ACOUSTIC PARAMETER PROCESSING FOR DETECTION OF LARYNGEAL PATHOLOGIES USING A BOLTZMANN MACHINE

Jocelyn Frank Trehern

Thesis submitted for the degree of Ph.D.

University of Edinburgh

1990

ACKNOWLEDGEMENTS

I would like to acknowledge the support I have received from many individuals during the preparation of this thesis.

Professor Mervyn Jack, Director of the Centre for Speech Technology Research, University of Edinburgh who has inspired and encouraged me throughout the course of this investigation into the neuromorphic processing of speech data.

Professor Laver, Chairman, Centre for Speech Technology Research, Dr. Steven Hiller and Eddie Rooney for there knowledge and experience in the speech pathology field.

Dr. Mark Terry whose kind advice concerning the idiosyncrasies of the vector accelerator computer system used, helped me to refine the simulation software.

The Centre for Speech Technology Research, University of Edinburgh for making available computing resources and technical support.

The support of the Science and Engineering Research Council Studentship 8431244X is also gratefully acknowledged.

I am also indebted to my many *ballooning* friends who have continually provided me with more than just hot air for support.

Special thanks are due to my good friend George Fletcher, for all his support and encouragement, especially during the latter stages of preparation. Finally, but by no means least, considerable thanks are due to Alison Dunn for her tremendous moral support.

DECLARATION

This thesis was composed by myself and the work herein is my own.

J. F. TREHERN

December 1990

ABSTRACT

This thesis examines the application of a particular neuromorphic computational model, the Boltzmann Machine, to the evaluation of laryngeal behaviour using parameters derived from the acoustic analysis of irregularities in the periodic structures of speech signals. Over the last twenty five years, researchers in various fields such as speech science, laryngology, speech pathology and phonetics have demonstrated a growing interest in the acoustic characterisation of healthy and pathological voices. This research activity has been in response to the need for non-invasive and quantitative techniques for the assessment of laryngeal function.

Over the past five years neuromorphic computation has undergone a dramatic transformation with the development of powerful learning algorithms and the promise of highly parallel implementations taking advantage of developments in high density integrated circuit technology. These neuromorphic systems are machines that behave in brain-like ways and compute by absorbing experience. The Boltzmann Machine learning algorithm provides a formally guaranteed procedure for performing gradient descent in a global error measure. This thesis presents, for the first time, results which demonstrate the potential of the Boltzmann Machine approach to the detection of laryngeal pathologies.

A simulation environment for Boltzmann Machines was successfully developed which provided acceptable speeds of operation for the sizes of network investigated, and the quantity of training data used, provided approximations to the *theoretical* Boltzmann Machine were made. Chapter 4 presents details of this implementation.

Experiments using various topologies of Boltzmann Machine made use of ten intonation and perturbation parameters, derived from the analysis of waveform perturbations of fundamental frequency and amplitude evidenced in samples of connected speech from groups of healthy and pathological male speakers. A series of experiments are presented in Chapter 6 which evaluate the performance of various Boltzmann Machine topologies and data representation formats to the

differentiation between groups of healthy speakers and speakers with known pathological conditions of the larynx. From these experiments it was concluded that the intonation and perturbation parameters could be processed using a Boltzmann Machine to provide useful differentiation between groups of healthy speakers and speakers with known pathological conditions.

Chapter 7 presents a series of experiments using various topologies of Boltzmann Machine to differentiate between broad classes of pathologies using ten intonation and perturbation parameters. The experiments showed that it was possible for various pathology groups to be differentiated in a training group.

The successful development of a neuromorphic system for determination of laryngeal function associated with healthy and pathological phonation has a number of potential applications, including screening, differential diagnosis and tracking changes in the condition of laryngeal pathology.

PRINCIPAL SYMBOLS AND ABBREVIATIONS

A0	Amplitude of fundamental frequency waveform peaks
CAM	Content Addressable Memory
CPU	Central Processing Unit
CSA	Classical Simulated Annealing
DAP	Distributed Array Processor
DMA	Direct Memory Access
E	Energy
F0	Fundamental frequency
F0-DEV	Standard deviation of F0 trendline values
FO-AV	Mean value of smoothed F0 trendline
FSA	Fast Simulated Annealing
G	An information theoretic distance measure
Hz	Hertz
J-AVEX	Average magnitude of excursions of the raw F0 contour from the local trendline
J-DEVEX	Standard deviation of (signed) excursions of the raw F0 contour from trendline
J-DPF	Directional Perturbation Factor for F0
J-RATEX	Rate of F0 excursions
LMS	Least Mean Square
LTM	Long Term Memory
MI	Memory Interconnect (bus)
N	Total number of units
$N(0,\sigma)$	Sample from a Gaussian distribution of mean 0 and standard deviation σ
net_i	Total input to unit *i*
NFS	Network File System
o_i	Output of unit *i*
P	Total number of nominated patterns

