
4.9.1	Basics	144
4.9.2	Phonetic Segmentation	145
4.9.3	Variable Rate Coders for ATM Networks	145
4.9.4	Voice over IP	146
4.10	Wideband Coders	146
4.11	Measuring Speech Coder Performance	147
4.12	Speech Coding over Noisy Channels	150
4.13	Speech Coding Standards	151
4.14	Conclusions	152
	References	153

Part II Current Issues in Speech Recognition

5	**Discriminative Prototype-Based Methods for Speech Recognition**	**159**
5.1	Introduction	159
5.2	Bayes Decision Theory	161
5.2.1	The Bayes Decision Rule	161
5.2.2	Discriminant Functions	161
5.2.3	Discriminant Functions for Prototype-Based Methods	163

Amazon / Zentian Limited
Exhibit 1021
Page 8

5.3	Example-Based Methods	165
5.3.1	Density Estimation	166
5.3.2	Estimation of Posterior Probabilities	167
5.3.3	The *k*-Nearest-Neighbor Method	168
5.3.3.1	The Nearest-Neighbor Rule	168
5.3.3.2	Error Bounds for *k*-Nearest-Neighbor Classification	168
5.3.4	Parzen Windows	168
5.3.4.1	An Example	170
5.3.5	Advantages and Limitations of Example-Based Methods	171
5.3.6	Smoothing	172
5.3.7	Applications to Speech Recognition	173
5.4	Prototype-Based Methods for Speech Recognition	173
5.5	Prototype-Based Classifier Design Using Minimum Classification Error	175
5.5.1	Definition of Discriminant Function	175
5.5.2	Definition of Misclassification Measure	176
5.5.3	Definition of Local Loss Function	176
5.5.4	Overall Loss Function and Optimization	177
5.5.5	Modified Newton's Method: The Quickprop Algorithm	178
5.5.6	Relation of MCE Loss to the Bayes Error	181
5.5.7	Choice of Smoothing Parameters for MCE-Based Optimization	182
5.6	Learning Vector Quantization	182
5.6.1	Shift-Tolerant LVQ for Speech Recognition	185
5.6.1.1	HMM Interpretation of STLVQ	187
5.6.1.2	Limitations and Strengths of STLVQ Architecture and Training	187
5.6.2	Expanding the Scope of LVQ for Speech Recognition: Incorporation into Hidden Markov Modeling	188
5.6.2.1	LVQ-HMM	189

Amazon / Zentian Limited
Exhibit 1021
Page 9

5.3	Example-Based Methods	165
5.3.1	Density Estimation	166
5.3.2	Estimation of Posterior Probabilities	167
5.3.3	The *k*-Nearest-Neighbor Method	168
5.3.3.1	The Nearest-Neighbor Rule	168
5.3.3.2	Error Bounds for *k*-Nearest-Neighbor Classification	168
5.3.4	Parzen Windows	168
5.3.4.1	An Example	170
5.3.5	Advantages and Limitations of Example-Based Methods	171
5.3.6	Smoothing	172
5.3.7	Applications to Speech Recognition	173
5.4	Prototype-Based Methods for Speech Recognition	173
5.5	Prototype-Based Classifier Design Using Minimum Classification Error	175
5.5.1	Definition of Discriminant Function	175
5.5.2	Definition of Misclassification Measure	176
5.5.3	Definition of Local Loss Function	176
5.5.4	Overall Loss Function and Optimization	177
5.5.5	Modified Newton's Method: The Quickprop Algorithm	178
5.5.6	Relation of MCE Loss to the Bayes Error	181
5.5.7	Choice of Smoothing Parameters for MCE-Based Optimization	182
5.6	Learning Vector Quantization	182
5.6.1	Shift-Tolerant LVQ for Speech Recognition	185
5.6.1.1	HMM Interpretation of STLVQ	187
5.6.1.2	Limitations and Strengths of STLVQ Architecture and Training	187
5.6.2	Expanding the Scope of LVQ for Speech Recognition: Incorporation into Hidden Markov Modeling	188
5.6.2.1	LVQ-HMM	189

Amazon / Zentian Limited
Exhibit 1021
Page 10

5.6.2.2	HMM-LVQ	190
5.6.3	Minimum Classification Error Interpretation of LVQ	192
5.6.4	Smoothness of MCE Loss	193
5.6.5	LVQ Summary	195
5.7	Prototype-Based Methods Using Dynamic Programming	195
5.7.1	MCE-Trained Prototypes for DTW-Based Speech Recognition	196
5.7.1.1	Practical Implementation of MCE/GPD	198
5.7.1.2	MCE-DTW Results	199
5.7.2	Prototype-Based Minimum Error Classifier	199
5.7.2.1	PBMEC State Distance and Discriminant Function	200
5.7.2.2	MCE/GPD in the Context of Speech Recognition Using Phoneme Models	202
5.7.2.3	PBMEC Results	202
5.7.3	Summary of Prototype-Based Methods Using DP	203
5.8	Hidden Markov Model Design Based on MCE	204
5.8.1	HMM State Likelihood and Discriminant Function	206
5.8.2	MCE Misclassification Measure and Loss	207
5.8.3	Calculation of MCE Gradient for HMMs	207
5.8.3.1	Derivative of Loss with Respect to Misclassification Measure	207
5.8.3.2	Derivative of Misclassification Measure with Respect to Discriminant Functions	208
5.8.3.3	Derivative of Discriminant Function with Respect to Observation Probability Density Function	208
5.8.3.4	Derivative of Observation Probability with Respect to Mixing Weights	209
5.8.3.5	Derivative of Observation Probability with Respect to Mean Vectors	209

Amazon / Zentian Limited
Exhibit 1021
Page 11

5.8.3.6	Derivative of Observation Probability with Respect to Covariances	210
5.8.3.7	Application of the Chain Rule	210
5.8.4	MCE-HMM Results	211
5.9	Conclusion	212
	References	213
6	**Recurrent Neural Networks for Speech Recognition**	**217**
6.1	Introduction	217
6.1.1	Background and Motivation	217
6.1.2	Chapter Overview	219
6.2	Speech Recognition Theory	220
6.3	Basics of Neural Networks	223
6.3.1	Parameter Estimation by Maximum Likelihood	224
6.3.2	Problem Classification	225
6.3.2.1	Regression	225
6.3.2.2	Classification	226
6.3.3	Neural Network Training	226
6.3.3.1	Gradient Descent Training	227
6.3.3.2	RPROP Training	228
6.3.3.3	ARPROP Training	229
6.3.4	Neural Network Architectures	230
6.3.4.1	Multilayer Perceptrons	232
6.3.4.2	Time-Delay Neural Networks	232
6.4	Recurrent Neural Networks	232
6.4.1	Unidirectional Recurrent Neural Network	233
6.4.1.1	RNN Architecture	233
6.4.1.2	RNN Training	235
6.4.2	Bidirectional Recurrent Neural Network	235
6.4.2.1	BRNN Architecture	235
6.4.2.2	BRNN Training	236
6.5	Modeling Phonetic Context	237

Amazon / Zentian Limited
Exhibit 1021
Page 12

6.6	System Training and Usage	239
6.6.1	Training	239
6.6.2	Usage	240
6.7	Conclusions	240
6.7.1	Discussion	240
6.7.1.1	Training Criterion	240
6.7.1.2	Discriminative Training	241
6.7.1.3	Distribution of Model Complexity	241
6.7.2	Summary	241
	References	242
7	**Time-Delay Neural Networks and NN/HMM Hybrids: A Family of Connectionist Continuous-Speech Recognition Systems**	**245**
7.1	Introduction	245
7.2	MS-TDNNs and NN/HMM Hybrid Approaches	246
7.2.1	The Time-Delay Neural Network (TDNN)	247
7.2.2	Multistate TDNN	249
7.2.3	MS-TDNN Variants	249
7.2.4	Hybrid NN/HMM Variants	249
7.3	Alphabet Recognition with the MS-TDNN	251
7.3.1	Training Procedures	251
7.3.2	Duration Modeling	254
7.3.3	Experiments	255
7.3.3.1	Speaker-Dependent Data	255
7.3.3.2	Speaker-Independent Data	255
7.3.3.3	Telephone Data	256
7.3.4	Searching in Large Name Lists	257
7.4	Multimodal Input: Lipreading	260
7.4.1	Motivation	260
7.4.2	The Recognizer	260
7.4.3	Results	264
7.5	Modular Neural Networks	265

Amazon / Zentian Limited
Exhibit 1021
Page 13

7.5.1	Architecture	266
7.5.2	Application to NN/HMM Models	267
7.5.3	Experiments with a Hybrid HME/HMM System	268
7.6	Context Modeling	269
7.6.1	Clustering Context Classes	269
7.6.2	Factoring Context-Dependent Posteriors	270
7.6.3	Hierarchies of Neural Networks	272
7.6.3.1	Manually Structured Hierarchies	273
7.6.3.2	Clustering Hierarchies of Neural Networks	274
7.6.4	Experiments and Results	275
	References	277
8	**Probability-Oriented Neural Networks and Hybrid Connectionist/Stochastic Networks**	**281**
8.1	Introduction	281
8.2	Fundamentals of Probability-Oriented Neural Networks	282
8.2.1	The Bayes Decision Framework	282
8.2.2	Training Procedures	283
8.2.3	Types of PONNs	284
8.2.3.1	Radial Basis Function Networks	284
8.2.3.2	Probabilistic Neural Networks	286
8.2.4	Summary	288
8.3	Learning Methods for PNNs	289
8.3.1	Position of the Problem	289
8.3.2	MLE and EM Algorithms for PNNs	290
8.3.3	MMIE for PNNs	290
8.4	Applications to Automatic Speech Recognition	291
8.4.1	Speech Recognition	291
8.4.2	Speaker Recognition	291
8.5	Hybrid Connectionist/Stochastic Models	293
8.5.1	Position of the Problem	293

Amazon / Zentian Limited
Exhibit 1021
Page 14

8.5.2	Proposed Solutions	293
8.5.2.1	ANNs as Front-Ends for HMMs	294
8.5.2.2	ANNs as Postprocessors of HMMs	296
8.5.2.3	Unified Models	297
8.5.3	Summary	298
8.6	Conclusion	299
	References	299

9	**Minimum Classification Error Networks**	**307**
9.1	Introduction	307
9.1.1	Speech Pattern Recognition Using Modular Systems	307
9.1.2	Classifier Design	309
9.1.3	What Is an Artificial Neural Network?	310
9.1.4	Minimum Recognition Error Network	311
9.1.5	Chapter Organization	312
9.2	Discriminative Pattern Classification	312
9.2.1	Bayes Decision Theory	312
9.2.2	Minimum Error Rate Classification	314
9.2.3	Discriminative Training	314
9.3	Generalized Probabilistic Descent Method	317
9.3.1	Overview	317
9.3.2	Formalization Fundamentals	319
9.3.2.1	Distance Classifier for Classifying Dynamic Patterns: Preparation	319
9.3.2.2	Emulation of Decision Process	321
9.3.2.3	Selection of Loss Functions	324
9.3.2.4	Design Optimality in Practical Situations	326
9.3.3	GPD-Based Classifier Design	327
9.3.3.1	*E*-Set Task	327
9.3.3.2	*P*-Set Task	328
9.4	Derivatives of GPD	328
9.4.1	Overview	328

Amazon / Zentian Limited
Exhibit 1021
Page 15

9.4.2	Segmental GPD for Continuous Speech Recognition	329
9.4.3	Minimum Error Training for Open-Vocabulary Speech Recognition	331
9.4.3.1	Open-Vocabulary Speech Recognition	331
9.4.3.2	Minimum Spotting Error Learning	332
9.4.3.3	Discriminative Utterance Verification	334
9.4.4	Discriminative Feature Extraction	336
9.4.4.1	Fundamentals	336
9.4.4.2	An Example Implementation for Cepstrum-Based Speech Recognition	338
9.4.4.3	Discriminative Metric Design	339
9.4.4.4	Minimum Error Learning Subspace Method	339
9.4.5	Speaker Recognition Using GPD	340
9.5	Summary	342
	Acknowledgments	343
	References	343
	Appendix 1 Probabilistic Descent Theorem for Probability-Based Discriminant Functions	347
	Appendix 2 Relationships Between MCE/GPD and Others	349

Part III Current Issues in Speech Signal Processing

10	**Networks for Speaker Recognition**	**357**
10.1	Introduction	357
10.2	Speaker Recognition Overview	359
10.3	Discriminative Information	363
10.3.1	Supervised Training	363
10.3.2	Cohort Normalization	365
10.4	Speaker Recognition Networks	366
10.4.1	Multilayer Perceptron	366
10.4.2	Radial Basis Functions	369

Amazon / Zentian Limited
Exhibit 1021
Page 16

10.4.3	Time-Delay Neural Networks	370
10.4.4	Recurrent Neural Networks	371
10.4.5	Learning Vector Quantization	372
10.4.6	Decision Trees	372
10.4.7	Neural Tree Network	376
10.4.8	Performance Summary	377
10.5	Model Combination	377
10.5.1	Model Combination Approaches	381
10.5.1.1	Linear Opinion Pool	381
10.5.1.2	Log Opinion Pool	381
10.5.1.3	Voting Methods	381
10.5.2	Error Correlation Analysis	382
10.5.3	Two-Model Combination	383
10.5.4	Three-Model Combination	384
10.6	Summary	386
	Acknowledgments	388
	References	388
11	**Neural Networks for Voice Conversion**	**393**
11.1	Introduction: Speech and Speaker Characteristics	393
11.2	Studies in Voice Conversion	401
11.3	Neural Networks for Transformation of Vocal Tract Shapes	405
11.3.1	Linear Approximation of Formant Transformation	406
11.3.2	Neural Network Models	410
11.3.3	Generalization	412
11.4	Implementation of Voice Conversion	420
11.4.1	Voice Transformation System	420
11.4.2	Normalization of Intonational Features	424
11.4.3	Evaluation of Voice Transformation	426
11.5	Summary	428
	References	430

Amazon / Zentian Limited
Exhibit 1021
Page 17

10.4.3	Time-Delay Neural Networks	370
10.4.4	Recurrent Neural Networks	371
10.4.5	Learning Vector Quantization	372
10.4.6	Decision Trees	372
10.4.7	Neural Tree Network	376
10.4.8	Performance Summary	377
10.5	Model Combination	377
10.5.1	Model Combination Approaches	381
10.5.1.1	Linear Opinion Pool	381
10.5.1.2	Log Opinion Pool	381
10.5.1.3	Voting Methods	381
10.5.2	Error Correlation Analysis	382
10.5.3	Two-Model Combination	383
10.5.4	Three-Model Combination	384
10.6	Summary	386
	Acknowledgments	388
	References	388
11	**Neural Networks for Voice Conversion**	**393**
11.1	Introduction: Speech and Speaker Characteristics	393
11.2	Studies in Voice Conversion	401
11.3	Neural Networks for Transformation of Vocal Tract Shapes	405
11.3.1	Linear Approximation of Formant Transformation	406
11.3.2	Neural Network Models	410
11.3.3	Generalization	412
11.4	Implementation of Voice Conversion	420
11.4.1	Voice Transformation System	420
11.4.2	Normalization of Intonational Features	424
11.4.3	Evaluation of Voice Transformation	426
11.5	Summary	428
	References	430

Amazon / Zentian Limited
Exhibit 1021
Page 18

12	**Neural Networks for Speech Coding**	**433**
12.1	Introduction	433
12.2	Source Coding and Neural Networks	433
12.2.1	Source Coding	433
12.2.2	Neural Networks	435
12.2.3	Source Coding with Neural Networks	436
12.2.3.1	Vector Quantization with Kohonen Self-Organizing Feature Maps	437
12.2.3.2	Multilayer Neural Network as Front-End of a Coder	437
12.2.3.3	Codebook-Excited Neural Networks	437
12.3	Quantization Performance of Neural Networks	438
12.3.1	Kohonen Self-Organizing Feature Maps	438
12.3.1.1	Architecture and Training Process	441
12.3.1.2	Conditional Histogram Neural Network FSVQ	443
12.3.1.3	Nearest-Neighbor Neural Network FSVQ	444
12.3.1.4	Simulations	445
12.3.2	Coders with Neural Network Front-Ends	447
12.3.3	Codebook-Excited Neural Networks	451
12.3.4	Concluding Remarks	452
12.4	Speech Coding with Neural Networks	453
12.4.1	Coding Speech Spectrum with Neural Networks	453
12.4.2	Nonlinear Prediction Speech Coding	454
12.4.2.1	A Neural Model of Nonlinear Prediction	455
12.4.2.2	Nonlinear Predictive Vector Quantization	456
12.4.2.3	Nonlinear Predictive Quantization Performance	458
12.4.3	Code-Excited Nonlinear Predictive Speech Coding	461
12.4.3.1	Nonlinear Predictive Filter Tolerance for an Excitation Disturbance	461
12.4.3.2	Gain-Adaptive Nonlinear Predictive Coding	463

Amazon / Zentian Limited
Exhibit 1021
Page 19

12.4.3.3	Coding Performance	464
12.4.4	Concluding Remarks	466
12.5	Conclusions	466
	References	467
13	**Networks for Speech Enhancement**	**471**
13.1	Introduction	471
13.1.1	Background	471
13.1.2	Model Structure	472
13.1.3	Chapter Overview	473
13.2	Neural Time-Domain Filtering Methods	474
13.2.1	Direct Time-Domain Mapping	474
13.2.2	Extended Kalman Filtering with Predictive Models	478
13.3	Neural Transform-Domain Methods	481
13.3.1	Spectral Subtraction	482
13.3.2	Neural Transform-Domain Mappings	484
13.4	State-Dependent Model Switching Methods	488
13.4.1	Classification Switched Models	488
13.4.2	Hybrid HMM and EKF	489
13.5	Online Iterative Methods	490
13.5.1	Online Predictive Enhancement	491
13.5.2	Maximum-Likelihood Estimation and Dual Kalman Filtering	492
13.5.3	Noise-Regularized Adaptive Filtering	496
13.6	Summary and Conclusions	499
	Acknowledgments	501
	References	501
	Index	**505**

Amazon / Zentian Limited
Exhibit 1021
Page 20

3

Speech Recognition

Shigeru Katagiri

3.1 Introduction

One aim of speech recognition is to encode acoustic speech signals into linguistic indexes such as words and sentences and also to interpret speech content by machine. Naturally, one of the most promising approaches to such speech recognition is to investigate the human process of speech perception and spoken language understanding and apply the resulting scientific findings to the development of a mathematical model of recognition or recognition machine (recognizer). However, this human-science-based approach has not yet achieved comprehensive analysis of the human mechanism. Though greatly supported by the scientific discipline of "learning the human mechanism," research into speech recognition has been strongly technologically-colored, i.e., it has long been conducted mainly in the research field of signal processing technology. The relationship between human recognition and machine recognition appears similar to that of "bird versus airplane." Actually, although the human mechanism has not yet been fully grasped, speech recognition technology is already providing many useful speech recognition systems and services in daily life. In this chapter, we shall discuss the basics of such speech recognition technologies.

Traditionally, speech recognizers have been developed using system structures other than artificial neural networks, such as a distance classifier using prototypes and a probabilistc classifier using

Amazon / Zentian Limited
Exhibit 1021
Page 21

hidden Markov models (HMMs). Accordingly, this chapter will incorporate these traditional system structures in its discussions. However, these classical structures actually have close relationships with neural networks, and the descriptions in this chapter will be useful for understanding the topics involving neural networks presented in later chapters.

3.2 Overview

3.2.1 Hearing and Machine Recognition

Usually, a speech signal to be recognized exists together with various types of acoustic signals, which inevitably interfere with the accurate recognition of the target speech. Let us imagine a lively party (see Figure 3.1). People are chatting and laughing. A band is playing music. Someone has pulled the cork out of a champagne bottle. In this loud environment, you can still listen to your conversational partner without difficulty. Humans live in a complex acoustic environment (auditory scene) in which various sounds coexist, but they can still easily distinguish a target sound from interference sounds. Recently, there has been a great deal of research interest in a challenging technology,

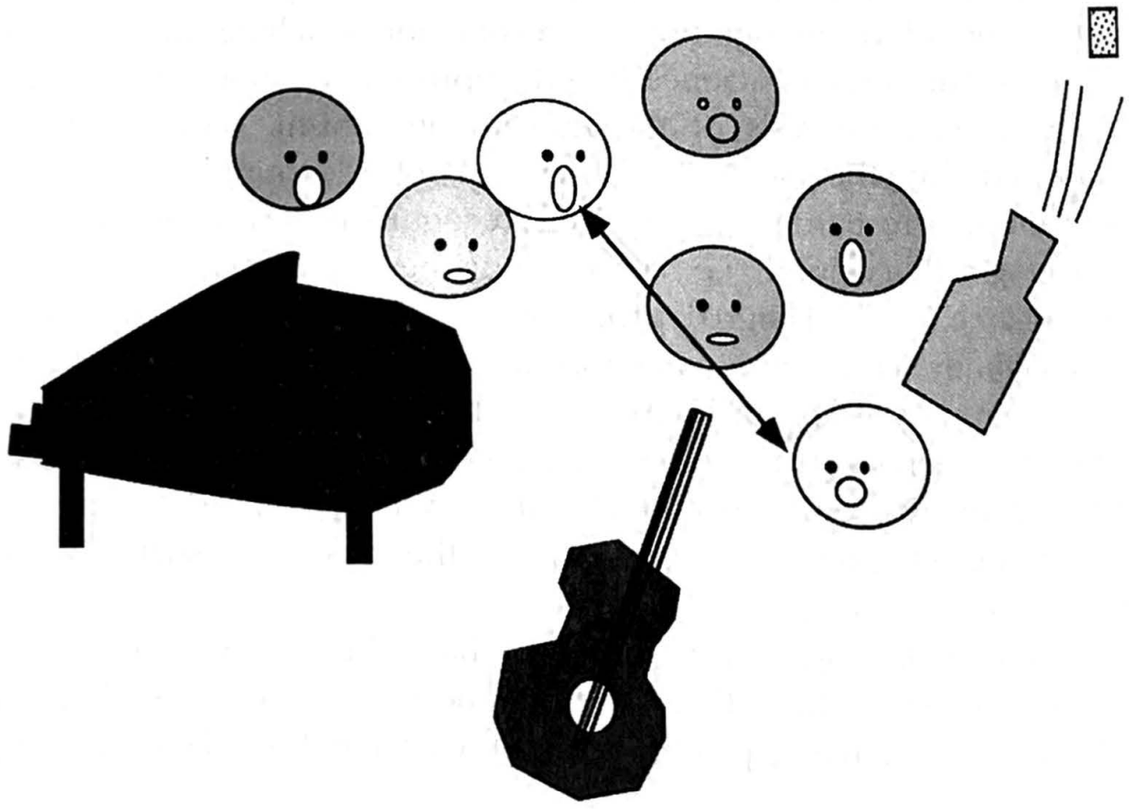


Figure 3.1 A party scene. Humans easily distinguish target sounds from interference sounds.

Amazon / Zentian Limited
Exhibit 1021
Page 22

sometimes called auditory scene analysis, for recognizing speech sounds in complex realistic acoustic environments [1, 2]. However, there has been neither a thorough understanding of the human auditory mechanism nor full development of auditory scene analysis technology; in reality, most existing speech recognition systems (speech recognizers) assume that a speech signal with a high signal-to-noise ratio is extracted from its auditory scene before being given to a recognizer. In this chapter, we will concentrate on this simplified but practical task setting for recognition; we will not discuss the technologies for extracting or segregating a target speech signal from the background auditory scene. Figure 3.2 illustrates the assumed relationship between a realistic acoustic environment and the speech input for a recognition task.

The definition of "recognition" is not necessarily unique to speech recognition research. There are several ways of defining this decision operation. In this chapter, we define it as a process that maps an input speech pattern to one of several prescribed classes (such as phonemes, words, or word sequences) through three modules (sub-processes): (1) feature extraction, (2) segmentation, and (3) classification. Figure 3.3 illustrates this modular recognition process. In the figure, an analogy is also illustrated between the human auditory

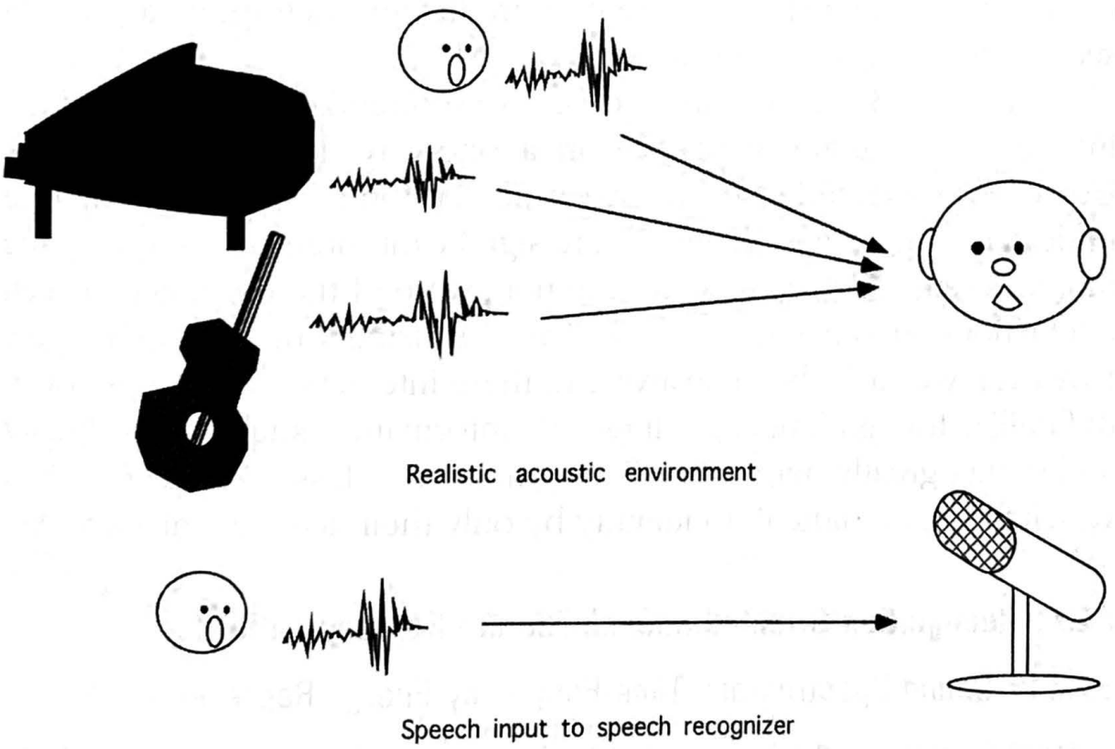


Figure 3.2 Relationship between a realistic acoustic environment and a speech input assumed for recognition.

Amazon / Zentian Limited
Exhibit 1021
Page 23

process and the recognition process. Encouraged by this analogy, the modular-process definition has been widely used in speech recognition, though obviously the modularity illustrated is not the only possible definition.

Figure 3.4 illustrates the basic procedures of speech recognition. Based on the definition cited in the above paragraph, let us assume that a speech wave signal, which is digitized with audio and A/D conversion equipment and extracted from its background acoustic signal, is provided to the recognizer. Recognition begins with feature extraction. This first-step module converts the input speech signal to a feature pattern that is suitable for recognition. The module corresponds to the human auditory (peripheral) system that transforms an acoustic input speech to a neural feature pattern by some feature extraction, which has not yet been fully clarified. Segmentation is an operation of dividing the feature pattern into segments (partial patterns), each corresponding to a linguistic unit such as phoneme or word. Then the final classification module maps a segment to one of the prescribed classes such as words and sentences. In this module, an acoustic model and a language model are assigned to each class in order to measure the degree to which the feature pattern belongs to one class. The acoustic model measures the acoustical class identity (or possibility) by which the feature pattern belongs to its corresponding class. The language model is a linguistic constraint or rule that enables one to predict and determine a possible sequence of linguistic units such as words.

In Figure 3.4, all of the modules are sequentially located, and the information is assumed to flow in a one-way, left-to-right manner. However, our experiences in speech perception and spoken language understanding suggest that there are significant interactions among the modules (e.g., [3]). Actually, we are not aware of the existence of such interactions in our daily communications using our mother tongue. However, we easily become aware of these interactions in the use of an unfamiliar foreign language: linguistic information such as vocabulary and syntax greatly helps identify (segment and classify) foreign words, which are often difficult to identify by only their acoustic information.

3.2.2 Recognition-Oriented Speech Feature Representation

3.2.2.1 Sound Spectrogram: Time-Frequency-Energy Representation

A speech wave is a one-dimensional signal having temporal structure. The signal can be considered a combination of different frequency sine-waves, and its acoustical characteristics are determined by the

Amazon / Zentian Limited
Exhibit 1021
Page 24

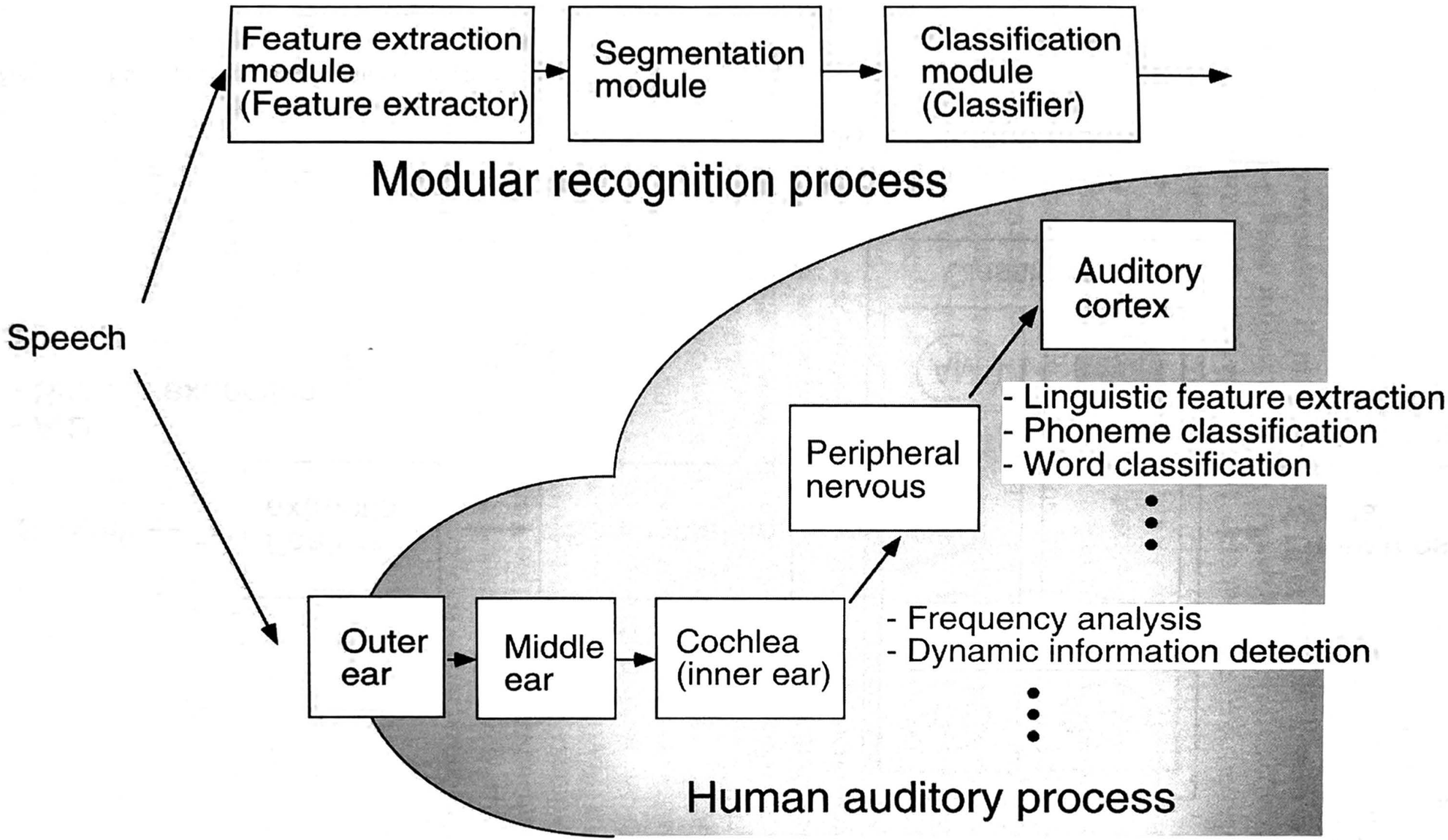


Figure 3.3 Schematic diagram of the human auditory process and a modular recognition process.

Amazon / Zentian Limited
Exhibit 1021
Page 25

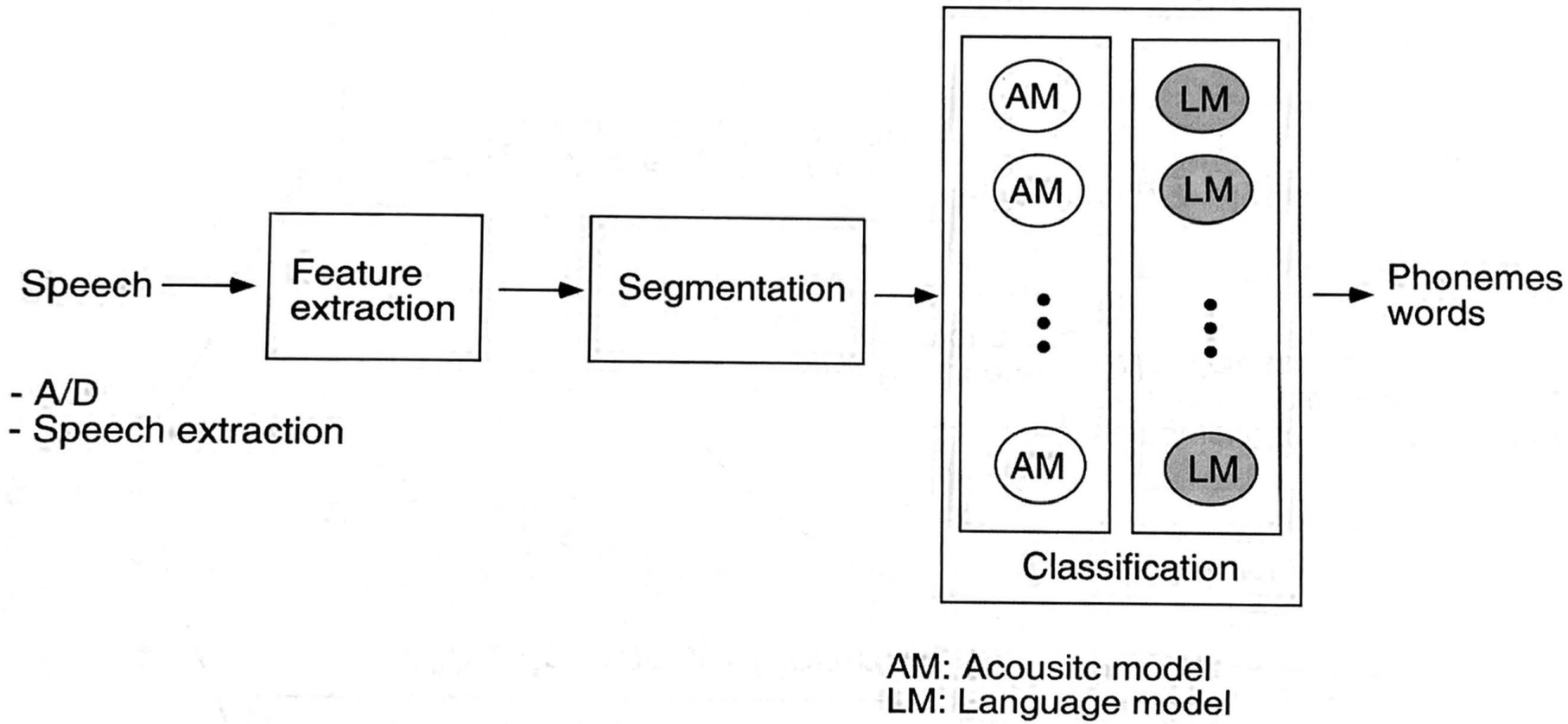


Figure 3.4 Basic procedures of speech recognition.

Amazon / Zentian Limited
Exhibit 1021
Page 26

frequency, energy (amplitude), and phase of each component sine-wave. However, for speech recognition, a speech signal is usually converted to a three-dimensional, time-frequency-energy feature pattern that is similar to a sound spectrogram. The main reason for this selection of conversion is that (1) the human auditory system is not sensitive to the phase information of a speech signal, and (2) the three factors, i.e., time, frequency, and energy, determine the basic perceptual nature of phonemes.

Figure 3.5 is a sample sound spectrogram and its corresponding original speech wave for the word "Yokozuna."[1] In the sound spectrogram, the shade indicates the intensity of energy at every time-frequency point. The darker the shade, the higher the intensity is. The sample includes two segments of the identical vowel class ⟨⟨ o ⟩⟩.[2] The time-fequency-energy patterns of these two segments look quite similar. As in this case, a speech segment is associated with the time-frequency-energy pattern specific to its corresponding linguistic class such as phoneme or syllable. Accordingly, a spoken word and sentence pattern is characterized by the concatenation of the time-frequency-energy patterns, each specific to the basic linguistic units such as phonemes.

Based on the change of speaking speed (rate), the speech signal of "yokozuna" varies in length. A slowly spoken ⟨⟨ yokozuna ⟩⟩ is represented by a long sound spectrogram along the time axis. Temporal warping (extension and shrinking) is observed locally in short segments such as phonemes (e.g., ⟨⟨ k ⟩⟩ and ⟨⟨ o ⟩⟩) as well as in the entire (usually long) signal of word/sentence. Generally, the warping range is large for vowel classes, small for consonant classes. For example, for slowly spoken ⟨⟨ yokozuna ⟩⟩, usually only vowel segments are lengthened while the length of consonant segments rarely changes. Such nonuniform change in temporal structure is called nonlinear time warping.

3.2.2.2 Acoustic Feature Vector

The gray-scaled sound spectrogram is originally a three-dimensional object with its height being the energy intensity. A section sliced at some time position in parrallel to the frequency axis can thus be observed as a two-dimensional frequency-energy pattern, i.e., short-time spectrum. The spectrum is illustrated in Figure 3.6. From this understanding, the sound spectrogram is treated as a sequence of short-time spectra.

---

1. Yokozuna is the rank of grand champion wrestler in Japanese sumo wrestling.
2. The expression ⟨⟨ ∗ ⟩⟩ denotes phoneme symbols (class names).

Amazon / Zentian Limited
Exhibit 1021
Page 27

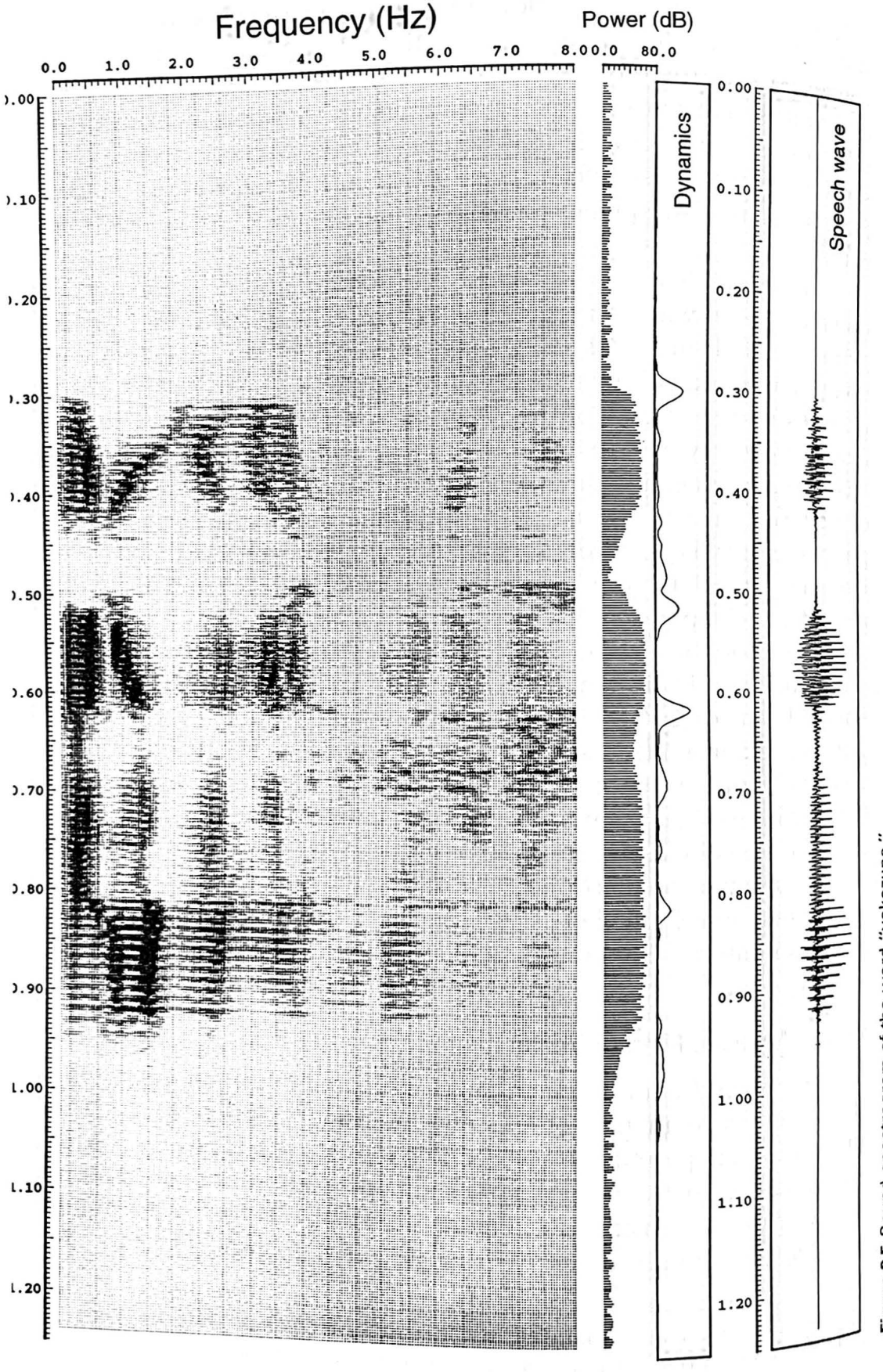


Figure 3.5 Sound spectrogram of the word "yokozuna."