p_{ij}^{+}	Probability of finding units *i* and *j* on together at thermal equilibrium when the input and output units are clamped (input/output model)
p_{ij}^{-}	Probability of finding units *i* and *j* on together at thermal equilibrium when the input units are clamped (Input/output model)
RTL	Run-Time Library
RTU	Real-Time Unix
S-AVEX	Average magnitude of excursions of the raw A0 contour from the local trendline
S-DEVEX	Standard deviation of (signed) excursions of the raw A0 contour from trendline
S-DPF	Directional Perturbation Factor for A0
S-RATEX	Rate of A0 excursions
SD	Standard Deviation
s_i	State of activation of unit *i*
SIMD	Single Instruction Multiple Data
STM	Short Term Memory
T	Temperature
TLU	Threshold Logic Unit
u_i	i^{th} unit
VLSI	Very Large Scale Integration
w_{ij}	connection strength and sense from unit u_j to unit u_i
XOR	Exclusive-OR

Table of Contents

ACKNOWLEDGEMENTS i

DECLARATION ii

ABSTRACT iii

PRINCIPAL SYMBOLS AND ABBREVIATIONS v

1 INTRODUCTION 1
1.1 INTRODUCTION 1
1.2 OVERVIEW OF REMAINING CHAPTERS 5

2 NEUROMORPHIC SYSTEMS 8
2.1 INTRODUCTION 8
2.2 HISTORY OF NEUROPHYSIOLOGY 8
2.2.1 Early Work 8
2.2.2 Microscopic Anatomy 10
2.2.3 Electrical Properties 12
2.2.4 Functional Organisation 13
2.2.5 Ionic Workings of Neurons 15
2.3 THE NEURON 16
2.4 GENERAL NOTATION FOR NEUROMORPHIC MODELS 21
2.4.1 Processing units 24
2.4.2 State of Activation 24
2.4.3 Output Function 25
2.4.4 Pattern of Connectivity 25
2.4.5 Rule of Propagation 26
2.4.6 Activation Rule 26
2.4.7 Learning Rule 27
2.4.8 Environment 28
2.5 INTERPRETATION OF NEURAL VARIABLES 28
2.6 EVOLUTION OF NEUROMORPHIC MODELS 30
2.6.1 McCulloch and Pitts 30
2.6.2 The Single-layer Perceptron 31
2.6.3 Grossberg 35
2.6.4 Hopfield Model 36
2.6.5 Boltzmann Machine 37
2.6.6 Multi-layer Perceptron 37
2.7 SUMMARY 38

3 THERMODYNAMIC NEUROMORPHIC MODELS 41
3.1 INTRODUCTION 41
3.2 HOPFIELD MODEL 43
3.2.1 Introduction 43
3.2.2 Hopfield Model 45
3.2.3 Relationship to the Brain 51
3.2.4 Ising Spin Analogies 53
3.3 BOLTZMANN MACHINE 56
3.3.1 Introduction 56
3.3.2 Minimising Energy 57
3.3.3 Simulated Annealing 58
3.3.4 Boltzmann Machine Learning Algorithm 63
3.3.5 Relationship to the Brain 69
3.4 CAUCHY MACHINE 70
3.5 GAUSSIAN MACHINE 71

3.6 HARMONY MACHINE 74
3.7 DETERMINISTIC MODEL 79
3.7.1 Multi-layer Perceptron 79
3.7.2 Relationship to Hopfield and Boltzmann Machines 82
3.8 SUMMARY 83

4 SIMULATION ENVIRONMENT 84
4.1 INTRODUCTION 84
4.2 VECTOR ACCELERATOR OVERVIEW 86
4.3 BOLTZMANN MACHINE LEARNING ALGORITHM 91
4.3.1 Network Structure 91
4.3.2 Learning Overview 94
4.3.3 Unit State Update 94
4.3.4 Simulated Annealing 98
4.3.5 G Measure 98
4.3.6 Co-occurrence Statistics 99
4.3.7 Weight Adaptation 100
4.4 SOFTWARE DEVELOPMENT METHODOLOGY 100
4.4.1 Introduction 100
4.4.2 Requirements Analysis 101
4.4.3 Design Specification 108
4.4.4 Implementation and Coding 109
4.4.5 Unit Test, Integration and System Test 109
4.4.6 Porting Array Processor Software to the Vector Accelerator 110
4.5 SOFTWARE IMPLEMENTATION FOR LEARNING 111
4.5.1 Data Structures 111
4.5.2 Main Component Module 116
4.5.3 Random numbers 118
4.5.4 Initialisation 118
4.5.5 Energy Calculation 119
4.5.6 Updating Unit States 120
4.5.7 Statistics Gathering 122
4.5.8 Updating the Weights 123
4.5.9 Error Measures 124
4.5.10 Learning Time Performance 124
4.6 SEARCH MODULE 128
4.7 GRAPHIC WEIGHTS DISPLAY MODULE 128
4.8 INTERPRETATION OF WEIGHT MAPS 129
4.9 XOR EXAMPLE 131
4.10 4-2-4 ENCODER EXAMPLE 136