Amazon / Zentian Limited
Exhibit 1021
Page 28

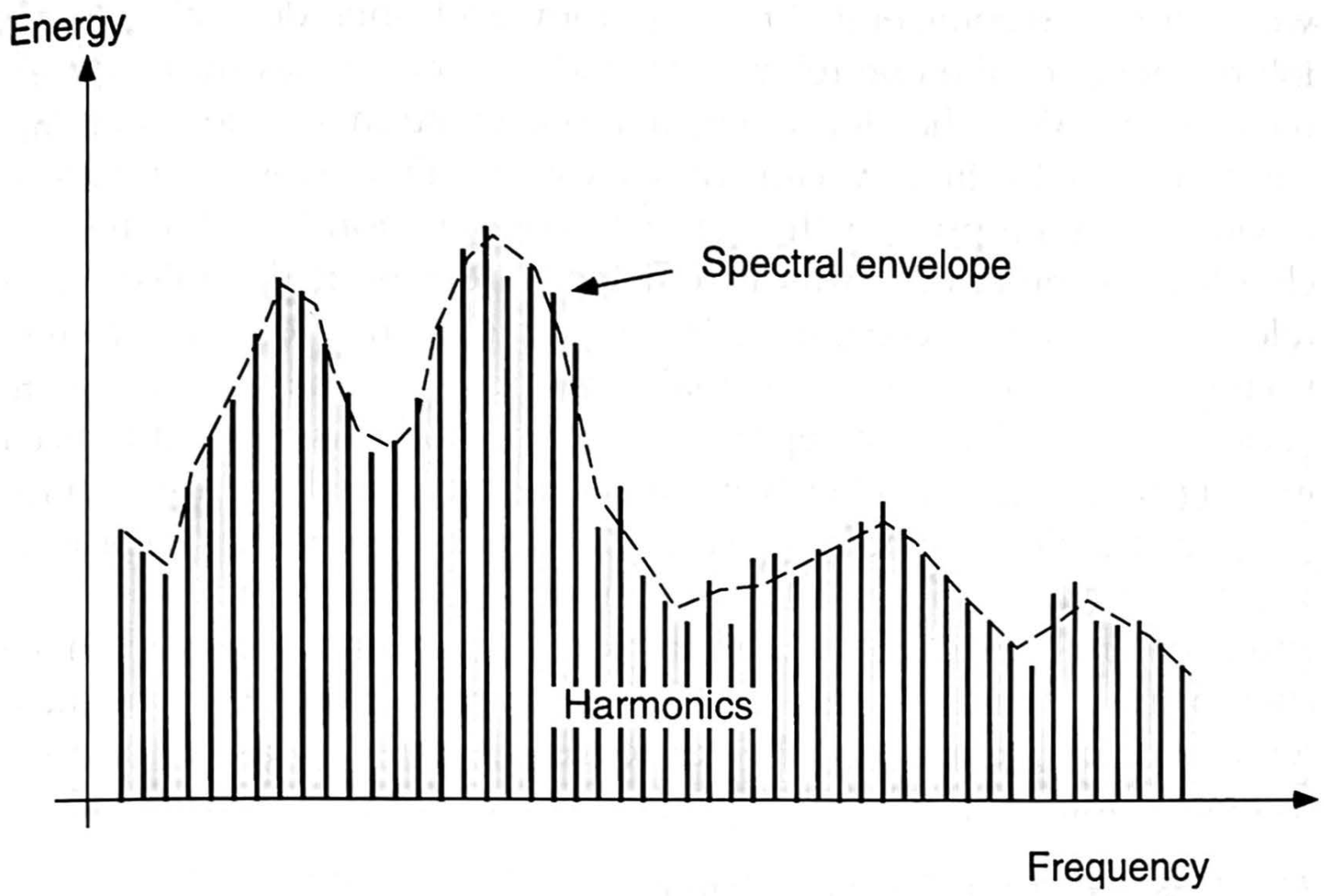


Figure 3.6 Short-time spectrum.

The short-time spectrum of Figure 3.6 shows two noticeable structures: spectral envelope and harmonics. The spectral envelope is an outline structure that smoothly varies and is mainly determined by the status (shape) of articulators such as lips and tongue. In Figure 3.6, the peaks of the spectral envelope correspond to acoustic resonance frequencies, which are determined by the articulator status and are referred to as *formants*. It is easy to observe that phoneme classes are determined by controlling the status of the articulators. Linguistic information, which is the result of articulator behavior (articulation) and thus important for speech recognition, is naturally included mainly in the spectral envelope. In particular, formants are considered a principal factor that determines the identity of vowel classes. In the sound spectrogram of Figure 3.5, one can observe that the formant structure varies as the phoneme class changes, or in other words, as the articulatory status changes. On the other hand, harmonic structure is caused by the vibration of the vocal cords; specifically, the spikes forming the harmonic structure appear at multiples of the fundamental frequency, which is the frequency of the vocal cords' vibration. This periodic structure is considered to mainly convey speaker identity and prosodic information.

Using the short-time spectra for speech recognition as they are

Amazon / Zentian Limited
Exhibit 1021
Page 29

would not be recommended because they each include both types of information, i.e., the one relevant to and the one irrelevant to speech recognition. Also, the direct digital representation of the short-time spectrum results in a vector of excessive dimensions,[3] and it is obviously inappropriate with regards to computation load for the later classification process. A way to efficiently represent the information relevant to speech recognition in low-dimensional features (feature vector) is obviously needed. Actually, in order to realize such feature vectors, various kinds of representation methods have already been investigated, e.g., the filter-bank method motivated by the human peripheral auditory system, linear predictive coding (LPC) to provide an efficient parametric model of the human speech production system, and another model of speech production, the cepstrum method. In the later pages of this chapter, the feature vectors derived from these representation methods will be collectively referred to as the acoustic feature vector.

3.2.2.3 Static Versus Dynamic Nature

As explained above, a single acoustic feature vector (or any feature set that is computed by further modifying the acoustic feature vector) can be considered a static feature in the sense that it represents a status (shape) of articulation at some time point. In contrast, a sequence of acoustic feature vectors or any set of features computed from the sequence are called dynamic features in the sense that they represent the dynamics (behavior) of articulators in the time period corresponding to the sequence. Generally, vowels such as ⟨⟨ a ⟩⟩ and ⟨⟨ i ⟩⟩, which are each produced by their own constant shape of the articulators, are characterized by static features. Such consonants as ⟨⟨ k ⟩⟩ and ⟨⟨ t ⟩⟩, which are each produced by their own rapid moving of the articulators (especially the lips and tongue) during a few tens msec, are characterized by dynamic features.

Usually, the dimensionality of an acoustic feature vector is fixed for every task and system design. Such a fixed-dimensional vector is often called a static pattern. In contrast, the duration of a speech signal or its corresponding sequence of acoustic feature vectors is essentially variable. Such a variable-durational pattern is called a dynamic pattern.

The definitions of "static" and "dynamic" in *static/dynamic feature* are different from those in *static/dynamic pattern*, and this

3. The sound spectrogram shown in Figure 3.5 was computed as a sequence of 256-dimensional short-time power spectra for the frequency range of 10 kHz.

Amazon / Zentian Limited
Exhibit 1021
Page 30

inconsistency is incovenient and confusing. However, because this terminology has been commonly used in the speech recognition field, we shall continue to use it here.

3.2.3 Variety of Recognition Tasks

Even if a system input is assumed to be a speech signal, the system has to handle various types of speech inputs due to the variety of speaking styles. In daily-life speech communication, people rarely speak in the isolated word manner; rather, they speak connected words fluently and continuously and their speech often does not conform to proper grammar. On the other hand, from the viewpoint of system design, the isolated word recognition that attempts to recognize a word spoken in the isolated word-by-word mode is more productive than continous speech (or connected word) recognition, and also the recognition of read speech is more tractable than the recognition of conversational speech, which does not often conform to grammatical rules. It seems that achieving tractability for system design conflicts with acceptability/comfort for system users (speakers). Figure 3.7 illustrates the relationship between speaking conditions (styles) and resulting characteristics of speech signals and also shows the taxonomy, with regard to this relationship, of speech recognizers. In general, the conditions in the left side are acceptable for system design but not for system users; the conditions in the right side are acceptable for system users but not for system design. The figure shows that there are many possibilities of system selection, ranging from limited-vocabulary isolated-word speaker-dependent recognition to open-vocabulary speaker-independent recognition.

3.2.4 Recognition Mechanism

3.2.4.1 Example Task Setting

Let us explain a recognition procedure by using the most fundamental example framework of recognizing words spoken in the isolated mode.

Figure 3.8 is a diagram of procedures employed in our example isolated-word recognizer. The recognizer is a modular system, as in Figure 3.3. An input speech is assumed to be a single word, such as "yokozuna," that is spoken as an answer to the question 'What do you call a grand champion sumo wrestler in Japanese?' Let us assume that our recognition task contains M word classes; C_j $(j = 1, \cdots, M)$.

Amazon / Zentian Limited
Exhibit 1021
Page 31

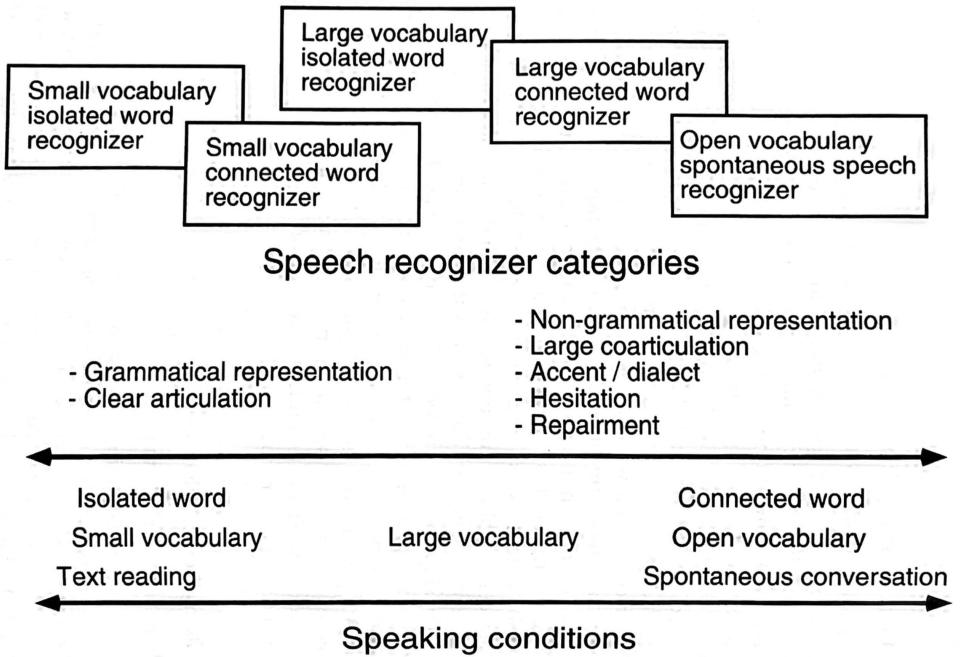


Figure 3.7 Relationship between speaking styles and speech sound characteristics.

Amazon / Zentian Limited
Exhibit 1021
Page 32

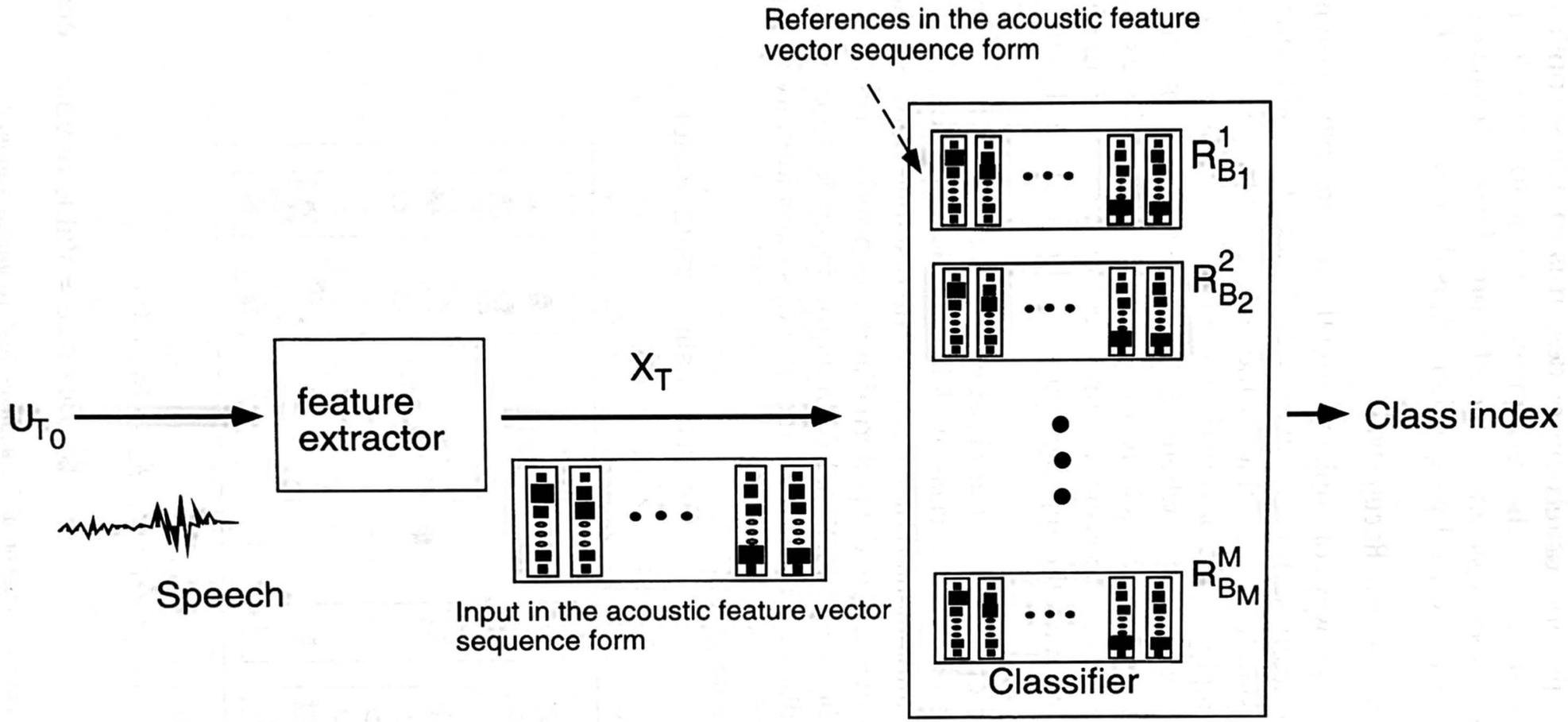


Figure 3.8 Diagram of isolated-word recognition.

Amazon / Zentian Limited
Exhibit 1021
Page 33

Accordingly, the goal of recognizer design is to adjust (optimize) the trainable parameters of the recognizer, aiming to encode an input speech to a correct class among the M word classes. Let us denote an input word speech wave by U_{T_0}, where T_0 is the duration of U_{T_0}.

3.2.4.2 Distance-Based Recognition

There are various ways of implementing the feature extraction module. In this introductory section, we avoid details about possible implementations and simply assume that in the feature extraction module the utterance sample U_{T_0} is converted to acoustic feature vector sequence $X_T = (\mathbf{x}_1, \cdots, \mathbf{x}_\tau, \cdots, \mathbf{x}_T)$, where T is the duration of the vector sequence, $\mathbf{x}_\tau$ is the acoustic feature vector at time index τ in the sequence, and the acoustic feature vectors are of F-dimensions. Figure 3.9 illustrates the representation of the acoustic feature vector sequence.

A basic operation of the classification module is to compare an input feature pattern and class models designed beforehand, and it is traditionally implemented by measuring the distance between the input feature pattern and a model pattern that corresponds to one class and is defined in the same functional form as the input feature pattern. The smaller the distance for C_j, the larger the C_j's class identity of the input

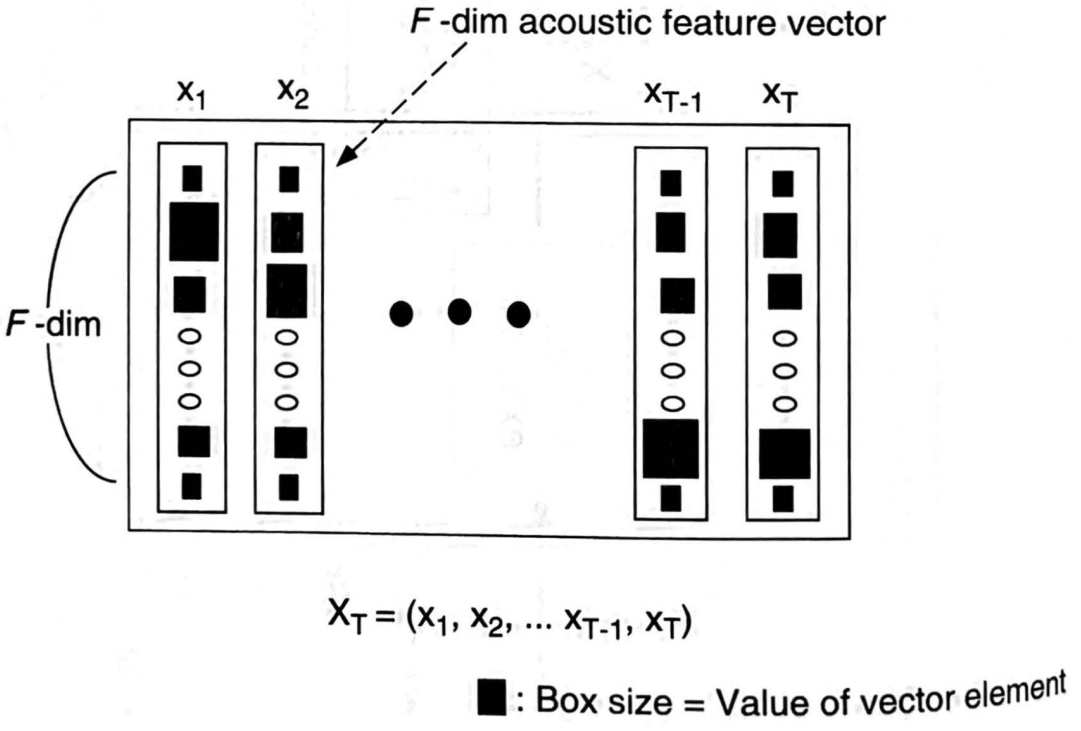


Figure 3.9 Schematic explanation of the acoustic feature vector sequence.

Amazon / Zentian Limited
Exhibit 1021
Page 34

feature pattern, i.e., the degree to which the input belongs to C_j. According to the conventional approach, we consider here a distance-based recognizer that employs this distance-based classification strategy.

Let us define a formal representation of C_j's class model by acoustic feature vector sequence $R^j_{B_j} = (\mathbf{r}^j_1, \cdots, \mathbf{r}^j_\tau, \cdots, \mathbf{r}^j_{B_j})$, where B_j is the duration of the vector sequence, $\mathbf{r}^j_\tau$ is acoustic feature vector at time index τ in the sequence, and $\mathbf{r}^j_\tau$'s are all of F-dimensions. A class model defined as $R^j_{B_j}$ is traditionally referred to as a reference pattern in the speech recognition field because it works as a reference for comparison with the input. In practical designs, multiple references are often assigned to each class. However, for simplicity of discussion, we consider here the simplest case of using a single reference pattern for each class. A classification rule used by our recognizer then becomes

$$C(X_T) = C_i, \quad \text{iff } i = \arg\min_j g_j(X_T; R^j_{B_j}) \tag{3.1}$$

where $g_j(X_T; R^j_{B_j})$ is the distance between X_T and $R^j_{B_j}$.[4]

3.2.4.3 Distance Computation Based on Dynamic Time Warping

Speech samples of a single word differ in duration from sample to sample. Also, as cited above, the durational warping rates differ from segment (such as phoneme) to segment. Thus, the distance computation between dynamic input/reference patterns is not so trivial. Recall that distance is usually defined between static patterns. Among many possible solutions to this difficulty, a dynamic-programming (DP)-based method has been traditionally employed in the speech recognition field.

Figure 3.10 illustrates the process of DP-based distance computation between input X_T and reference $R^j_{B_j}$. In the figure, each dot indicates where the distance between the two corresponding acoustic feature vectors, that of X_T and that of $R^j_{B_j}$, is computed. Such a dot is usually called a matching node, and the corresponding distance is called local distance. We then define path distance by accumulating the local distance values along some matching path that consists of the matching nodes. Because the temporal structure is essentially one-way (past-to-future), matching paths are expected to basically expand in the left-to-right manner. In Figure 3.10, an example of a matching path is also

4. The definition of this distance will be discussed in Section 3.2.4.3.

Amazon / Zentian Limited
Exhibit 1021
Page 35

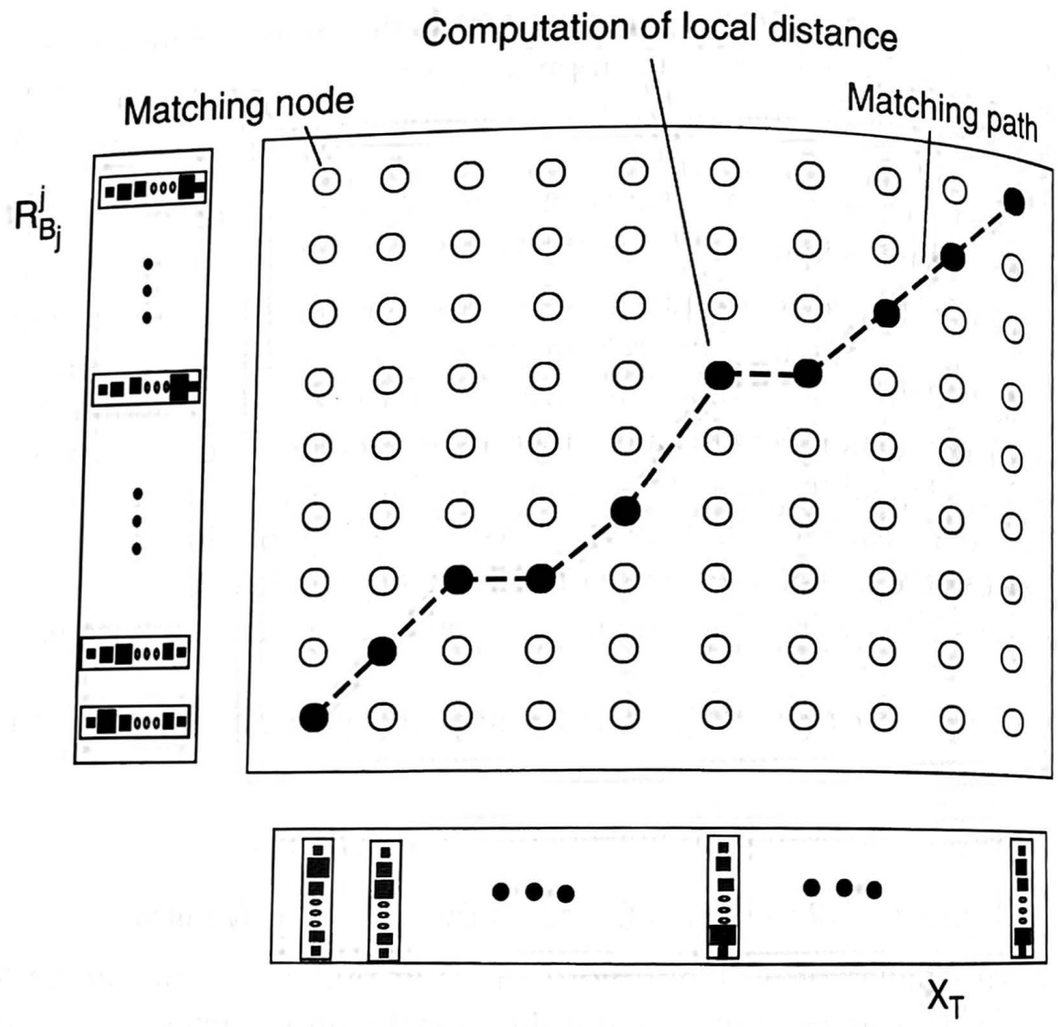


Figure 3.10 Distance computation based on nonlinear time warping.

illustrated by the broken line. Each path has its own path duration and corresponding distance. Consequently, path distance can be considered a duration-normalized distance measure between the dynamic patterns. Obviously, there are many possible ways of drawing the matching paths, and one can observe many different path distance values, one for each matching path. The most natural choice among the many possibilities is to use the minimum path distance, and this special selection is indeed a distance concept that has widely been used for speech recognition.

In the following paragraphs, we describe the procedure of computing the minimum path distance in more detail.

First, we define the distance measure between the acoustic feature vectors in a more concrete manner. As a natural candidate of distance in the fixed (F-dimensional) vector space, we can select a common distance measure such as Euclidean distance. By way of example, let us consider the squared Euclidean distance δ_{j,θ_τ} between $\mathbf{r}_i^j$, which is the

Amazon / Zentian Limited
Exhibit 1021
Page 36

t-th component vector of $R_{B_j}^j$, and $\mathbf{x}_{\theta_\tau}$, which is the θ_τ-th component vector of X_T and is also selected with the matching node corresponding to $R_{B_j}^j$ along path θ. Because this distance is defined locally at every time index over the acoustic feature vector sequence, it is often called local distance. Accordingly, path distance, which is computed by accumulating the local distances along a path, is provided as

$$D_\theta(X_T, R_{B_j}^j) = \sum_{\tau=1}^{T} \delta_j^{\theta_\tau} \tag{3.2}$$

Even though the temporal structure of an utterance sample can warp nonlinearly, such an extreme and unnatural matching as the one between the beginning feature vector of the input and the final feature vector of the reference pattern does not usually occur. Thus, we consider possible matching paths only in some reasonably preset restricted region when searching for the best path having the minimum

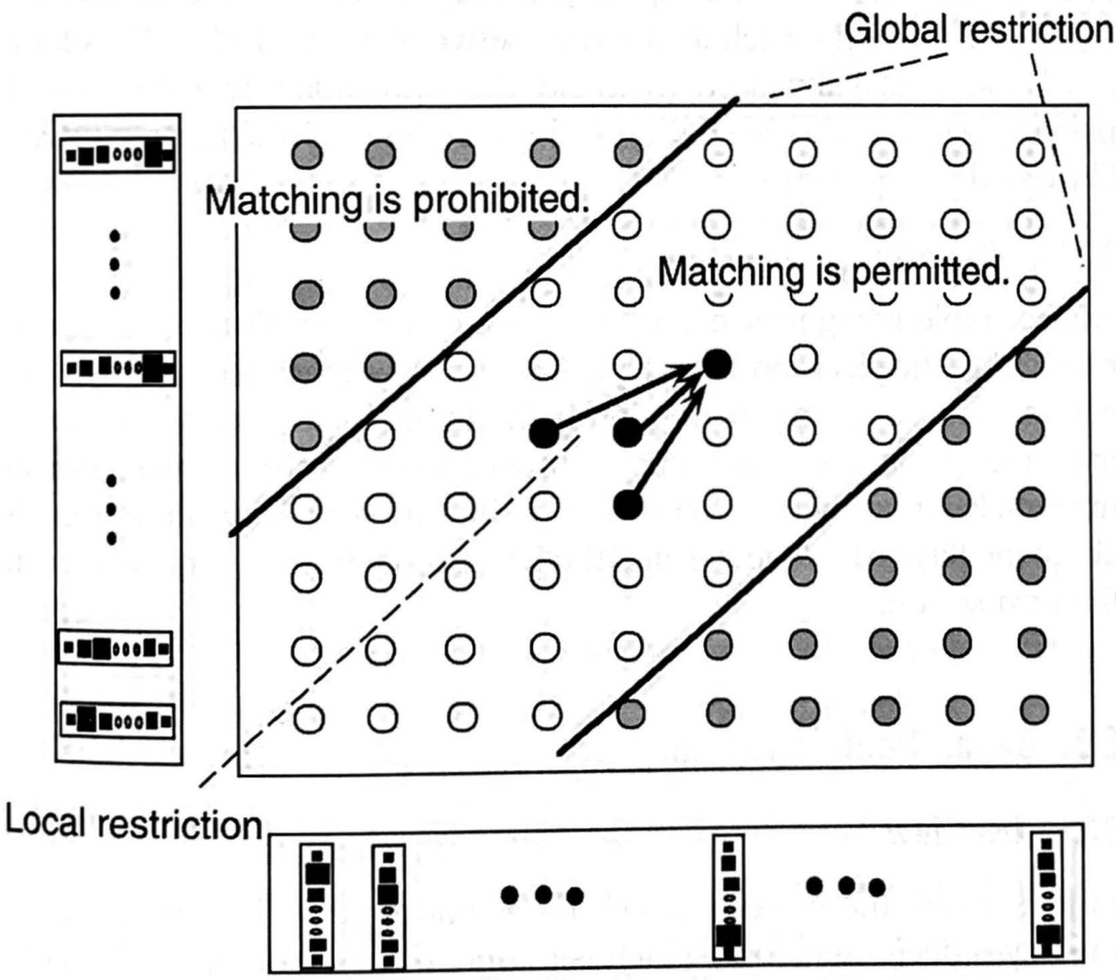


Figure 3.11 Restrictions to finding the minimum distance path: global restriction and local restriction.

Amazon / Zentian Limited
Exhibit 1021
Page 37

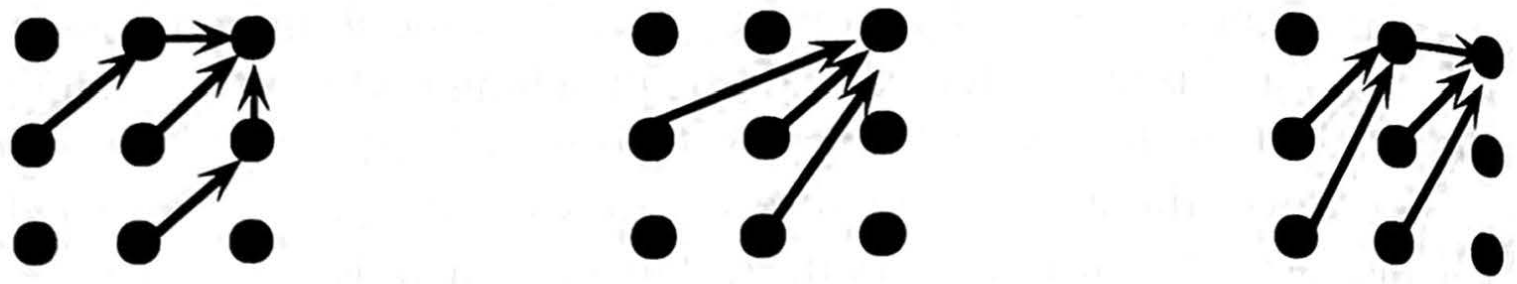


Figure 3.12 Examples of implementing local restriction.

path distance.[5] Figure 3.11 illustrates the concept of restriction in finding the minimum distance path. There are mainly two types of restriction: global restriction and local restriction. Global restriction is applied to the entire input pattern, as shown by the hatched region in the figure. This is mainly used to avoid extreme warping from the global point of view in the matching space. On the other hand, local restriction is set to control the direction of path extension at every matching node. Let us consider an example case that $\mathbf{x}_{t_0}$ is matched to $\mathbf{r}_{t_1}^j$. A question here is whether $\mathbf{x}_{t_0}$ can be matched to $\mathbf{r}_{\hat{t}}^j$'s ($\hat{t} = 1, \cdots, t_1 - 1$). Based on the one-way (forward direction) nature of temporal signals such as speech, it may be reasonable to prohibit backward matching, such as the one between $\mathbf{x}_{t_0}$ and $\mathbf{r}_{t_1-1}^j$. This local restriction is actually used to avoid the unreasonable extension of paths at every node. Several typical examples of local restriction are illustrated in Figure 3.12.

3.2.4.4 Remarks

Our example recognizer makes its classification decision according to rule (3.1). The decision rule using distance comparisons seems quite natural and conforms to our intuition. In fact, the rule is a typical implementation of the discriminant function (DF) approach to a fundamental classification decision rule, i.e., the Bayes decision rule. Details about the Bayes decision rule and the DF approach will be introduced in the next secion.

3.3 Bayes Decision Theory

3.3.1 Overview

In most cases, the present speech recognition technology is based on the Bayes decision theory, which is a fundamental principle of pattern

---

5. The best path is also called minimum distance path.

Amazon / Zentian Limited
Exhibit 1021
Page 38

classification: the final stage of recognition, i.e., classification, is generally performed based on this classical theory. We shall focus in this chapter only on the fundamentals of the theory. A more mathematical formalism of the theory can be found in other sources (see [4, 5]).

A speech pattern is essentially dynamic, possessing nonlinearly-warping temporal structure. Accordingly, the pattern discussed in this chapter should be dynamic. However, handling dynamic patterns would require us to define new functions such as the probability of dynamic patterns and would make this introductory chapter unnecessarily complicated. Therefore, let us assume that our input pattern to the recognizer is static, as do most textbooks (e.g., [4]). Our example task here is thus to classify a static F-dimensional pattern $\mathbf{x} = ((x_1, \cdots, x_f, \cdots, x_F)^T)$ to one of M classes $(C_j; j = 1, \cdots, M)$, where x_f is the f-th component of $\mathbf{x}$ and T is vector transpose. In addition, we assume that the input pattern $\mathbf{x}$ is directly put to the classification stage; which means that here we consider neither the feature extraction subprocess nor the segmentation subprocess. We also assume that $\mathbf{x}$ is a random vector that is characterized by unknown but fixed continuous distribution functions.

We assume that in order to accomplish the task, we are given (1) a classifier consisting of a set of designable (adjustable in the design stage) parameters Λ $(=\{\lambda_j\}_{j=1}^M)$ and (2) a set of design samples $\mathcal{X}$ $(=\{\mathcal{X}_j\}_{j=1}^M = \{\mathbf{x}_n\}_{n=1}^N)$ that is used for designing the classifier, where λ_j is the classifier parameters assigned to C_j, $\mathcal{X}_j$ is the set of design samples of C_j, N is the total number of design samples, and $\mathbf{x}_n$ is the nth design sample of $\mathcal{X}$ that is also assumed to belong to one of $\{\mathcal{X}_j\}$. Note here that λ_j can be either scalars, vectors, or a combination of scalars and vectors; in addition, the design samples are randomly extracted from the corresponding class sample distribution and they are mutually independent.

In the Bayes decision theory, the occurrence of a sample is treated as a probabilistic phenomenon. The probability with which the observation (observed sample) $\mathbf{x}$ occurs is denoted by $p(\mathbf{x})$; the probability with which an event of C_j occurs, i.e., the a priori class probability of C_j, is denoted by $P(C_j)$; the probability with which an event of C_j occurs conditioned by the observation of $\mathbf{x}$, i.e., the a posteriori probability, is denoted by $p(C_j|\mathbf{x})$. The classification that encodes $\mathbf{x}$ to a class is fundamentally performed by measuring the *a posteriori* probability $p(C_j|\mathbf{x})$. Basically, various types of classification decision rules are possible. Among the many, let us consider the most natural and

Amazon / Zentian Limited
Exhibit 1021
Page 39

simplest rule,

$$C(\mathbf{x}) = C_i, \quad \text{iff} \quad i = \arg\max_j p(C_j|\mathbf{x}) \tag{3.3}$$

where $C(\mathbf{x})$ is the classification operation. This rule requires $\mathbf{x}$ to be classified as the class having the largest a posteriori probability. This rule is called the Bayes decision rule, and a precise execution of the rule is shown to lead to the optimal, minimum classification error probability condition, where the probability with which an incorrect classification decision occurs is the smallest. In practice, however, we rarely use this rule directly because the direct measurement of $p(C_j|\mathbf{x})$ is quite difficult. The rule is usually used in the form

$$C(\mathbf{x}) = C_i, \quad \text{iff} \quad i = \arg\max_j p(\mathbf{x}|C_j)P(C_j) \tag{3.4}$$

which is derived according to the Bayes rule of probability

$$p(C_j|\mathbf{x}) = \frac{p(\mathbf{x}|C_j)P(C_j)}{p(\mathbf{x})} \tag{3.5}$$

where $p(\mathbf{x}|C_j)$ is the class-conditional probability.

Figure 3.13 illustrates a task of classifying two-class one-dimensional ($S = 1$) samples, where each class distribution is unimodal. An optimal class boundary, which corresponds to the minimum classification error probability condition, exists at the crossing point of the two distribution curves. Moving the class boundary anywhere from the crossing point results in an increase in the error probability.

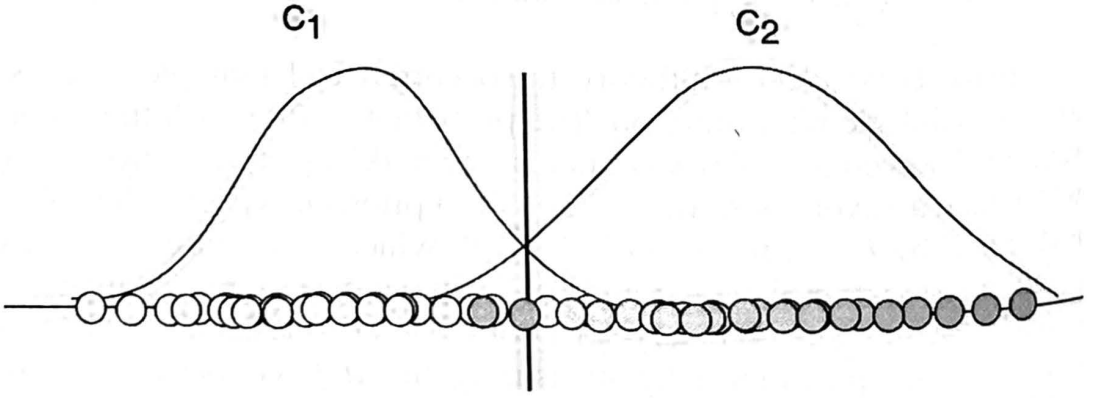


Figure 3.13 A two-class one-dimensional classification task.

Amazon / Zentian Limited
Exhibit 1021
Page 40

As cited above, the precise execution of the Bayes decision rule leads to the optimal pattern classification. Thus, a central issue in classifier design becomes an accurate estimation of the probability functions included in the rule and also a faithful execution of the entire operation of the rule. Approaches to optimal classification include (1) maximum likelihood estimation approach (MLE approach), (2) Bayesian approach, and (3) discriminant function approach (DF approach). The design concepts underlying the MLE approach and the Bayesian approach are similar, and both aim at the accurate estimation of the probability functions. By contrast, the DF approach aims at a faithful execution of the decision operation by formalizing the decision rule as a loss (risk) minimization problem.

We shall explain the three approaches in more detail in the following sections.

3.3.2 Maximum Likelihood Estimation Approach

From (3.4), it is clear that the execution quality of the Bayes decision rule depends on the estimation accuracy of the a priori class probabilities and the class-conditional probabilities. The MLE approach then aims to achieve accurate estimation by computing the most likely estimates of the probabilities.

In principle, $P(C_j)$ is determined by some knowledge about the task that is independent of the observation of samples. Therefore, in the MLE approach, the a priori probabilities are assumed to be provided by a designer's knowledge and preferences, and the main design efforts are made to achieve the most likely (maximum likelihood) estimates of the class-conditional probabilities.

In order to estimate the class-conditional probabilities, the MLE approach employs two assumptions: (1) letting the functional form of the class-conditional probabilities be a function of the set of unknown but fixed (not random) variables (parameters) (for C_j, the class-conditional probability is expressed by using λ_j as $p(\mathbf{x}|C_j, \lambda_j)$), and (2) using the available design samples $\mathcal{X}_j$ in place of $\mathbf{x}$, which is unobservable in the design stage. Consequently, the design goal of the approach becomes finding the parameters that correspond to the maximum value of $p(\mathcal{X}_j|C_j, \lambda_j)$, i.e., the maximum likelihood estimate defined as

$$\hat{\lambda}_j \equiv \arg\max_{\lambda_j} p(\mathcal{X}_j|C_j, \lambda_j) \tag{3.6}$$

Amazon / Zentian Limited
Exhibit 1021
Page 41

Recall that the design samples are mutually independent. Thus (3.6) is rewritten as

$$\hat{\lambda}_j = \arg\max_{\lambda_j} \prod_{n=1}^{N} p(\mathbf{x}_n|C_j, \lambda_j)\mathbf{1}(\mathbf{x}_n \in C_j) \tag{3.7}$$

where $\mathbf{1}(\cdot)$ is an indicator function that takes 1 for *true* and 0 for *false*. Intuition here indeed supports that $\hat{\lambda}_j$ is a reasonable solution to λ_j in the design strategy of using only available samples. The parameters λ_j determine the model function $p(\cdot|C_j, \lambda_j)$. In the MLE approach, we aim at making the value of $\max_{\lambda_j} p(\mathcal{X}_j|C_j, \lambda_j)$ as large as possible (as likely as possible) by repeating the selection of parameter type (type of λ_j) and the (3.7)-based optimal status search.

The computation of (3.7) is relatively easy. For example, when $p(\cdot|C_j, \lambda_j)$ is a Gaussian function, the maximum likelihood estimate of λ_j, i.e., $\hat{\lambda}_j$, can be computed with a simple gradient computation (in terms of λ_j) of $p(\cdot|C_j, \lambda_j)$. The principle of the expectation-maximization (EM) method [6], which is widely used for designing HHMs, also essentially relies on the MLE concept (the EM method will be further explained in Section 3.5). Indeed, based on the simplicity of the estimation, this approach has long been used as the most standard concept in a wide range of speech recognizer designs. However, in practice, it has also been pointed out that this simple MLE approach suffers from several serious problems. One basic problem is that a true functional form of the probability is usually more complex than the Gaussian form and is not accessible. To cope with this problem, a mixture Gaussian function, consisting of multiple Gaussian functions, is often used. However, because the true functional form is unknown, the problem of determining the structure of the mixture function, or in other words, the number of the Gaussian-form components, obviously remains unsolved. Another problem is that the number of available design samples is limited. In this limited resource environment, the estimation of the probability is almost always incomplete, even though the true functional form of the probability is known. Moreover, according to (3.7) the incompleteness is caused evenly by individual estimation errors, each computed at one design sample. That is, high accuracy in the estimation of the probability function, which corresponds to a large value of $p(\mathcal{X}_j|C_j, \lambda_j)$, does not necessarily mean high classification accuracy. There could even be an extreme case in which the estimated probability function is highly accurate in a region far from the class boundary and less than accurate near the class boundary, resulting in

Amazon / Zentian Limited
Exhibit 1021
Page 42

inaccurate classification decisions. The mismatch between the dotted curve and the hard curve in Figure 3.3 clearly illustrates this basic problem of the approach.

3.3.3 Bayesian Approach

In the Bayesian approach, we aim at computing the a posteriori probabilities used in (3.3) through the estimation of class-conditional probabilities. This design concept is apparently the same as that of the MLE approach. However, the Bayesian approach is clearly distinct from the MLE approach, because of the assumption that the classifier parameters λ_j's, each determining the functional form of the class-conditional probability, are treated as random variables. The Bayesian approach assumes that $\{\lambda_j\}$ possesses its own a priori probability function, and its design goal thus becomes computing the a posteriori probability for every class by estimating this a priori probability function, which is associated with the classifier parameter, and the class-conditional probability function through the observation of design samples.

The functional form of the class-conditional probability is assumed to be known, and it is expressed as $p(\mathbf{x}|C_j, \mathcal{X}_j)$ on the premise that all of the information used for estimating the functions are obtained through sample observation. Also, based on the assumption that λ_j is a random vector (variables), the approach introduces *a priori* probability $p(\lambda_j|C_j)$. Accordingly, the class-conditional probability $p(\mathbf{x}|C_j, \mathcal{X}_j)$ is rewritten as

$$p(\mathbf{x}|C_j, \mathcal{X}_j) = \int p(\mathbf{x}, \lambda_j|C_j, \mathcal{X}_j)d\lambda_j$$

$$= \int p(\mathbf{x}|\lambda_j)p(\lambda_j|C_j, \mathcal{X}_j)d\lambda_j \tag{3.8}$$

where $p(\lambda_j|C_j, \mathcal{X}_j)$ is an estimate, which is obtained by observing the samples of $\mathcal{X}_j$, of the a priori probability $p(\lambda_j|C_j)$. Therefore, if $p(\lambda_j|C_j, \mathcal{X}_j)$ peaks sharply around $\hat{\lambda}_j$, and $\hat{\lambda}_j$ is a true value of λ, we can achieve an approximation of $p(\mathbf{x}|C_j, \mathcal{X}_j)$ as

$$p(\mathbf{x}|C_j, \mathcal{X}_j) \approx p(\mathbf{x}|\hat{\lambda}_j) \tag{3.9}$$

The estimation method summarized above is quite reasonable, but it requires a large amount of computation for most functional forms of

Amazon / Zentian Limited
Exhibit 1021
Page 43

the a priori probability $p(\lambda_j|C_j, \mathcal{X}_j)$. Therefore, in practice, the Bayesian approach is usually applied only when the probability functions are assumed to belong to an exponential family, which is a special probability (and probability density) function group that enables us to reduce the computation load of the above estimation. Members of the exponential family include the normal (Gaussian) function, the Poisson function, and many other familiar distribution functions.

Similar to the MLE approach, the Bayesian approach has been used for speech pattern recognition. However, in reality, design results based on the Bayesian approach are often less than satisfactory: the true distributions of samples (probability functions) are usually more complicated than the exponentional families, and it is unrealistic to collect enough design samples to accurately estimate the probability functions.

3.3.4 Discriminant Function Approach

The MLE and Bayesian approaches aim at computing the a posteriori probabilities, which are components of the decision rule (3.3). In contrast, the DF approach attempts to embed the entire rule in a design process. In this approach, in place of the *a posteriori* probability, a discriminant function is used to measure the degree to which an input sample belongs to some class, i.e., sample's class identity. As a discriminant function, one can use various familiar measures such as distance and similarity as well as probability. In the old days of insufficient computation power, this approach was actually a major design concept because of its simple computation. Embodiments of the approach include the linear discriminant function method and the perceptron learning method, which is a classic version of artificial neural network (ANN) learning.