5 LARYNGEAL PATHOLOGY AND WAVEFORM PERTURBATIONS 141
5.1 INTRODUCTION 141
5.2 SPEECH PRODUCTION 142
5.3 THE LARYNX 144
5.4 LARYNGEAL PATHOLOGY 152
5.4.1 Diagnosis 152
5.4.2 Screening 153
5.4.3 Structural Pathologies of the Larynx 154
5.5 PERTURBATION STUDIES 161
5.6 PERTURBATION MEASUREMENT SYSTEM 163
5.6.1 Pitch Detection 164
5.6.2 Perturbation Algorithm 168
5.7 LARYNGEAL PATHOLOGY DETECTION STUDIES USING ACOUSTIC PARAMETERS 172
5.8 SPEAKERS USED FOR PERTURBATION ANALYSIS 174

6 SEPARATION OF CONTROL AND PATHOLOGICAL GROUPS 177
6.1 INTRODUCTION 177
6.2 DATA PREPARATION 182
6.3 TRAINING AND TEST GROUPS 185
6.4 SIMULATIONS 188
6.4.1 No Hidden Units 188
6.4.2 Hidden Units 189
6.4.3 Intonation, Jitter and Shimmer Receptive Field Networks 190
6.4.4 Intonation and Perturbation Parameter Receptive Field Networks 190
6.4.5 Selected Parameter Networks 191
6.5 RESULTS 192
6.5.1 No Hidden Units 192
6.5.2 Hidden Units 215
6.5.3 Intonation, Shimmer and Jitter Receptive Fields 228
6.5.4 Intonation and Perturbation Parameter Receptive Fields 233
6.5.5 Selected Parameter Networks 238
6.6 COMPARISON WITH RESULTS FROM OTHER WORK 245
6.7 SUMMARY 246

7 PATHOLOGY IDENTIFICATION 251
7.1 INTRODUCTION 251
7.2 SIMULATIONS 252
7.2.1 No hidden units 252
7.2.2 Hidden units 252
7.3 RESULTS 253
7.3.1 No hidden units 253
7.3.2 Hidden units 263
7.4 COMPARISON WITH RESULTS FROM PREVIOUS WORK 266

8 CONCLUSION 268
8.1 Review of Results Presented in Thesis 268
8.2 Comparison to Other Work 274
8.3 Areas for Further Work 275

REFERENCES 278

APPENDICES 291
APPENDIX 1 Speech Processing with a Boltzmann Machine (Reprint of Trehern, Jack & Laver 1986).
APPENDIX 2 Acoustic Screening for Vocal Pathology with a Boltzmann Machine. (Reprint of Trehern, Jack, Laver & Hiller 1987).