For a long time, attempts to apply this classic approach to speech recognition were rare. However, after the reemergence of research interest in ANNs, the approach came to attract greater interest from speech researchers. The ANN-based recognition methods introduced in Part II of this handbook actually belong to the DF approach. Therefore, in this subsection, we shall elaborate on the fundamentals of the DF approach.

3.3.4.1 Example Task and Decision Rule

Let us consider the task of classifying M-class static samples by using the classic and most fundamental implementation of the DF approach, i.e., the linear discriminant function method. Given $\mathbf{x}$, a discriminant

Amazon / Zentian Limited
Exhibit 1021
Page 44

function $g_j(\mathbf{x};\Lambda)$ is defined for C_j as

$$g_j(\mathbf{x};\Lambda) = \sum_f \lambda_j[f] \cdot \mathbf{y}[f] \tag{3.10}$$

where $\mathbf{y} = (1, x_1, \cdots, x_F)$, $\mathbf{y}[f]$ is the f-th element of $\mathbf{y}$, and accordingly λ_j is an $(F+1)$-dimensional weight vector and $\lambda_j[f]$ is the f-th element of λ_j. Note that the discriminant function is linear in terms of λ_j. As in the MLE and Baysian approaches, λ_j also forms a model of C_j in the DF approach. Let us refer to a classifier that is composed of the set $\{\lambda_j\}$ $(= \Lambda)$ as a linear discriminant function classifier. Figure 3.14 illustrates the computation of the linear discriminant function classifier, i.e., (3.10). A classification rule then becomes

$$C(\mathbf{x}) = C_i, \quad \text{iff} \quad i = \arg\max_j g_j(\mathbf{x};\Lambda) \tag{3.11}$$

It should be noted here that classification rule (3.1) is essentially equivalent to rule (3.11), and distance-based classification decisions such as (3.1) accordingly belong to the DF approach.

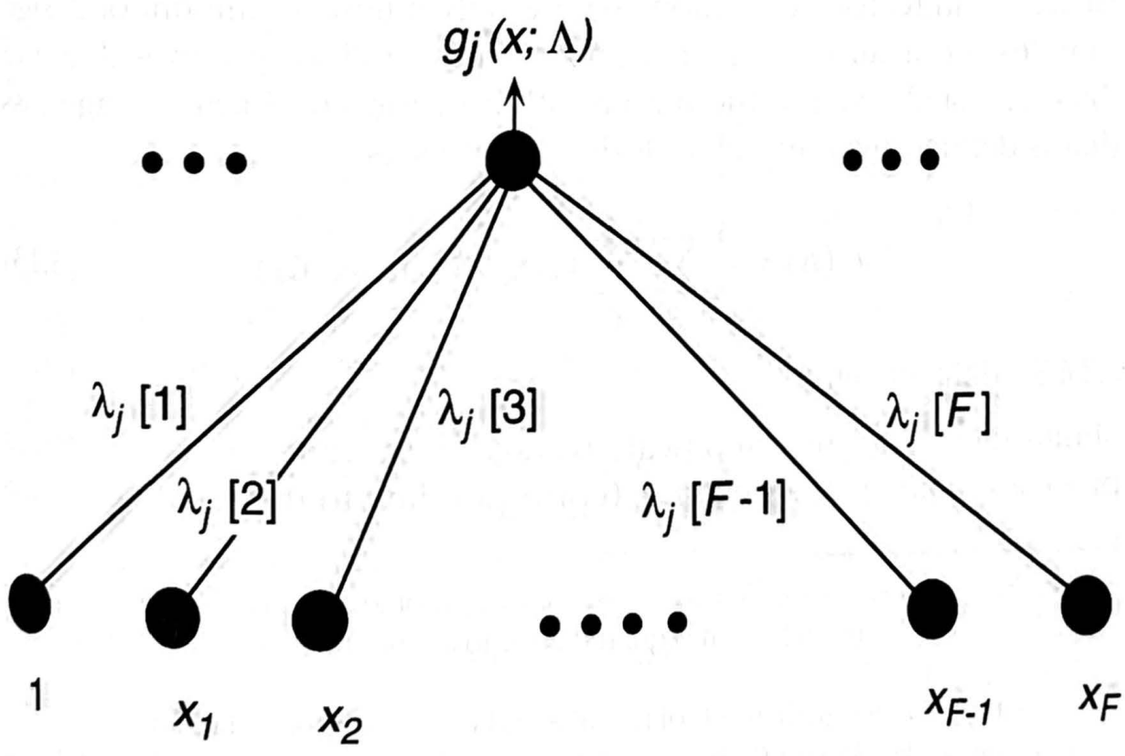


Figure 3.14 Linear discriminant function classifier.

Amazon / Zentian Limited
Exhibit 1021
Page 45

3.3.4.2 Loss

In contrast to the likelihood maximization in the MLE approach, we aim at minimizing loss,[6] which corresponds to the evaluation of a classification result of each design sample. There are various implementations of DF-based design (e.g., see [4]), and each of them is characterized by the selection of loss function form and a minimization algorithm. Generally, the design of discriminant functions, which is based on loss minimization, is executed competitively in terms of classes. Thus, the design method of the DF approach is often referred to as discriminative learning and competitive learning. Also, the concept of using the classification results in the training (loss minimization), which is employed by the DF-based methods, can be considered a learning mechanism using supervision information. The DF-based classifier design methods are therefore sometimes referred to as supervised learning.[7]

Let us express a loss for $\mathbf{x}$ ($\in C_k$) as $\ell_k(\mathbf{x};\Lambda)$. The classifier that we now try to design is expected to perform as correctly as possible for all future samples. An ideal way of design thus becomes the minimization of the expected loss that is defined over the entire set of possible samples as

$$L(\Lambda) = \sum_k P(C_k) \int \ell_k(\mathbf{x};\Lambda) p(\mathbf{x}|C_k) d\mathbf{x} \tag{3.12}$$

However, in realistic conditions where only a limited amount of design samples are available, this minimization is obviously impossible and thus we usually aim at the minimization of the empirical average loss that is defined over available design samples as

$$L_0(\Lambda) = \frac{1}{N} \sum_k \sum_n \ell_k(\mathbf{x}_n;\Lambda) \mathbf{1}(\mathbf{x}_n \in C_k) \tag{3.13}$$

3.3.4.3 Optimization

Minimization of the empirical average loss can be considered a process of aiming at an *optimal* (corresponding to the minimum status

6. The loss is sometimes called risk, cost, or design objective. Typical loss functions include the perceptron loss, the squared error loss, and the classification error count loss (e.g., [4]).
7. In contrast, the design methods of the MLE and Bayesian approaches are categorized as *unsupervised learning* because they do not utilize (feedback) the evaluation results of system outputs (classification decisions) for training.

Amazon / Zentian Limited
Exhibit 1021
Page 46

of the empirical average loss) status of Λ, i.e., *optimization*. Useful optimization methods here include the relaxation method, the gradient method, and simulated annealing (e.g., [7, 8]).

Originally, these are methods for optimizing optimizable parameters with a general class of objective functions (criteria) such as the empirical average loss. However, not all of the methods can necessarily achieve true optimization, i.e., global optimization (global minimization of objective function). Some of them result only in local optimization (local minimization of objective function); furthermore, some attempt to simply reduce the value of an objective function.

Generally, the relaxation methods lack a sufficient mathematical basis to guarantee the minimization of objective function, and its effectiveness is sometimes limited.

The gradient methods such as the steepest descent method are mathematically shown to lead to at least a local minimum of objective function by following the gradient directions of objective function. Actually, this method has been widely used because of its high practicality based on a simple computation mechanism.

Simulated annealing is a method that attracted recent research interest in the ANN field. An infinite repetition of adjustment has been demonstrated to lead to the minimum state of objective function in the probabilistic sense, though its execution needs high computational load.

The optimization methods can also be classified into *batch* optimization and *adaptive* (sequential) optimization. Batch optimization explicitly computes the empirical average loss over all of the design samples and adjusts Λ to achieve the minimum status of the empirical average loss. The steepest descent method, introduced above, is a typical example of batch optimization. On the other hand, adaptive optimization selects a limited number of design samples (usually one sample) randomly from a pool of design samples and adjusts Λ to reduce the individual loss (i.e., $\ell_k(\cdot\,; \Lambda)$) for the selected sample(s) in place of the explicit computation of the empirical average loss. Typical examples of adaptive optimization are methods based on stochastic approximation (e.g., [7]) and the probabilistic descent method [9, 10].

In a practical environment where only a limited number of design samples are available, the difference between batch optimization and adaptive optimization is small. However, adaptive optimization offers the potential of adapting a designed classifier to an unknown application environment, based on its sample(s)-by-sample(s) training mechanism.

Amazon / Zentian Limited
Exhibit 1021
Page 47

3.3.4.4 Design of Linear Discriminant Function Classifier

From among the many possible selections of loss and optimization, we shall specifically select perceptron loss and batch optimization for designing our linear discriminant function classifier. It should be noted that the perceptron loss used here originated from perceptron learning, which played a historic role in neural network research, and also that the design method described in the following paragraphs is fundamentally equivalent to a neural network learning algorithm called the generalized error-correction rule (e.g., [11]).

First, let us initialize Λ to arbitrary random numbers. Obviously, in this initial stage, there is no guarantee that the classifier works accurately.

By using the initialized classifier, we next make a classification decision for $\mathbf{x}_n$ ($\in C_k$) that is randomly selected from design sample set $\mathcal{X}$. Obviously, it cannot be guaranteed that the classification in this initial stage is complete. The classifier works correctly in some cases and makes errors in other cases. Then we perform the adjustment (training) of the classifier parameters in a twofold fashion. That is, when $\mathbf{x}_n$ is correctly classified, we do not adjust Λ; when $\mathbf{x}_n$ is misclassified, in other words, at least one discriminant function of a class other than C_k reads a larger value than that of C_k (we refer to such a class as a confusion class), we adjust the parameters of both C_k and the confusion classes so as to increase the discriminant function of C_k and decrease the discriminant functions of the confusion classes.

The adjustment procedure described above is mathematically formulated in the following way. First, the loss for design sample $\mathbf{x}_n$ is expressed as

$$\ell_k(\mathbf{x}_n;\Lambda) = \frac{1}{N_n}\sum_{i\in\Im}(-g_k(\mathbf{x}_n;\Lambda) + g_i(\mathbf{x}_n;\Lambda)) \tag{3.14}$$

where $\Im$ is the confusion classes defined as

$$\Im = \{C_j; g_j(\mathbf{x}_n;\Lambda) > g_k(\mathbf{x}_n;\Lambda)\} \tag{3.15}$$

and N_n is the number of the classes included in $\Im$. Since the adjustment occurs only in the case of misclassification, the empirical average loss becomes

$$L_p(\Lambda) = \sum_n \sum_k \ell_k(\mathbf{x}_n;\Lambda)\mathbf{1}(\mathbf{x}_n \in C_k)\mathbf{1}(\Im \neq \emptyset) \tag{3.16}$$

Amazon / Zentian Limited
Exhibit 1021
Page 48

The adjustment based on the gradient optimization is then

$$\Lambda(t+1) = \Lambda(t) - \epsilon \nabla_{\Lambda} L_p(\Lambda) \tag{3.17}$$

where t is the time index of adjustment repetition, ϵ is a small positive constant, and $\Lambda(t)$ represents the status of Λ at t. Consequently, the adjustment procedure for the weight vector of each class is given as

$$\lambda_i(t+1) = \lambda_i(t) - \epsilon \sum_n \sum_k \frac{\partial \ell_k(\mathbf{x}_n; \Lambda)}{\partial \lambda_i} 1(\mathbf{x}_n \in C_k) 1(\Im \neq \emptyset) \tag{3.18}$$

where

$$\frac{\partial \ell_k(\mathbf{x}_n; \Lambda)}{\partial \lambda_i} = \begin{cases} -\mathbf{y} & (\text{for } i = k) \\ \dfrac{1}{N_n}\mathbf{y} & (\text{for } i \neq k) \\ 0 & (\text{otherwise}) \end{cases} \tag{3.19}$$

3.3.4.5 Remarks

Compared with the MLE and Bayesian approaches that attempt to design class models separately (in terms of class), the DF-based design introduced above seems to be more direct in its classification decision rule: the loss minimization is a straight mechanism of feeding back classification decisions to the design stage, and also the definition of loss in (3.14) is a quite reasonable approximation of the search operation max in the decision rule.

However, it was shown that this linear discriminant function classifier led to accurate classification only in the simple task setting called the linear-separable case. Its performance under realistic task conditions is actually unsatisfactory. Moreover, the relationship between the minimum status of $L_p(\Lambda)$ of (3.16) and the desirable, minimum classification error status is unclear. The gradient computation for $L_p(\Lambda)$ is also mathematically inadequate: $L_p(\Lambda)$ is not differentiable in Λ due to $1(\Im \neq \emptyset)$ of (3.16). Therefore, further improvement of the DF approach is clearly needed.

3.4 Acoustic Feature Extraction

3.4.1 Filter-Bank

Based on psychoacoustic findings, it is known that the human auditory system has a function of filter-bank-like frequency analysis of input

Amazon / Zentian Limited
Exhibit 1021
Page 49

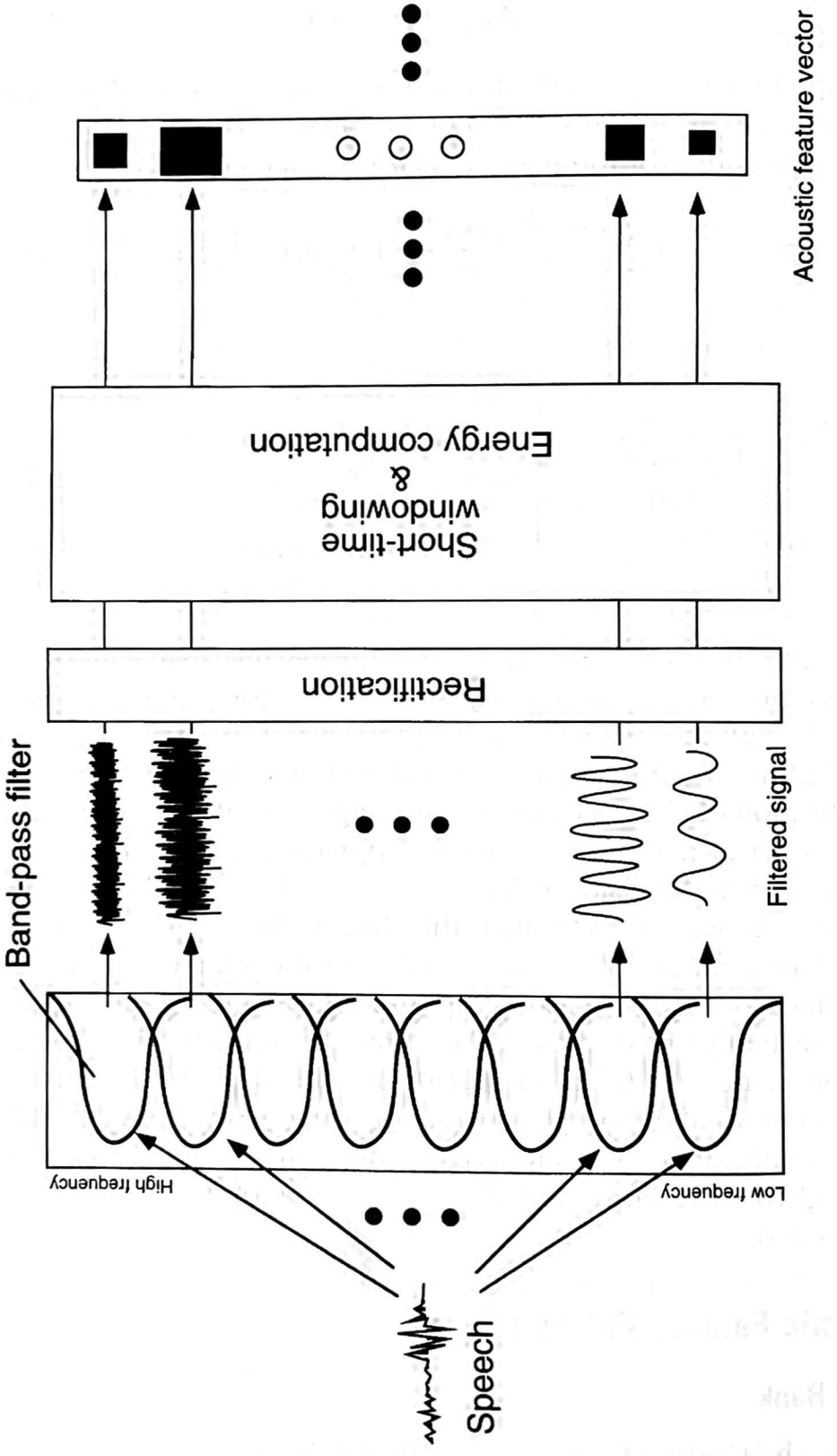


Figure 3.15 Diagram of the filter-bank feature extractor.

Amazon / Zentian Limited
Exhibit 1021
Page 50

sound (e.g., [12, 13]). Thus, in a direct attempt to emulate this human system, various filter-bank feature extractors have been studied.

Filter-bank feature extraction is illustrated in Figure 3.15. A speech pattern U_{T_0} is passed through a bank of $\mathcal{I}$ bandpass filters. Generally, the individual filters overlap in frequency and are spaced according to a nonuniform frequency scale, such as the Mel scale or Bark scale, that are formed based on perceptual experiments. Each filter outputs a kind of short-time energy that can be observed through its corresponding frequency range.

Implementations of a filter-bank are mainly grouped into two classes: (1) perception-research-oriented attempts in which one emulates the human auditory (cochlea) system with real (infinite impulse response) filtering computation, and (2) engineering-oriented attempts in which one simply approximates the basic function of a filter-bank with Fourier transform.

3.4.1.1 Artificial Cochlea Filter

The human auditory peripheral system consists of the outer ear, the middle ear, the inner ear (cochlea), and the auditory nerve. Figure 3.16 illustrates a block diagram of the system.

The outer ear and the middle ear are basically considered a simple, linear and time-invariant bandpass filter. The cochlea works, in contrast, as a nonlinear, time-variant filter-bank that contains many narrow-band

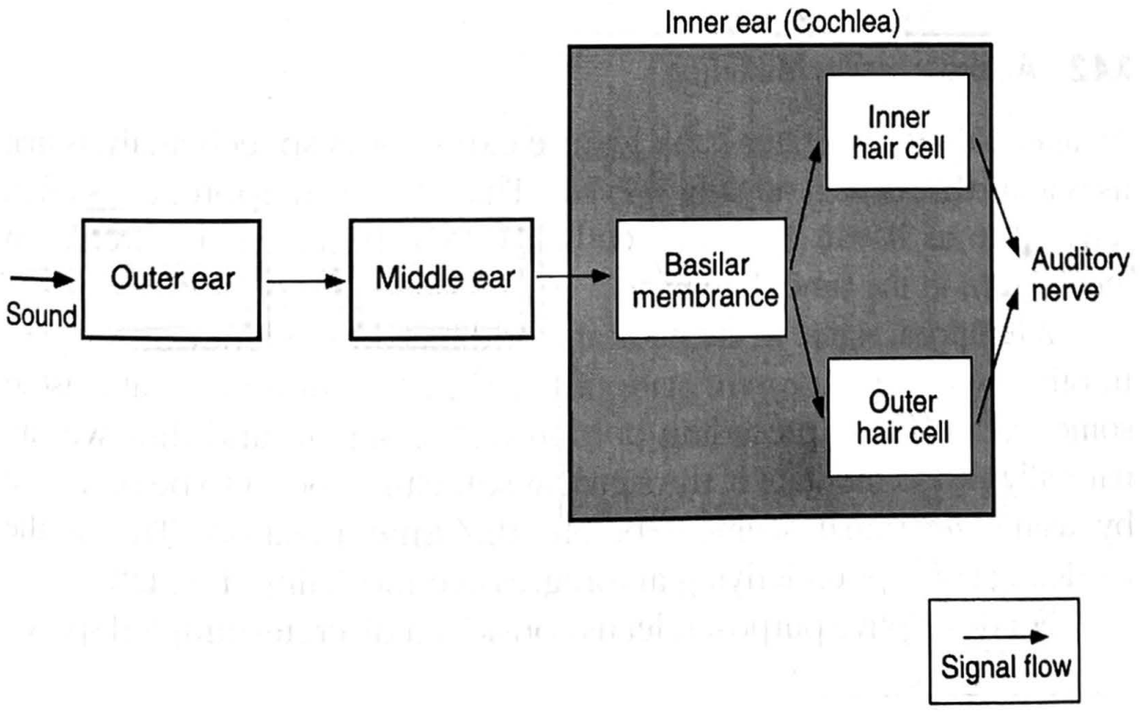


Figure 3.16 Block diagram of the human auditory peripheral system.

Amazon / Zentian Limited
Exhibit 1021
Page 51

(high Q[8]) bandpass filters. The main functions of the cochlea include (1) frequency scaling such as the Bark scale, (2) adaptive Q, which controls filter gain in response to the energy of signals through the filter channel, (3) frequency masking, and (4) rectification. Therefore, it would seem that emulating these functions of the cochlea would provide a direct route to effective modeling.

To date, there have actually been many implementation examples (e.g., [14, 15]). Among them, we briefly introduce in this chapter the artificial cochlea filter-bank that consists of adaptive Q cochlea filters [16]. Figure 3.17 illustrates a block diagram of the adaptive Q cochlea filter-bank. A major concern in this modeling is to simulate the function of the basilar membrane (in the cochlea) filtering, which changes its Q in response to the energy of the input signal. The filter-bank is formed by adding an adaptive Q circuit to a fixed Q cochlea filter that consists of a notch filter and a bandpass filter.

3.4.1.2 Fourier-Transform-Based Filter

The most standard technical approach to the frequency analysis of a temporal signal is to use the short-time Fourier transform. We can easily compute the power spectrum of a windowed, short input speech segment with the transform and can then compute energy in a form similar to bandpass filter output by accumulating spectral elements in a narrow frequency band. Figure 3.18 illustrates a block diagram of this Fourier-transform-based implementation.

3.4.2 Autoregressive Modeling

An alternative to the filter-bank feature extractor is speech analysis that uses autoregressive modeling [17, 18]. This statistical approach is often referred to as linear predictive coding (LPC), based on its prediction mechanism in the time domain.

A temporal signal in the natural world has dependency on the past. In other words, the present state of the signal is influenced at least to some extent by the preceding portion of the signal, and thus we can naturally expect the state of the signal at some time point to be predicted by using the signal segment before the time position. This is the modeling concept underlying autoregressive modeling, i.e., LPC.

For descriptive purposes, let us consider a discrete-sampled speech

8. Q comes from the quality factor that indicates the resonance sharpness. The larger Q is, the narrower is the bandwidth of its corresponding bandpass filter.

Amazon / Zentian Limited
Exhibit 1021
Page 52

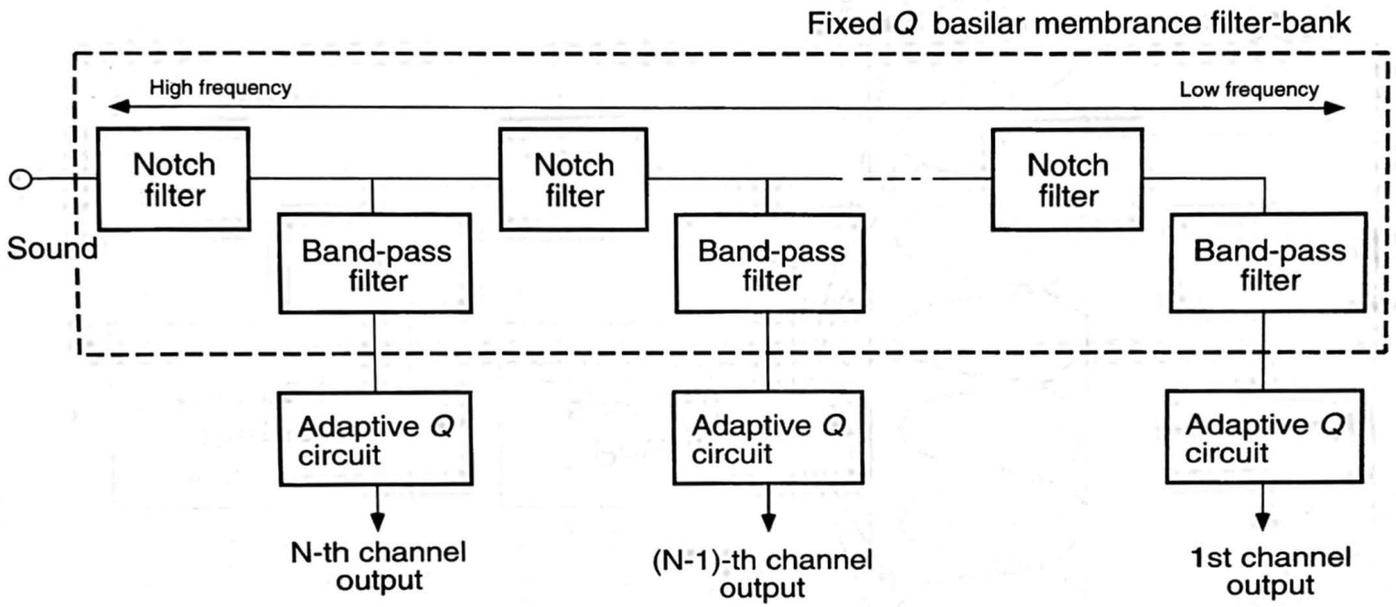


Figure 3.17 Block diagram of the adaptive Q Cochlea filter-bank.

Amazon / Zentian Limited
Exhibit 1021
Page 53

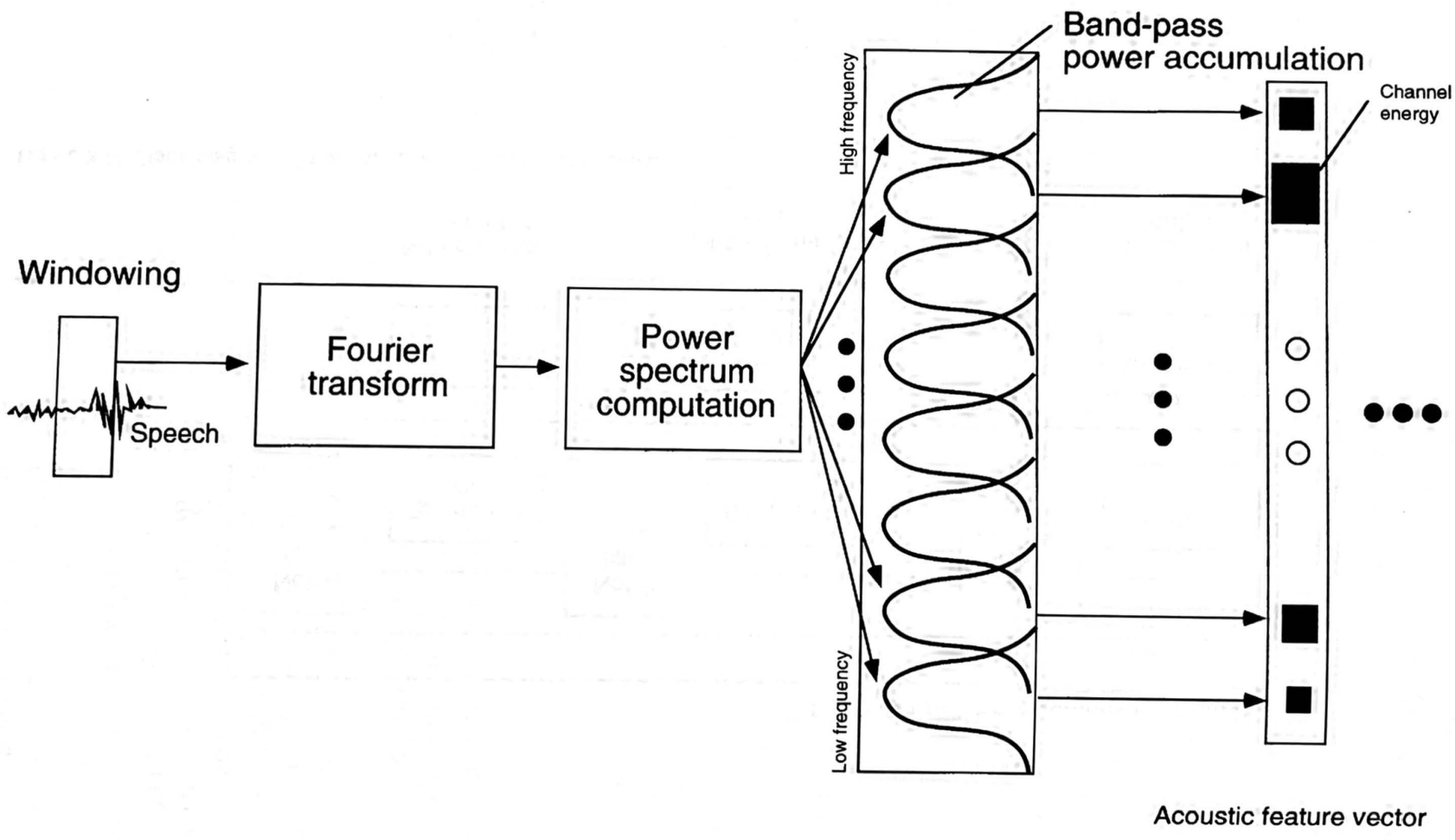


Figure 3.18 Block diagram of the filter-bank implementation using the Fourier transform.

Amazon / Zentian Limited
Exhibit 1021
Page 54

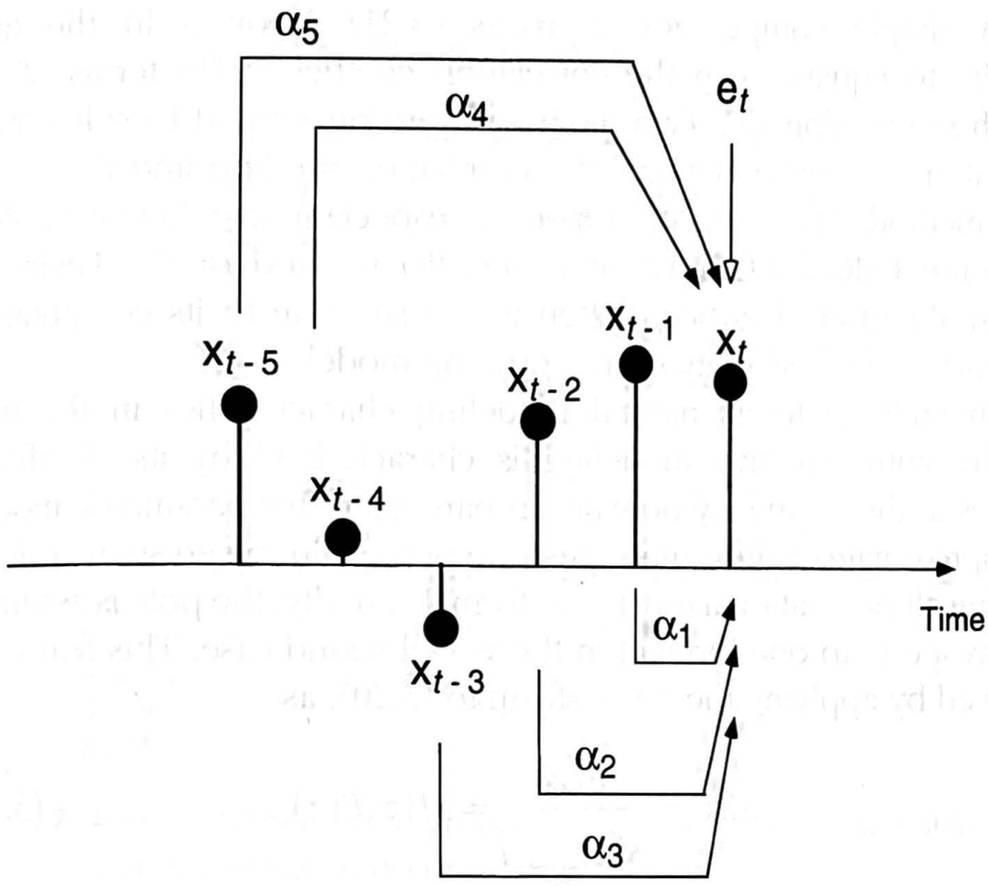


Figure 3.19 The mechanism of autoregressive modeling.

signal, which is a common representation in digital signal processing. The mechanism of the autoregressive modeling is illustrated in Figure 3.19. Discrete speech sample s_t at time point t is modeled as a linear combination of previous samples plus a nonpredictable (independent) additional noise e_t in the following way:

$$s_t = -\sum_{i=1}^{p} \alpha_i s_{t-1} + e_t \tag{3.20}$$

where p is the order of modeling and α_i's are called prediction coefficients. The goal of modeling is naturally to find the $\{\alpha_i\}$ that minimizes the mean-square error calculated over a windowed short speech segment, defined as

$$E_{rr} = \sum_{t=\tau_1}^{\tau_2} \{e_t\}^2 \tag{3.21}$$

where τ_1 and τ_2 are the left (past) and right (future) ends of the time window, respectively.

Amazon / Zentian Limited
Exhibit 1021
Page 55

A simple computation converts (3.21), defined in the time domain, to equations in the correlation coefficient (in terms of the speech signal) domain. Two methods were developed to solve these correlation-domain equations: the correlation method and the covariance method. The correlation method, especially its efficient version, called the PARtial CORrelation (PARCOR) method or the Levinson-Durbin algorithm, has been extensively used due to its computation efficiency and the stability of the resulting model.

In addition to its natural modeling characteristics in the time domain, autoregressive modeling is characterized by its significant features in the frequency domain. In particular, this parametric modeling approximates a short-time spectrum of the windowed speech signal by using all-pole rational functional form. Naturally, the pole is assumed to correspond to one formant in the vowel sound case. This feature is observed by applying the z-transform to (3.20), as

$$S(z) = \frac{E(z)}{\sum_{i=0}^{p} \alpha_i z^{-i}} = H(z)E(z) \tag{3.22}$$

where $S(z)$ and $E(z)$ are the z-transform of the speech signal and the noise, respectively, $H(z) = 1/\sum_{i=0}^{p} \alpha_i z^{-i}$, and $\alpha_0 = 1$. Importantly, the acoustical physics shows that the all-pole system function corresponds to a nonbranching acoustic tube, i.e., a nonbranching vocal tract model in speech. Therefore, from (3.22) we can conclude that the system function (spectrum) of speech signal $S(z)$ can be obtained as the product of the system function of the vocal tract $H(z)$ and the system function of the noise $E(z)$. Figure 3.20 illustrates a vocal tract based on autoregressive modeling. Consequently, noise can be interpreted here as a vocal source (excitation) signal.

From the above descriptions, it is obvious that the prediction coefficients convey the shape information of the vocal tract or, in other words, articulation information and the identity of linguistic class (such as phoneme). This is one of the reasons these coefficients and their derivatives have been widely used as acoustic feature vectors for speech recognition.

3.4.3 Cepstrum Modeling

As introduced in Section 3.2 and illustrated in Figure 3.20, a speech signal can be considered an output of a vocal tract filter ($H(z)$ in the case

Amazon / Zentian Limited
Exhibit 1021
Page 56

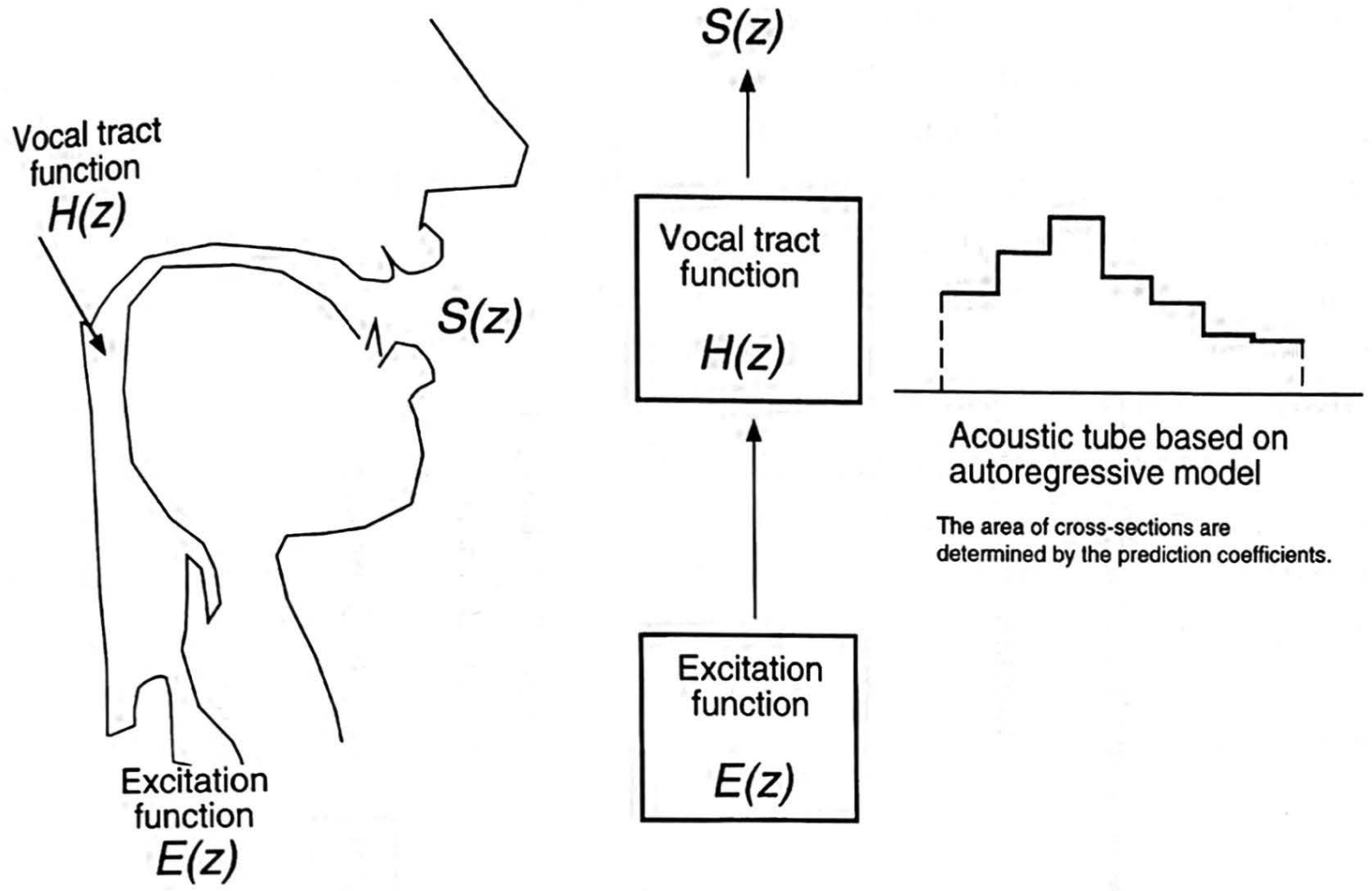


Figure 3.20 A vocal tract based on autoregressive modeling and an illustration of the human speech production system.

of autoregressive modeling) excited by an excitation function ($E(z)$ in the case of autoregressive modeling). For speech recognition, we need to extract only the information based on the vocal tract filter, which conveys linguistic class identity, from the speech signal. The acoustic feature representation/modeling using cepstrum has also been used extensively in the speech recognition field.

Cepstrum is an inverse Fourier transformation of a logarithmic power spectrum. A variable for cepstrum function is then defined as quefrency, which is effectively a time domain variable. Figure 3.21 illustrates the transformation relations. A speech signal is originally a scalar value function (discrete sample sequence after sampling) of time that is defined as the convolution of the excitation signal and the impulse response of a vocal tract filter. Through the Fourier transform, the speech signal is converted to its corresponding power spectrum, which is a function of frequency and is defined as the production of the vocal tract filter spectrum and the excitation spectrum. A simple logarithmic operation converts production to summation. In the log power spectrum domain, the speech spectrum becomes the summation of the two log spectra, that of the vocal tract filter and that of the excitation signal. Cepstrum is accordingly defined as the inverse Fourier transform of the summation of these two spectra.

Amazon / Zentian Limited
Exhibit 1021
Page 57

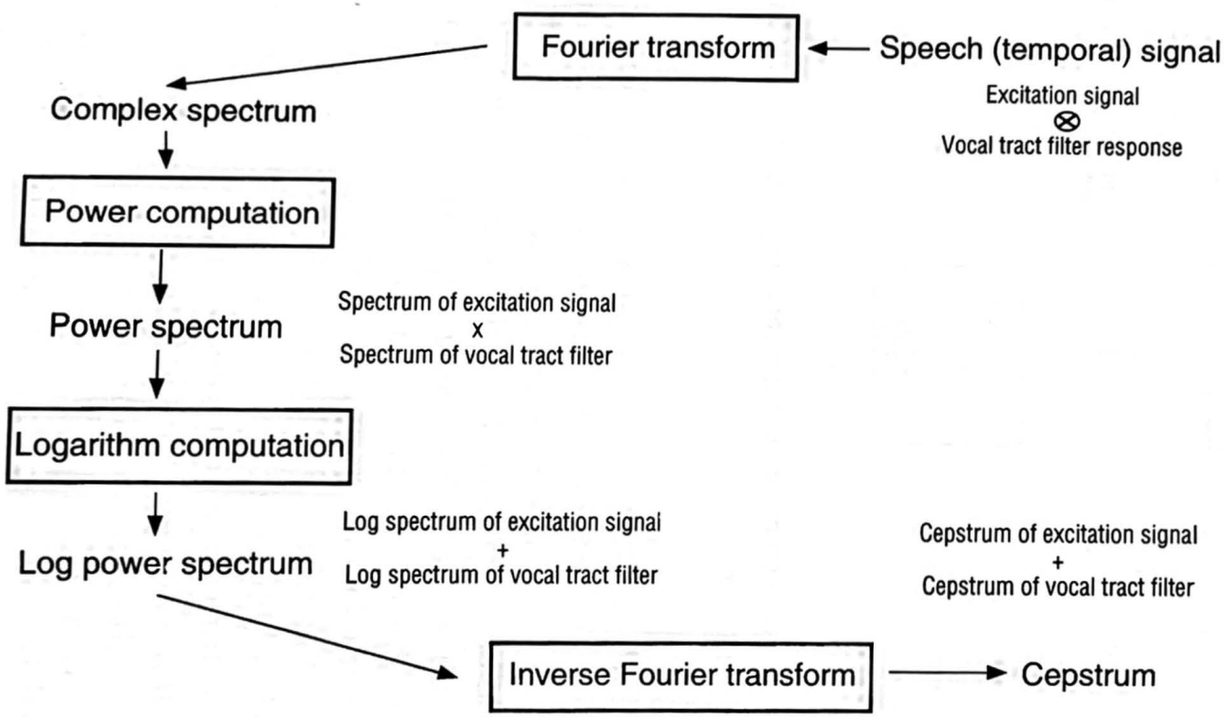


Figure 3.21 Relationships among a speech signal, a vocal tract filter, and a vocal source excitation signal.

Amazon / Zentian Limited
Exhibit 1021
Page 58

As mentioned previously, the (log) spectrum of the vocal tract filter is smooth; the (log) spectrum of the excitation is flat for unvoiced speech segments and periodic for voiced segments. Thus the smooth spectrum of the vocal tract filter is transformed to cepstrum components in the low quefrency region, and the periodic spectrum of the (voiced) excitation signal is transformed to cepstrum components in the high quefrency region. The information localization in the quefrency domain is clearly useful for extracting the vocal tract filter information or, in other words, the speech-recognition-oriented linguistic class information.

Although we have defined cepstrum by using the Fourier transform in the above paragraphs, there are actually two types of cepstrum computation. The first type is based on the Fourier transform and the second type is derived from autoregressive modeling. Below, for reference purposes, we summarize the procedures of these two types of computation.