Table of Figures

Figure 2.1 Neurons stained by the Golgi method 11
Figure 2.2 A typical phrenological map 14
Figure 2.3 The classical neuron 17
Figure 2.4 Membrane events 20
Figure 2.5 Basic components of a neuromorphic system 23
Figure 2.6 Single-layer Perceptron 33
Figure 3.1 Four unit fully interconnected Hopfield network 46
Figure 3.2 Firing rate vs membrane voltage for a neuron 53
Figure 3.3 Probability values as a function of net input 62
Figure 3.4 A graphical representation of a Harmony Machine 77
Figure 4.1 Block Diagram of Simulation System 85
Figure 4.2 Vector Accelerator and Host Block Diagram 88
Figure 4.3 Structure of the Input/Output Model 93
Figure 4.4 Superimposed curves for probability function 97
Figure 4.5 Structure for Nets with 3 Receptive Fields 106
Figure 4.6 Structure for Nets with Ten Receptive Fields 107
Figure 4.7 Vector accelerator memory allocation 112
Figure 4.8 Weight matrix format 115
Figure 4.9 Simulator learning time performance 127
Figure 4.10 A simple XOR network with one hidden unit 132
Figure 4.11 Smoothed percentage bit error for XOR learning 133
Figure 4.12 Weight map for the XOR network 135
Figure 4.13 4-2-4 Encoder 137
Figure 4.14 Smoothed percentage bit error for encoder 138
Figure 4.15 Weight map for 4-2-4 Encoder 140
Figure 5.1 The vocal apparatus 143
Figure 5.2 Cross-sectional view through the larynx 145
Figure 5.3 Schematic view of the vocal folds seen from above 146
Figure 5.4 Schematic rep. of ligamental part of vocal fold 148
Figure 5.5 Parallel processor pitch detector 165
Figure 6.1 Weight map for single-layer net (dot & 12 levels) 195
Figure 6.2 Weight map for single-layer net (slide/12 levels) 198
Figure 6.3 Weight map for single-layer net (bar/12 levels) 199
Figure 6.4 Results for single-layer net (dot & 12 levels) 202
Figure 6.5 Results for single-layer net (slide & 12 levels) 203
Figure 6.6 Results for single-layer net (bar & 12 levels) 204
Figure 6.7 Weight map for single-layer net (dot/12) 207
Figure 6.8 Weight map for single-layer net (slide/12) 208
Figure 6.9 Weight map for single-layer net (bar/12) 209
Figure 6.10 Weight map for single-layer net (dot/12) 212
Figure 6.11 Weight map for single-layer net (slide/12) 213
Figure 6.12 Weight map for single-layer net (bar/12) 214
Figure 6.13 Results for network with 2 hidden units (12/dot) 219
Figure 6.14 Results for network with 4 hidden units (bar/12) 220
Figure 6.15 Results for network with 8 hidden units (bar/12) 221
Figure 6.16 Weight map for net with 2 hidden units (bar/12) 223
Figure 6.17 Weight map for net with 4 hidden units (bar/12) 224
Figure 6.18a Weight map for net wih 8 hidden units (bar/12) 226
Figure 6.18b Weight map for net with 8 hidden units (bar/12) 227
Figure 6.19 Weight map for 3 receptive field net (dot/12) 229
Figure 6.20 Weight map for 3 receptive field net (slide/12) 230
Figure 6.21 Weight map for 3 receptive field net (bar/12) 231
Figure 6.22a Weight map for 10 receptive fields net 235
Figure 6.22b Weight map for 10 receptive fields net 237
Figure 6.23 Weight map for net trained with set 1 (slide/12) 240

Figure 6.24 Weight map for net trained with set 2 (slide/12) 241
Figure 6.25 Weigh map for net trained with set 3 (slide/12) 242
Figure 6.26 Weight map for net trained with set 4 (slide/12) 243
Figure 7.1 Weight map for single-layer net (dot/12) 255
Figure 7.2 Weight map for single-layer net (slide/12) 256
Figure 7.3 Weight map for single-layer net (bar/12) 257
Figure 7.4 Weight map for single-layer net (dot/12) 258
Figure 7.5 Weight map for single-layer net (slide/12) 259
Figure 7.6 Weight map for single-layer net (bar/12) 260
Figure 7.7 Weight map for net with 2 o/p units (dot/6) 264
Figure 7.8 Weight map for net with 4 o/p units (dot/6) 265

Table of Tables

Table 2.1 Summary of neuron features 21
Table 2.2 Principle components of neuromorphic systems 22
Table 2.3 Interpretation of neural variables 30
Table 4.1 Network Topologies 105
Table 4.2 Simulator Modules 108
Table 4.3 Locations 0-5 of Vector Accelerator Memory 113
Table 4.4 Vector Accelerator Timings for Sigmoid Function 121
Table 4.5 Vector Accelerator Timings for Gaussian Method 121
Table 4.6 Total Number of Weights for Different Net Sizes 125
Table 4.7 Relative Performance of Functions 126
Table 4.8 The XOR Problem 131
Table 4.9 The XOR problem showing hidden unit states 136
Table 4.10 4-2-4 Hidden unit states 139
Table 5.1 A summary of chars. of vocal fold path. 157
Table 5.2 Group separation results - (linear discriminant) 173
Table 5.3 Laryngeal disorders diagnosed in pathologic. group 175
Table 6.1 Intonation and perturbation parameters 182
Table 6.2 Examples of the three binary formats 184
Table 6.3 Laryngeal disorders diagnosed in small train. set 186
Table 6.4 Laryngeal disorders diagnosed in large train. set 187
Table 6.5 Parameter subsets 191
Table 6.6 Generalisation results (hidden units, dot/6) 215
Table 6.7 Generalisation results (hidden units, slide/6) 215
Table 6.8 Generalisation results (hidden units, bar/6) 216
Table 6.9 Generalisation results (hidden units, dot/12) 216
Table 6.10 Generalisation results (hidden units, slide/12) 217
Table 6.11 Generalisation results (hidden units, bar/12) 217
Table 6.12 Generalisation results for 3 receptive field net 228
Table 6.13 Results for 10 receptive field nets (12 levels) 233
Table 6.14 Results for 10 receptive field nets (12 levels) 234
Table 6.15 Results for single-layer net (subset 1/12 levels) 238
Table 6.16 Results for single-layer net (subset 2/12 levels) 238
Table 6.17 Results for single-layer net (subset 3/12 levels) 239
Table 6.18 Results for single-layer net (subset 4/12 levels) 239
Table 6.19 Results for net with 2 hid. (subset 1/6 levels) 244
Table 6.20 Results for net with 2 hid (subset 2/6 levels) 244
Table 6.21 Results for net with 2 hid. (subset 3/6 levels) 245
Table 6.22 Results for net with 2 hid (subset 4/6 levels) 245

1 INTRODUCTION

1 INTRODUCTION

1.1 INTRODUCTION

The aims of this thesis are to assess a neuromorphic computational approach, based on the Boltzmann Machine model, to the processing of acoustic parameters for screening voices for the presence of laryngeal pathologies, and for the differentiation of such pathologies.

Deviant laryngeal behaviour associated with pathology of the larynx is assessed and quantified by laryngologists. Their chief concerns are the medical diagnosis and treatment of laryngeal pathologies. Two of the basic techniques used by the laryngologist to diagnose the various disorders of the laryngeal mechanism are visual examination of the larynx and auditory evaluation of the voice.

Visual laryngeal examination is usually completed by indirect laryngoscopy in which a mirror is placed in the patient's throat in order to observe the vocal folds and surrounding tissues. This visual assessment technique however, only provides a restricted supralaryngeal view of the larynx under static conditions. Other instruments such as the direct laryngoscope can be used for improved visual examination of the larynx, but all these techniques are invasive and may not be readily accepted by certain patients. Both the indirect and direct methods may be combined with stroboscopic illumination techniques to provide a more dynamic view of vocal fold activity. Auditory appraisal of the phonatory quality of a patient's voice suffers from non-pertinent factors, with the differences between individual's perceptions being the greatest problem.

Although there are alternative methods (such as stroboscopy, glottography, electrolaryngography, electromyography and laryngoscopy) widely available for quantitative laryngeal research and diagnosis, an acoustic technique has the distinct benefit of being non-invasive, and also produces objective quantitative measures of dynamic laryngeal behaviour.

The non-invasive nature also makes it suitable for routine clinical assessment of laryngeal function and, in particular, for screening populations for the presence of laryngeal pathologies. Voice samples may be collected using standard digital tape recording procedures, which are simple and highly portable. These may be operated by non-medical personnel in any relatively quiet situation in clinics, factories and schools. This technique also ensures that speakers are only subjected to minimal distress.

An automatic acoustic system which can detect laryngeal pathology therefore has several potential applications. These include screening of populations; allowing assessment of priorities among a pre-selected population of patients already complaining of voice problems; diagnostic support where a particular laryngeal pathology is already suspected; longitudinal monitoring to assess change in phonatory efficiency in patients undergoing surgery, speech therapy, radiotherapy or chemotherapy; or to track deterioration in progressive disease.

However the validity of acoustic assessment procedures is dependent on the complex relationship between the vibrating source function and the resultant speech signal output by the production system. Davis (1979) summarises the nature of this complex relationship in the following way:-

1) In general, asymmetrical changes in the mass and elastic properties of the vocal folds are created by the presence of laryngeal pathology.

2) These asymmetrical changes result in modulation of the subglottal airstream by unbalanced vocal fold movement.

3) Irregular air pulses are emitted by the larynx into the supraglottal structures which are then radiated at the lips and nose.

4) The resultant acoustic signal is therefore affected by a disturbance of the vocal folds -- and the acoustic speech signal can be used to quantify the disturbance.

Studies by Hiller (1985) and Beck (1988) using long-term intonational and perturbation parameters to discriminate speakers as either belonging to a control group or to a pathological group have been based on bivariate plot, linear discriminate analysis or maximum likelihood estimation techniques. From these studies it has emerged that many of the parameters studied represent redundant information and that no single parameter is sufficient for satisfactory discrimination between the two groups. If these parameters are indeed capable of providing an adequately robust and reliable discrimination system, then the problem of determining which features of the individual parameters or groups of parameters are pertinent to this process ensue. Neuromorphic models offer an alternative approach to the above mentioned techniques.

Many problems cannot be assessed in terms of isolated facts or even in terms of a body of isolated facts. Rather, there is a need to describe situations in terms of patterns of interrelated facts. Sometimes, the interrelation is implicit, in the sense that we know that all those facts pertain to the same object or situation. In other cases a pattern may be meaningful only because of explicit relationships among the various features of the pattern. Our perceptive powers seem to be well adapted to such pattern-processing tasks. Neuromorphic computational systems are machines that are able to process pattern-information in neurally inspired ways.