- **Cepstrum computation based on the Fourier transform**

$$c(n) = \begin{cases} \dfrac{s(n)}{s(0)} - \dfrac{1}{n}\displaystyle\sum_{k=1}^{n-1} k \cdot c(k)\frac{s(n-k)}{s(0)} & (n \neq 0) \\ log\, s(0) & (n = 0) \end{cases} \tag{3.23}$$

  where $s(n)$ is the n-th discrete-sampled speech signal, $c(n)$ is the n-th cepstrum coefficient, $x(n) = c(n) = 0$ for $n < 0$, and $x(0) \neq 0$.

- **Cepstrum computation based on autoregressive modeling**

$$c(n) = -\alpha_n - \frac{1}{n}\sum_{i=1}^{n-1} ic(i)\alpha_{n-i} \quad n \geqslant 1 \tag{3.24}$$

  where $\alpha_0 = 1$ and $\alpha_i = 0$ when $i > p$ (p is the order of autoregressive modeling).

3.4.4 Dynamic Feature Modeling

Originally, the Fourier transform and autoregressive modeling were methods for estimating the spectrum of a statistically stationary system (signal). However, speech signals are obviously non-stationary due to the dynamics of the speech production system, and this dynamics often

Amazon / Zentian Limited
Exhibit 1021
Page 59

plays a key role in the recognition of rapidly changing phonemes such as plosives and liquids. Therefore, as a practical compromise, these spectrum estimation methods are applied to a short speech segment that is assumed to be stationary and is extracted through a short time window (usually of about 20 msec). Consequently, in order to adequately represent the dynamics information for classification, a subsequence of the acoustic feature vectors is often used as an input pattern to the post-end classifier. The length of the subsequence is usually 30 msec–100 msec.

To date, two main methods of dynamics representation have been studied: (1) feature matrix and (2) delta-feature.

The first method simply uses a feature matrix that consists of successive acoustic feature vectors. Figure 3.22 shows an example of a feature matrix. An individual acoustic feature vector is a 16-dimensional vector of filter-bank outputs, and the matrix is composed of a sequence of 15 filter-bank output vectors. The size of the box indicates the normalized value of the filter-bank output (white for negative; black for positive). This type of representation has been widely used in ANN-based speech recognizers [19, 20].

The second method represents the direction of successive feature component change by linear regressive modeling. Figure 3.23 illustrates the concept and computation procedure of this representation. The figure illustrates an example sequence (in time window position) of the

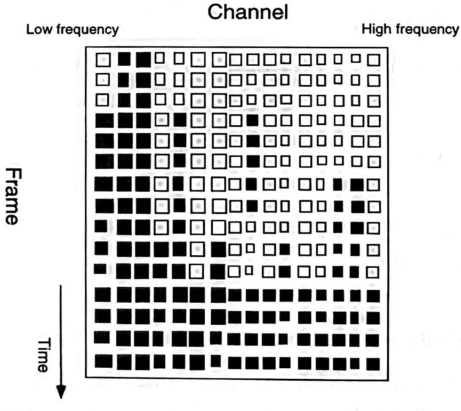


Figure 3.22 Example of a feature matrix used for representing the dynamics of a speech signal.

Amazon / Zentian Limited
Exhibit 1021
Page 60

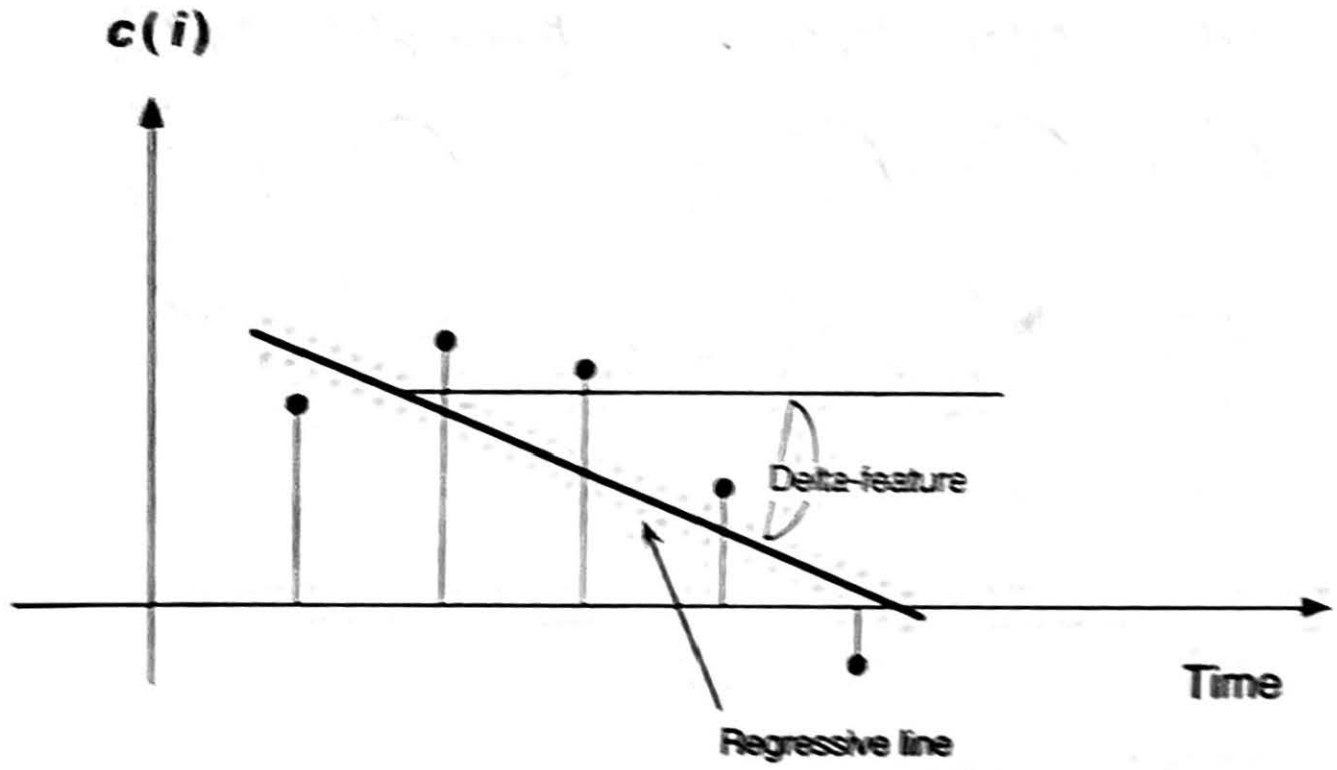


Figure 3.23 Concept of delta-feature used for representing the dynamics of a speech signal.

i-th cepstrum coefficients, $c(i)$. To represent the change direction (i.e., dynamics) in each feature component space, we attempt line fitting or, in other words, linear regressive modeling, over this coefficient sequence. The slant of this regressive function, called delta-feature, indicates the direction and amount of coefficient change. The delta-feature is computed for every original feature component, e.g., i-th delta-cepstrum $\delta(i)_\tau$ for i-th cepstrum $c(i)_\tau$ at time window position τ, and accordingly a new delta-feature vector having the same dimension as its corresponding original acoustic feature vector is computed. For recognition, a new high-dimensional feature vector consisting of the original acoustic feature vector and its corresponding delta-feature vector is used.

3.5 Probabilistic Acoustic Modeling Based on Hidden Markov Model

3.5.1 Principles of Hidden Markov Model

HMM is a system, or process, that probabilistically emits an observable output (observation) according to an unobservable state-transition mechanism. The mechanism is quite useful for modeling the temporal structure of speech signals, and the probabilistic formalization of HMM is also quite effective for modeling the acoustical variety of speech samples, due mainly to the variation in speaker and speaking style. Thus, HMM has become the standard technology in recent speech recognition research.

Amazon / Zentian Limited
Exhibit 1021
Page 61

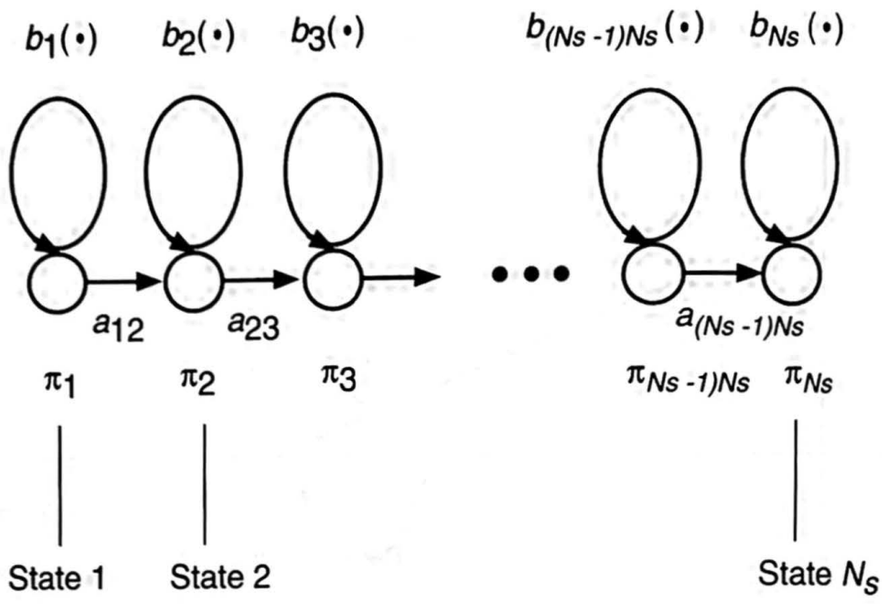


Figure 3.24 Example structure of left-to-right HMM.

Figure 3.24 illustrates a left-to-right HMM that is considered suitable for the modeling of speech signals. The model consists of N_s states, each indicated by a circle. It is assumed that the transition occurs in the probabilistic sense from one state to another or recursively within single state according to the initial state probabilities $\{\pi_i\}$ $(1 \leq i \leq N_s; \pi = \{\pi_i\})$ and transition probabilities $\{a_{ij}\}$ $(1 \leq i,j \leq N_s; A = \{a_{ij}\})$.

It should be emphasized here that this transition cannot be observed. The model is also supposed to emit an observation $\mathbf{o}_t$, which is a vector in the acoustic feature vector space at hand, at observation time index t. The emission of these observations is according to output probabilities $\{b_j(\cdot)\}$ $(b_j(\mathbf{o}_t)$ for observing $\mathbf{o}_t$ at state j; $B = \{b_j(\cdot)\})$. Therefore, an observation sequence O_T is defined as the sequence of model outputs that is probabilisticly produced through the state-transition and its corresponding observation emission; $O_T = \{\mathbf{o}_1\mathbf{o}_2 \cdots \mathbf{o}_t \cdots \mathbf{o}_T\}$. The probability for observing O_T is accordingly provided as

$$\begin{aligned} P(O_T|\lambda) &= \sum_{\text{all } \mathbf{q}} P(O_T|\mathbf{q},\lambda)P(\mathbf{q}|\lambda) \\ &= \sum_{q_1,q_2,\cdots,q_T} \pi_{q_1} b_{q_1}(\mathbf{o}_1) a_{q_1q_2} b_{q_2}(\mathbf{o}_2) \times \cdots \\ &\quad \cdots \times a_{q_{T-1}q_T} b_{q_T}(\mathbf{o}_T) \end{aligned} \tag{3.25}$$

where $\mathbf{q}$ is a state sequence represented as $\mathbf{q} = (q_1q_2 \cdots q_T)$, q_t is the state at t in $\mathbf{q}$, and $\lambda = (A, B, \pi)$.

Amazon / Zentian Limited
Exhibit 1021
Page 62

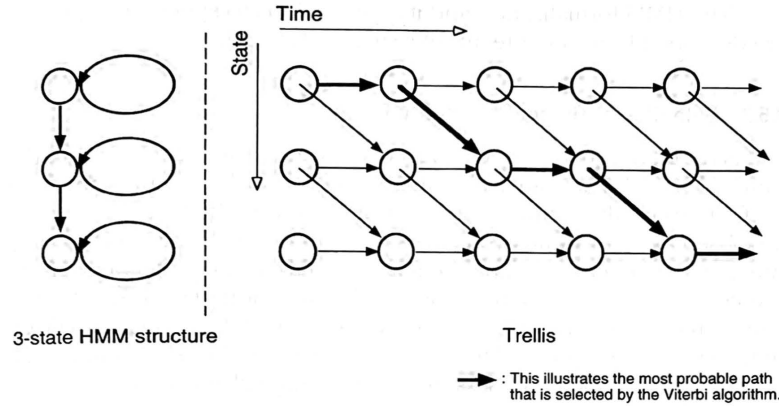


Figure 3.25 Schematic explanation of state-transition: this diagram is usually called a trellis.

It should be noted here that the HMM probability (3.25) is the result of the probability accumulation over all possible state-transition paths. A diagram of these possible state-transition paths is referred to as a trellis and is illustrated in Figure 3.25.

Basically, for speech recognition a single HMM such as the model described above is assigned to every class. To classify an input acoustic feature vector sequence that belongs to some class, we then use the HMM probability, defined in (3.25), to measure the degree to which an input speech sample (its corresponding acoustic feature vector sequence) belongs to each class. A primary goal of classifier design is thus to find the desirable state of HMM parameters λ for every class.

In the MLE and Bayesian approaches, the HMM probability is used as the estimate of class-conditional probability. Accordingly, the design aims at the state of λ that leads to the maximum value of the HMM probability over design samples (of one class) in the class-by-class mode, by substituting each design sample to the observation sequence.[9] On the other hand, the design of the DF approach treats the HMM probability as the discriminant function and aims at the state of HMM parameters (of all classes) that corresponds to the minimum loss condition over the design samples of all prescribed classes.

9. The resulting HMM is the most likely model for observing the available design samples of its corresponding class.

Amazon / Zentian Limited
Exhibit 1021
Page 63

The HMM formalization and its applications to speech recognition are described in detail in textbooks such as [21, 22].

3.5.2 Selection of Output Probability Function

Let us recall that an observed acoustic feature vector corresponds to the output of HMM. There are two major ways of probabilistic interpretation (definition) for the feature vectors: (1) interpreting the feature vector as an observation (output) from the corresponding discrete probability function, and (2) interpreting the feature vector as an observation from the corresponding continuous probability function. The HMM formalized in the former case is referred to as discrete HMM; the one formalized in the latter case is referred to as continuous HMM. We shall summarize these two major model classes in the following subsections.

3.5.2.1 Discrete Model

Discrete HMM is the Markov process that first encodes an observed acoustic feature vector into a code (symbol) and then models the sequence of the codes. Figure 3.26 illustrates the basic structure of discrete HMM. In the figure, the codebook is a set of codes used to encode an input speech pattern into a code sequence. Each code is associated with a static (fixed-dimensional) code vector, which is defined with the same form as the acoustic feature vector.

Generally, the code vectors are designed to minimize a distortion measure defined between the set of the code vectors and the design samples. The design is usually done by using the vector-quantization-based minimum distortion methods.

An input speech pattern is converted into its corresponding code sequence by encoding each acoustic feature vector to its nearest-neighbor code. When there are only two codes, i.e., $\{c_a, c_b\}$, both c_a and c_b are considered outputs of the binomial distribution. Similarly, in the case of using multiple codes (usually a few hundred codes for speech recognition), each code is considered an output of the multinomial distribution. The probability of the discrete HMM is accordingly determined by the output probabilities based on the multinomial distribution, the state-transition probabilities, and the initial probabilities.

3.5.2.2 Continuous Model

Continuous HMM directly models a speech pattern without transforming it into a code sequence. Figure 3.27 illustrates the basic structure of

Amazon / Zentian Limited
Exhibit 1021
Page 64

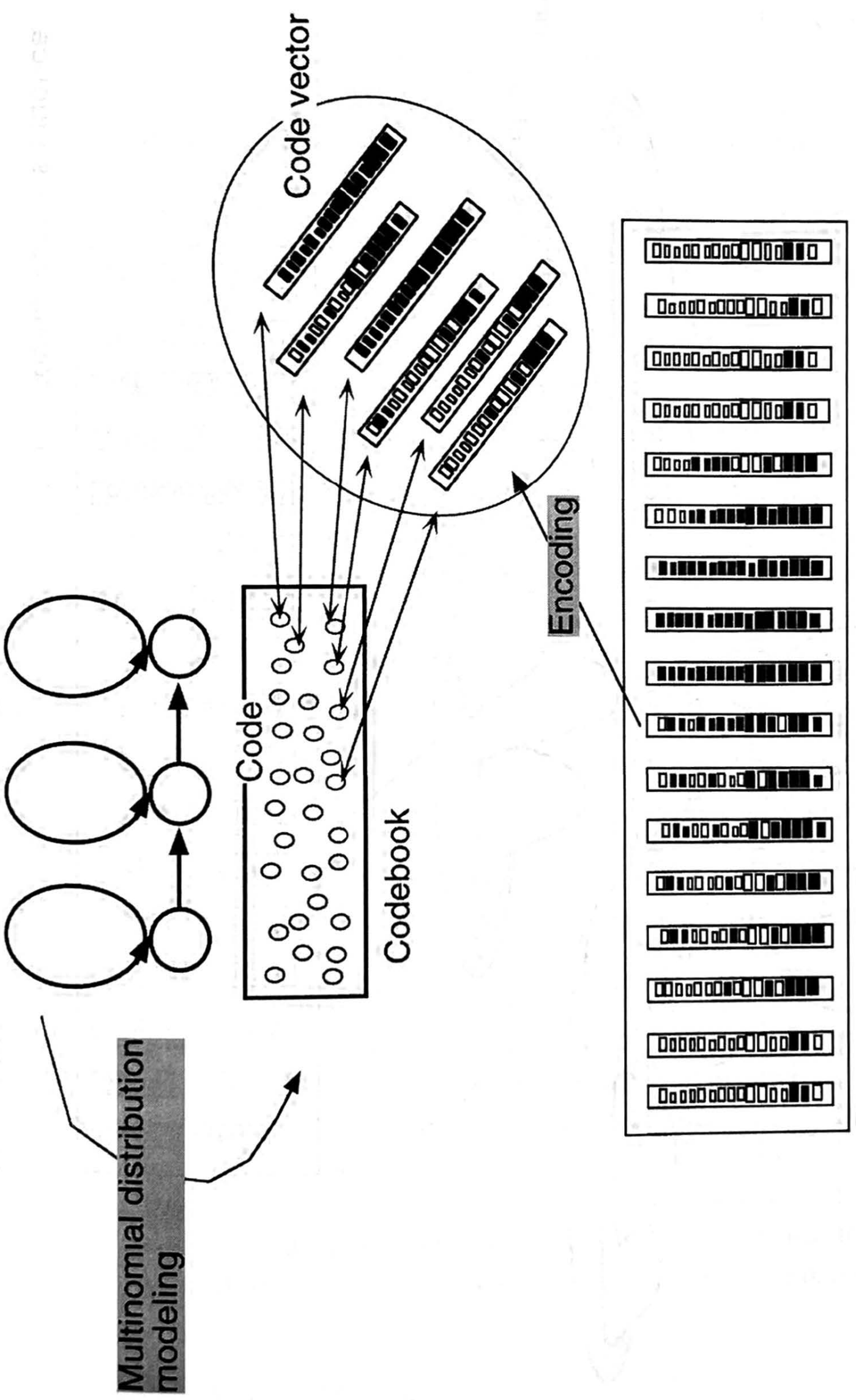


Figure 3.26 Structure of discrete HMM.

Amazon / Zentian Limited
Exhibit 1021
Page 65

Figure 3.27 Structure of continuous HMM.

Amazon / Zentian Limited
Exhibit 1021
Page 66

continuous HMM. Each state is incorporated with a continuous output probability density function, such as the Gaussian probability function and the mixture Gaussian probability function. The acoustic feature vector of a speech pattern is treated as an output of the probability distribution function determined by the continuous probability density function. Therefore, continuous HMM consists of the continuous output probability functions, the state-transition probabilities, and the initial state probabilities.

When the output probability density function is the Gaussian function, the output probability is computed by the mean vector and covariance matrix of the function. A simple analysis in this particular case shows that the mean vector also works as a component acoustic feature vector of a dynamic (variable-length) reference pattern for distance-based classification such as (3.1); furthermore, it was shown that there is a close link between the Gaussian-based probability and the Euclidean distance. These results demonstrated that a classifier using the distance computation based on dynamic time warping, described in Section 3.2, is closely related to a continuous HMM classifier using the Viterbi algorithm (section 3.5.4).

3.5.3 MLE-Based Design Method

The design target of the MLE and Bayesian approaches is to achieve accurate estimates of the class-conditional probabilities and the a posteriori probabilities by computing the HMM probabilities, conditioned by observations of design samples. Most design efforts to date have actually been of the MLE approach and have attempted to gain the maximum likelihood (most probable) estimates of the HMM parameters by applying MLE optimization methods. The design policy based on the MLE concept is clear and reasonable. However, it is impossible to apply a standard and simple version of the MLE optimization, such as the one based on the gradient computation (section 3.3.2), because HMM has an unobservable part, i.e., the state-transition, which is intractable with simple gradient-based optimization. As a common solution to this problem, an advanced version of the MLE method, called the forward-backward method, has been used in a wide range of speech recognition systems.

3.5.3.1 Forward-Backward Method

The procedure that follows is a summary of the forward-backward method.

Amazon / Zentian Limited
Exhibit 1021
Page 67

1. Define the current model as $\lambda = \{A, B, \pi\}$ and initialize it in any reasonable way.
2. Use λ to compute the right-hand sides of the following equations.

$$\bar{\pi}_i = \text{expected frequency (number of times) in state } i \text{ at time } (t = 1)$$
$$= \gamma_1(i) \tag{3.26}$$

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$
$$= \frac{\sum_{t=1}^{T-1} \zeta_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{3.27}$$

$$\bar{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } o_k}{\text{expected number of times in state } j}$$
$$= \frac{\sum_{t=1, s.t. o_t = o_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{3.28}$$

where $\zeta_t(i,j)$ is the probability of being in state i at time t and state j at time $t+1$, given the model and the observation ($\zeta_t(i,j) = P(q_t = i, q_{t+1} = j | O_T, \lambda)$); $\gamma_t(i)$ is the a posteriori probability variable that is the probability of being in state i at time t, given the observation and the model ($\gamma_t(i) = P(q_t = i | O_T, \lambda)$).[10]

10. Here, $\zeta_t(i,j)$ is further rewritten in the iterative form as

$$\zeta_t(i,j) = \frac{P(q_t = i, q_{t+1} = j, O|\lambda)}{P(O|\lambda)}$$
$$= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)}$$

Amazon / Zentian Limited
Exhibit 1021
Page 68

3. Define a reestimated model as $\overline{\lambda} = \{\overline{A}, \overline{B}, \overline{\pi}\}$ by using the left-hand sides of the above equations.
4. Repeat the reestimation of (2)–(3) by replacing λ with $\overline{\lambda}$ until the probability for observing the design samples exceeds a preset threshold.