"Neural computing is .... concerned with a class of machines that compute by absorbing experience, and in that sense is a class which includes the brain, but may include other forms with similar properties." (Aleksander, 1989).

The architecture of these systems differs radically from that of the Von Neumann machine. In neuromorphic systems the processing takes place in memory, and is distributed over many elements operating in parallel. These elements are arranged in patterns similar to those found in biological systems, and connected to each other by adaptive weights. Knowledge is acquired through training rather than programming and is retained using interconnection weights. In the operation

of the net, the knowledge takes the form of stable states or cycles of states. A key property of such nets is to recall these states or cycles in response to the presentation of input stimuli.

Aleksander (1989) neatly summarises the promise that neuromorphic computing systems hold as follows:-

1) They are computationally complete, meaning that, given an appropriate neural structure, and appropriate training, there are no computational tasks that are not available to neuromorphic systems.

2) The ability to perform functions beyond the capability of rule-based, conventional systems by functional use of experiential knowledge. This also includes the ability to generalise, without requiring any special generalisation formation mechanisms.

3) Rapid solutions to problems which in conventional computers would take a long time.

4) Insights into the computational characteristics of the brain.

This thesis is an attempt to develop a neuromorphic computational approach to the processing of acoustic perturbations found in the speech signals produced by pathological speakers as well as healthy speakers. It presents for the first time, results which demonstrate the potential of this approach to the detection and differentiation of laryngeal pathologies using intonation and perturbation parameters.

The neuromorphic systems described in this work attempt to implement approximations of the useful computational properties exhibited by neuronal systems. The Boltzmann Machine may be regarded as a special case of the Gaussian Machine (Akiyama et al., 1989). The software simulations undertaken in this study are in fact close approximations to the Boltzmann Machine model, as Gaussian rather than logistic noise has been used. One may therefore argue that the network models described in the following chapters should be referred to as Gaussian Machines. However, the Gaussian Machine should be regarded

as the general model, with models such as the McCulloch-Pitts, Hopfield and Boltzmann Machine being special cases. Where relevant, the network models described in this work are referred to as Boltzmann Machines.

The Boltzmann Machine structure requires symmetric links, an appropriate decision rule and the absence of connections from a unit to itself, but makes no other restrictions on the topology of the network employed. Units directly affected by the environment may be connected to each other, or not. Hidden units can be connected to each other, or not. Units can be arranged in layers or not. The formal proof of the Boltzmann Machine learning algorithm is particularly elegant and is completely insensitive to such issues.

1.2 OVERVIEW OF REMAINING CHAPTERS

The opportunity is taken in Chapter 2 to present an introduction to neuromorphic systems, which as the name implies are largely based on our knowledge of the nervous system. A brief historical development of significant neurophysiological milestones is presented from the time of the ancient Greeks to the demonstration of the ionic workings of the nerve impulse by Hodgkin and Huxley in 1952. An outline of the basic mechanisms of operation of the neuron, which is essentially a communications device, and the basic element of the nervous system, is then discussed. A general framework for describing neuromorphic models is then presented along with a brief history of the development of these models.

The Boltzmann Machine may be described as a thermodynamic model because it allows a rigorous mathematical description using statistical mechanics techniques. Chapter 3 presents details of the development of the Boltzmann Machine, starting with the Hopfield model. The analogy with Ising spin systems and a biological interpretation of this model are also discussed. A modification to the Hopfield updating rule by Hinton, Sejnowski and Ackley (1984) resulted in the Boltzmann Machine. This is discussed along with the technique of simulated annealing which allows the Boltzmann Machine to find its global minimum. A brief review of the Harmony Machine, which is rather similar to the Boltzmann Machine

is also presented before detailing the general class of machines known as Gaussian Machines. A deterministic model, the Multi-layer Perceptron, is also capable of learning with hidden units using the back propagation algorithm and is also discussed for completeness.

The Boltzmann Machine simulation environment using a Masscomp MC5700[1] and vector accelerator platform is described in Chapter 4. This includes details of the implementation of the learning algorithm and an overview of the functional elements of the software developed. The graphical weight display routines and interpretation of the weight maps are also discussed before presenting two simple but explanatory learning examples. The first example is that of the classic logical exclusive-or (XOR) learning problem and the other the 4-2-4 encoder problem. Both these problems depict the ability of the Boltzmann Machine to use hidden units in determining the general underlying features of its environment.