The computation iteration of the forward-backward method was shown to increase the HMM probability monotonically at least up to its local maximum value unless the initial model is at a critical point of the probability function (e.g., [23, 24]). It is also known that this training optimality is a direct result of the EM method generally used for a wide range of problems of estimating system parameters under the missing-data condition, in which some part of the data is unobservable even though it is necessary for the estimation [6]. Consequently, the forward-backward method is often called the EM method. The method is also sometimes called the Baum-Welch method in honor of its inventors.

3.5.4 Trellis Algorithm and Viterbi Algorithm

In (3.25), the HMM probability was defined as the accumulation of the elemental probabilities such as the output probabilities over all state-transition cases. The forward-backward method is an efficient implementation of the way of computing the exact value of (3.25). Because all of the nodes on the trellis are used in the computation, this method is sometimes referred to as the trellis algorithm.

Even the efficient forward-backward method is time-consuming in such task conditions as large-vocabulary continuous speech recognition. To further reduce the computational load, the method of using only the most probable state-transition path has been widely used. The mechanism of searching for the most probable path is equivalent to

---

$$= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}, \tag{3.30}$$

where $\alpha_t(i)$ is a forward variable defined as $\alpha_t(i) = P(\mathbf{o}_1 \mathbf{o}_2 \cdots \mathbf{o}_t, q_t = i|\lambda)$, and $\beta_t(i)$ is a backward variable defined as $\beta_t(i) = P(\mathbf{o}_{t+1} \mathbf{o}_{t+2} \cdots \mathbf{o}_T | q_t = i, \lambda)$. The name of the method originated from the mechanism of combining the forward and backward probability computations.

Amazon / Zentian Limited
Exhibit 1021
Page 69

the Viterbi search and the dynamic programming search, and thus this simplified method is usually referred to as the Viterbi algorithm.

3.5.5 Discriminative Design Methods

As described in previous sections, the MLE and Bayesian approaches are often less than satisfactory for achieving high classification accuracy. The HMM classifier designed with the EM method also suffers from this problem. Thus, to increase classification accuracy, various DF-based design methods have been investigated.

The early methods, mainly studied in the 1980s, include maximization of mutual information [25] and the error-correction method [26]. Maximization of mutual information is discussed in more detail in Chapter 9, "Minimum Classification Error Networks." The design concept of the error-correction method is basically the same as the error-correction learning described in Section 3.3.4.

In the 1990s, the main research interest in discriminative design for HMM classifiers moved to the application of ANN-related methods. Although HMM can be considered a type of ANN [27–28], it is usually treated as a system structure different from neural networks. Therefore, ANN-based discriminative training methods, which are based on the minimization of squared error loss and sometimes on the maximization of mutual information, were applied to HMM-structure classifiers. Various examples of this type of ANN/HMM hybrid are elaborated in the literature [29–33].

Moreover, another type of ANN-related design method, i.e., learning vector quantization (e.g., [34]), was extensively applied to the design of HMM classifiers (e.g., [35, 36]).

The latest trend in discriminative design for HMM classifiers is based on the recent generalized probabilistic descent method [27, 37, 38]. Discriminative design, which has a close link to ANN-based recognition, will be discussed in more detail in several later chapters.

3.6 Language Modeling

3.6.1 Role of Language Modeling

There are several different levels of language information, e.g., syntax, semantics, and pragmatics. In most practical speech recognition designs,

Amazon / Zentian Limited
Exhibit 1021
Page 70

language modeling is treated as a process of controlling the syntactic information of language. In particular, it is used to estimate the a priori probabilities for the occurrence of an event of word class.

To understand the role of language modeling, let us consider an example task of classifying ten-figure telephone numbers, such as {0-1-2-9-8-7-6-5-4-3}. The vocabulary of the task consists of the following ten words: zero, one, two, three, four, five, six, seven, eight, nine. Our classifier is assumed to use an HMM-based acoustic word model for every digit word. A string input of ten figures is thus acoustically modeled by concatenating the HMM acoustic word models, and its corresponding class-conditional probability is computed through the accumulation of the individual HMM-based word probabilities.[11] Here, to achieve correct final decisions, we need accurate estimates of the a priori probabilities of all of the possible string classes.

In the framework of the above telephone number recognition task, let us assume three different example conditions in terms of the availability of knowledge about possible strings. The first condition is that no information is available. We have no knowledge about the telephone number system and the users' call histories in the task. The only reasonable attempt at modeling the a priori probabilities in this case would be to set an identical small value to each a priori probability. Obviously, however, this simple solution is inadequate. Such estimates of the a priori probabilities would not only be useless but in fact distort classification accuracy: some value of the a priori probability would be assigned to even a digit string class that corresponds to a nonexisting telephone number.

The second condition is that the information about the telephone number system is accessible but we have no information on the actual phone call histories. Knowledge is limited, but we can remove in this case the possibility of considering nonexisting phone number strings from the decision procedure of our classifier by setting the *a priori* probabilities of the corresponding nonexisting string classes to zero. The estimation quality of the a priori probabilities is still low, but the productive effect of removing the nonexisting classes from the list of possible class candidates would not be significant.

In the final case, we assume that in addition to the knowledge about the telephone number system, the call record corpus of telephone

11. HMMs are used in the concatenated form when classifying a long speech input such as concatenated words and sentences.

Amazon / Zentian Limited
Exhibit 1021
Page 71

users is available. Clearly, this additional knowledge helps improve the estimation quality for the a priori probabilities, probably resulting in the accurate classification of the inputs. The above three example cases clearly demonstrate the role and effectiveness of incorporating language information into the classification decision.

3.6.2 *N*-Gram Language Modeling

In the recent literature, many types of language models have been investigated. Among them, probabilistic models have attracted much research attention. The probabilistic language models include the *N*-gram [39], the probabilistic LR parser [40], and the stochastic context-free grammars (e.g., [41]). In this subsection, we shall elaborate on *N*-gram language modeling, which is becoming a standard choice for modeling.

Let us consider a task of recognizing fixed-length connected word strings, where the vocabulary size is M and the string length is $\mathscr{L}_s$. No strict restriction of word string, such as the above phone-number restriction, is assumed. Thus the number of possible word string classes becomes $M^{\mathscr{L}_s}$. We denote here the j-th string class among the $M^{\mathscr{L}_s}$ classes as C_j. Therefore, the a priori probability $P(C_j)$ is expressed as

$$\begin{aligned} P(C_j) &= P(w_1, w_2, \cdots, w_{\mathscr{L}_s}) \\ &= P(w_{\mathscr{L}_s} | w_1, \cdots, w_{\mathscr{L}_s - 1}) \\ &\quad \times P(w_{\mathscr{L}_s - 1} | w_1, \cdots, w_{\mathscr{L}_s - 2}) \times \cdots \\ &\quad \cdots \times P(w_2 | w_1) P(w_1) \\ &= \prod_{i=1}^{\mathscr{L}_s} P(w_i | w_1, w_2, \cdots, w_{i-1}) \end{aligned} \tag{3.29}$$

where w_i is one word class among the M vocabulary word classes, and $P(w_i|w_1, w_2, \cdots, w_{i-1})$ is the probability that w_i will be spoken given that word subsequence $w_1, w_2, \cdots, w_{i-1}$ was spoken previously. In principle, $P(w_i|w_1, w_2, \cdots, w_{i-1})$ is possible to estimate. However, we should note here that the accurate estimation of $P(w_i|w_1, w_2, \cdots, w_{i-1})$ requires us to observe (count) all of the M^i possible word strings. Under realistic task conditions, this computation is actually difficult to complete: most of the strings rarely occur and this insufficiency of observations makes the probability estimation inaccurate. Our daily-life

Amazon / Zentian Limited
Exhibit 1021
Page 72

experiences certainly show that we use only a small part of our vocabulary frequently and that we quite often and naturally face the difficulty of string class probability computation.

Even if some long word string rarely occurs, it often includes substrings that can be frequently observed. Therefore, it seems that approximating $P(w_i|w_1, w_2, \cdots, w_{i-1})$ by using a shorter, N-word-length history that is comparatively easy to observe is more feasible than pursuing the hopeless collection of rarer strings. This is the modeling concept of the N-gram language model. Naturally, the estimation of shorter versions of the model, where N is small, is easier than that for longer versions. Popular implementations of this model are the unigram ($N = 1$), the bigram ($N = 2$), and the trigram ($N = 3$).

N-gram modeling is a typical probabilistic kind of modeling based on the discrete multinomial distribution. Thus, we need as many observations as possible in order to increase the accuracy of the probability estimation. In reality, however, it is not easy to collect enough design strings to achieve the desired accuracy. Therefore, further improvement of the N-gram model remains one of the major research challenges in language modeling for speech recognition (e.g., [42]).

3.7 Concluding Remarks

3.7.1 Overview

We have explained the fundamental issues related to speech recognition by using simple example task settings of isolated-word recognition and static-pattern classification.[12] Unfortunately, however, these settings are primitive and limited. In this subsection, we shall give additional explanations of two important issues: (1) the selection of class model units and (2) open-vocabulary recognition.

3.7.2 Selection of Model Units

For most languages, the vocabulary size exceeds several tens of thousands, and even in our daily life we use more than several thousand words. The number of possible sentences that can be made

12. In Section 3.6, we introduced the task of recognizing word strings, but used it only as a task framework for describing language modeling.

Amazon / Zentian Limited
Exhibit 1021
Page 73

by concatenating these words easily amounts to hundreds of thousands or millions. In the case of recognizing continuously spoken words or, in other words, sentences, it is obviously impractical to handle sentences such as {How long does it take to go from Kyoto to Tokyo?} and {How much does it cost to go from Kyoto to Tokyo?} as distinct classes and make class (sentence) models in a sentence-by-sentence way because of the huge number of possible sentences. Therefore, a short linguistic segment such as the phoneme is generally used as a class to recognize and as a model unit. Consequently, the class model of a long segment, such as a sentence, is formed by concatenating the short unit models.

In using short model units, recognition of words and sentences is executed by repeating two operations: one that segments an input utterance (word or sentence) to short segments, each corresponding to a model unit, and another that classifies the segment as the model's class. Generally, when the unit models are designed appropriately for the accurate classification of the corresponding segments, the classification of the entire input is accurate. Thus, in most cases, a basic recognizer design target is to increase the accuracy of the unit models. However, a simple concatenation of the short unit models that are separately designed does not necessarily result in the desired accuracy of word and sentence recognition; this shortfall is due to the coarticulation phenomenon, observed in the acoustic feature vector sequence, that is caused by the smooth behavior of speech production organs. The selection of model units is currently one of the most urgent research issues for speech recognizer design. Also, several design methods have been investigated with the aim of adjusting the short unit models to accurate word and sentence models (e.g., [43, 44]).

3.7.3 Open-Vocabulary Recognition

The recognition of conversational utterances is more exciting and promising in its potential applications than that of read-speech samples. However, it is hopelessly difficult to completely specify all of the possible acoustic and linguistic phenomena of conversational large-vocabulary speech signals. As one approach to alleviating this difficulty, the research on open-vocabulary recognition is becoming increasingly active.

The open-vocabulary condition means that a recognizer requires no restriction on vocabulary and thus its users can basically speak freely by using any word. Because it is almost impossible for the recognizer to model all of the possible word-string inputs, the recognizer attempts to

Amazon / Zentian Limited
Exhibit 1021
Page 74

spot and extract only the keywords that are useful for understanding the input word strings.[13] The system behavior would be similar to a person listening to a foreign language: because the foreign language listener's vocabulary size and listening ability for short segments such as phonemes are limited, he or she only tries to spot keywords and understand what the speaker intends.

The quality of open-vocabulary recognition mainly depends on the selection of keywords and the accuracy of keyword spotting. In particular, two types of techniques have been investigated for keyword spotting: (1) spotting based on threshold comparison and (2) spotting based on garbage modeling. Study on these techniques remains an important research issue.

3.7.4 Bibliographical Remarks

The main purpose of this chapter has been to provide fundamental information on speech recognition. However, the scope of this chapter is obviously insufficient for describing the details of the various recognition techniques.

Textbooks such as [21, 22, 45] are useful for readers to study the fundamentals of speech recognition in more detail. One recent textbook [31] invites readers to the new world of ANN/HMM hybrid speech recognition. Review papers [32, 46] provide a convenient means of grasping the present research environment. One handbook-style textbook [47] clearly explains the advanced topics in modern speech recognition.

To study the fundamentals of statistical pattern recognition, textbooks such as [4, 5, 48, 49] are recommended. Other textbooks [11, 50] provide ANN perspectives on pattern recognition.

References

[1] Bregman, A. S., *Auditory Scene Analysis*, Cambridge, Mass.: MIT Press, 1990.

[2] Cooke, M., A. Morris, and P. Green; "Missing Data Techniques for Robust Speech Recognition," *Proc. ICASSP97*, Vol. 2, 1997, pp. 863–866.

[3] Akahane-Yamada, R., Y. Tohkura, A. R. Bradlow, and D. B. Pisoni, "Does Training in Speech Perception Modify Speech Production?" *Proc. ICSLP96*, 1996, pp. 606–609.

13. Understanding here implies grasping the main points of recognizer input.

Amazon / Zentian Limited
Exhibit 1021
Page 75

[4] Duda, R., and P. Hart, *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.

[5] Fukunaga, K., *Introduction to Statistical Pattern Recognition*, New York: Academic Press, 1972.

[6] Dempster, A. P., N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Roy. Stat. Soc.*, Vol. 39, No. 1, 1977, pp. 1–38.

[7] Fu, K., *Sequential Methods in Pattern Recognition and Machine Learning*, Academic Press, 1968.

[8] Geman, S., and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. on PAMI*, Vol. PAMI-6, No. 6, 1984, pp. 721–741.

[9] Amari, S., "A Theory of Adaptive Pattern Classifiers," *IEEE Trans. Electronic Computers*, Vol. EC-16, No. 3, 1967, pp. 299–307.

[10] Amari, S., "Information Theory II—Geometrical Theory of Information," Tokyo: Kyoritsu, 1968 (in Japanese).

[11] Nilsson, N., *The Mathematical Foundations of Learning Machines*, San Mateo, CA: Morgan Kaufmann Publishers, 1990.

[12] Bekesy, V. G., *Experiments in Hearing*, New York: McGraw-Hill, 1960.

[13] Zwicker, E., "Subdivision of the Auditory Frequency Range into Critical Bands," *J. Acoust. Soc. Am.*, Vol. 33, 1961, p. 248.

[14] Lyon, R. F., and C. Mead, "An Analog Electri Cochlea," *IEEE Trans. ASSP*, Vol. 36, 1988, pp. 1119–1134.

[15] Zwicker, E., and W. Peisl, "Cochlea Preprocessing in Analog Model, in Digital Model and in Human Inner Ear," *Hearing Research*, Vol. 44, 1990, pp. 209–216.

[16] Hirahara, T., "A Nonlinear Cochlea Filter with Adaptive Q Circuits," *J. Acost. Soc. Jpn.*, Vol. 47, 1991, pp. 327–335 (in Japanese).

[17] Atal, B. S., and S. L. Hanauer, "Speech Analysis and Synthesis by Linear Prediction of the Speech Wave," *J. Acoust. Soc. Am.*, Vol. 50, No. 2, 1971, pp. 637–655.

[18] Itakura, F., and S. Saito, "A Statistical Method for Estimation of Speech Spectral Density and Formant Frequencies," *Electronics and Communications in Japan*, Vol. 53-A, 1970, pp. 36–43.

[19] McDermott, E., and S. Katagiri, "LVQ-Based Shift-Tolerant Phoneme Recognition," *IEEE Trans. SP*, Vol. 39, No. 6, 1991, pp. 1398–1411.

[20] Waibel, A., et al., "Phoneme Recognition Using Time-Delay Neural Networks," *IEEE Trans. ASSP*, Vol. 37, No. 3, 1989, pp. 328–339.

[21] Huang, X., Y. Ariki, and M. Jack, "Hidden Markov Models for Speech Recognition," Edinburgh: Edinburgh University Press, 1990.

[22] Rabiner, L., and B.-H. Juang, *Fundamentals of Speech Recognition*, Englewood Cliffs, NJ: Prentice Hall, 1993.

[23] Baum, L. E., and T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains," *Ann. Math. Stat.*, Vol. 37, 1966, pp. 1554–1563.

Amazon / Zentian Limited
Exhibit 1021
Page 76

[24] Baum, L. E., and G. R. Sell, "Growth Functions for Transformations on Manifolds," *Pac. J. Math.*, Vol. 27, No. 2, 1968, pp. 211–227.

[25] Bahl, L., P. Brown, P. de Souza, and R. Mercer, "Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition," *IEEE Proc. ICASSP86*, Vol. 1, 1986, pp. 49–52.

[26] Bahl, L., P. Brown, P. de Souza, and R. Mercer, "A New Algorithm for the Estimation of Hidden Markov Model Parameters," *IEEE Proc. ICASSP88*, Vol. 1, 1988, pp. 493–496.

[27] Katagiri, S., C.-H. Lee, and B.-H. Juang, "New Discriminative Training Algorithms Based on the Generalized Probabilistic Descent Method," *IEEE Neural Networks for Signal Processing*, 1991, pp. 299–308.

[28] Kung, S.-Y., "Digital Neural Networks," Prentice-Hall, 1993.

[29] Bengio, Y., R. De Mori, G. Flammia, and R. Kompe, "Global Optimization of a Neural Network-Hidden Markov Model Hybrid," *IEEE Trans. NN*, Vol. 3, No. 2, pp. 252–259.

[30] Bourlard, H., and C. Wellekens, "Links Between Markov Models and Multilayer Perceptrons," *IEEE Trans. PAMI*, Vol. 12, No. 12, 1990, pp. 1167–1178.

[31] Bourlard, H. and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*, Norwell, MA: Kluwer Academic Publishers, 1994.

[32] Morgan, N. and H. A. Bourlard, "Neural Networks for Statistical Recognition of Continuous Speech," *Proc. of IEEE*, Vol. 83, No. 5, pp. 742–770.

[33] Niles. L., and H. Silverman, "Combining Hidden Markov Model and Neural Network Classifier," *IEEE Proc. ICASSP90*, Vol. 1, 1990, pp. 417–420.

[34] Kohonen, T., "The Self-Organizing Map," *Proc. IEEE*, Vol. 78, No. 9, 1990, pp. 1464–1480.

[35] Iwamida, H., S. Katagiri, E. McDermott, and Y. Tohkura, "A Hybrid Speech Recognition System Using HMMs with an LVQ-Trained Codebook," *J. Acoust. Soc. Jpn.* (E), Vol. 11, No. 5, 1990, pp. 277–286.

[36] Katagiri, S. and C.-H. Lee, "A New Hybrid Algorithm for Speech Recognition Based on HMM Segmentation and Learning Vector Quantization," *IEEE Trans. SAP*, Vol. 1, No. 4, 1993, pp. 421–430.

[37] Juang, B.-H., and S. Katagiri, "Discriminative Learning for Minimum Error Classification," *IEEE Trans. SP.*, Vol. 40, No. 12, 1992, pp. 3043–3054.

[38] Katagiri, S., B.-H. Juang, and C.-H. Lee, "Pattern Recognition Using a Family of Design Algorithms Based Upon the Generalized Probabilistic Descent Method," *Proc. IEEE*, Vol. 86, No. 11, 1998, pp. 2345–2373.

[39] Jelinek, F., "Self-Organized Language Modeling for Speech Recognition," IBM, T. J. Watson Research Center Report, 1985.

[40] Wright, J., "LR Parsing of Probabilistic Grammars with Input Uncertainty for Speech Recognition," *Computer Speech and Language*, Vol. 4, 1990, pp. 297–323.

[41] Lari, K., and S. Young, "The Estimation of Stochastic Context-Free Grammars Using the Inside-Outside Algorithm," *Computer Speech and Language*, Vol. 4, 1990, pp. 35–56.

Amazon / Zentian Limited
Exhibit 1021
Page 77

[42] Nakamura, M., K. Maruyama, T. Kawabata, and K. Shikano, "Neural Network Approach to Word Category Prediction for English Texts," *Proc. COLING90*, pp. 213–218.

[43] Rabiner, L., J. Wilpon, and B.-H. Juang, "A Segmental k-Means Training Procedure for Connected Word Recognition," *AT&T Tech. Journal*, Vol. 65, No. 3, 1986, pp. 21–31.

[44] Chou, W., B.-H. Juang, and C.-H. Lee, "Segmental GPD Training of HMM Based Speech Recognition," *IEEE Proc. ICASSP92*, Vol. 1, 1992, pp. 473–476.

[45] Morgan, D. P., and C. L. Scofield, *Neural Networks and Speech Processing*, Boston: Kluwer Academic, 1991.

[46] Young, S., "A Review of Large-Vocabulary Continuous-Speech Recognition," *IEEE SP Magazine*, Vol. 13, No. 5, 1996, pp. 45–57.

[47] Lee, C.-H., F. K. Soong, and K. K. Paliwal, *Automatic Speech and Speaker Recognition*, Boston: Kluwer Academic, 1996.

[48] Iijima, T., *Pattern Recognition Theory*, Tokyo: Morikita, 1989 (in Japanese).

[49] Oja, E., *Subspace Methods of Pattern Recognition*, Letchworth: Research Studies Press, 1983.

[50] Pao, Y.-H., *Adaptive Pattern Recognition and Neural Networks*, Reading, MA: Addison-Wesley, 1989.

Amazon / Zentian Limited
Exhibit 1021
Page 78

SIGNAL PROCESSING



Artech House Publishers BOSTON • LONDON
www.artechhouse.com

Amazon / Zentian Limited
Exhibit 1021
Page 79