Chapter 5 provides an introduction to laryngeal pathology and waveform perturbations. Details of the larynx and vocal folds are discussed together with the acoustic significance of various laryngeal pathologies. A brief review of perturbation analysis is presented together with details of the Hiller (1985) Perturbation Measurement System. This system provided the acoustic parameter data which was made available to the Research Group in which the various Boltzmann Machine studies were made. Finally, detail of the subjects and speech samples which were used in creating this data are provided.

The use of the Boltzmann Machine model to estimate whether speakers have healthy or pathological voices as evidenced by intonational and perturbation parameters is described in Chapter 6. The ability of networks with and without hidden units is discussed along with the difficulties associated in using a small data sample and achieving good generalisation. Architectural constraints using restricted receptive fields for the intonation, shimmer and jitter parameters are also investigated. In addition, the use of restricted receptive fields confined to

---

1 Masscomp and MC5700 are trademarks of Massachusetts Computer Corporation.

each of the ten intonation and perturbation parameters is presented. A comparison of Boltzmann Machine results to the results obtained in other studies is also made.

The differentiation of laryngeal pathologies using acoustic intonation and perturbation parameters with different architectures of Boltzmann Machines is discussed in Chapter 7. Comparisons to findings produced in previous studies are also made. Finally, Chapter 8 presents the conclusions. Published papers that have appeared in relation to this thesis are to be found in the Appendices.

2 NEUROMORPHIC SYSTEMS

2 NEUROMORPHIC SYSTEMS

2.1 INTRODUCTION

The origins of neuromorphic models stem from insights gained into the workings of the brain. These findings reveal three main points where the perceived operation of the brain differs radically from that of the conventional Von Neumann architecture machine. These differences: parallel rather than serial processing; distributed rather than local representation; and stochastic rather than deterministic algorithms; have deep implications for the solving of perceptual problems.

Neuromorphic systems attempt to implement approximations of useful computational properties exhibited by the brain. In this chapter the opportunity is therefore taken to acquaint the reader with the historical development of neurophysiology and to outline the basic mechanisms of operation of the nervous system. Following this, a general framework for describing neuromorphic models is presented along with an interpretation of neural variables. Finally, a brief history of the development of various significant neuromorphic models is presented.

2.2 HISTORY OF NEUROPHYSIOLOGY

Neuromorphic computational systems are, as the name implies, largely based on our knowledge of the nervous system. Inquiries into improving our understanding of the nervous system have a history that stretches back to ancient times. The considerable extent of this research is such that only the more significant episodes in the history of the subject have been selected. For a more extensive background see Clarke & O'Malley, (1968).

2.2.1 Early Work

The ancient Eygptians, Mesopotamians and Hebrews regarded the heart as the source of life. Homer wrote that the heart harboured intelligence and emotion. Greek medicine however, based on the work of Democritus and later Hippocrates

and his colleagues believed that the intellect and emotions were to be found within the brain. It was Hippocrates and his colleagues, who while making clinical studies of head injuries, showed that they could cause motor impairments. Moreover, they made the further discovery that these impairments were on the right when the left hand side of the brain was involved. Aristotle however revived the cardiocentric ideas of Homer and the Hebrews because the exposed brain was not sensitive to mechanical stimulation, while the heart was, and furthermore there was nothing resembling the vertebrate brain in such animals as worms, insects and shellfish.

The ancient Greeks, Herophilus and Erasistratus (3rd century BC) were probably the first to dissect the human body. Celsus wrote

> *"the Kings removed from the prisons to give to them, and they examined them while they were still breathing".*

This rather unsavoury enterprise led to the distinction of the cerebellum from the brain and the spinal chord. They showed that the brain contained ventricles, that the cortex was folded into convolutions and that nerves were different from blood vessels, that they originated not in the heart as Aristotle thought, but in the brain or the spinal column. They also distinguished between motor and sensory nerves. Thus, by the time of Galen a Greek anatomist and physician (200 BC), a good deal of the naked eye anatomy of the nervous system had been discovered. Galen was able to demonstrate that the brain played a principle role in controlling bodily and mental activity. These experiments should have dealt a fatal blow to the cardiocentric theorems.

Anatomical data alone, however, was not enough to undermine Aristotle's thesis that the heart was the seat of sensations, passions and the intellect. Medieval scholars forgot the work of Herophilus and Erasistratus, and this erroneous opinion survived until the eighteenth century. The Renaissance brought about the rediscovery of the ancient Greek observations and further advances were made as the dissection of animals, and above all the human body began again.

2.2.2 Microscopic Anatomy

The surface of the brain was first observed through a magnifying glass in 1685 by Marcello Malpighi. However, it was not until about 1718 , when the Dutch scientist Anton van Leeuwenhoek made the first faithful study of the microscopic organisation of the nervous system, observing long nerve fibres, we now know as axons.

Little significant progress was made until 1824 when René Dutrochet described and drew the nerve cells of snails and slugs. His descriptions resulted in the nerve cell making its first appearance in scientific literature. Shortly after this, Gabriel Valentin noted that some of these cells in the cerebellum possessed *tails*. These were later recognised as multiple and widely ramified like the branches of a tree, and hence came to be known as dendrites.

In order to observe the basic structure of the nervous system, a staining method was required that would allow a single cell to be viewed in its entirety. Camillo Golgi (1843-1926) developed a new staining technique using silver, which selectively stained only a few cells in the tissue examined (1-10%). Of those selected, it impregnated the soma, dendrites and axons, and thus meant that for the first time researchers could observe the fundamental nervous unit, (Figure 2.1).

Golgi was an advocate of the *reticular* hypothesis in which the nerve cells form a continuous network or reticulum, and rather than working individually, the cells act together. The alternative view was known as the *neuron* hypothesis, where the nerve cells were independent but contiguous units. Ramón y Cajal (1852-1934) announced an end to the reticular theory by making use of the Golgi method of staining. He discovered that axons had endbulbs that came very close to the membranes of other cells, but did not actually fuse with them. Observation of nerve terminal contacts finally became possible by means of the electron microscope in the 1950's. The junction was named a synapse by C. S. Sherrington

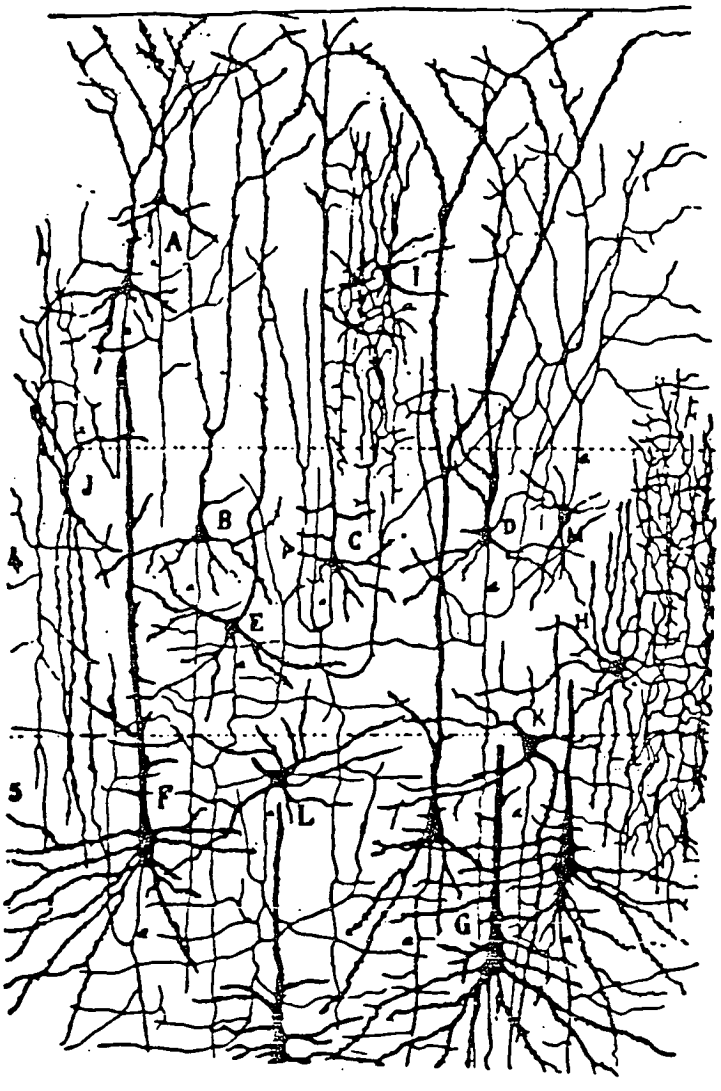

Figure 2.1 Neurons stained by the Golgi method

in 1897, who interestingly did not base his claim that the nervous system contained synapses on the direct observation of synaptic junctions, but a study of simple reflexes in dogs.

2.2.3 Electrical Properties

In 1780 Luigi Galvani observed that muscles contracted when subjected to discharges of static electricity from Leyden jars. Later in 1786 Galvani and his wife observed that a frog's leg contracted spontaneously when the preparation was suspended from an iron bar with a copper hook implanted in the spinal cord. This was in fact due to the rudimentary electrical battery that had been set up, rather than the production of current by the muscle as Galvani concluded. This principle was in fact later used by Volta, a fierce critic of Galvani, in developing a storage cell.

The development of the galvanometer allowed Carlo Matteucci in 1838 to record (for the first time) the production of an electric current by a muscle. In 1843 Du Bois-Reymond demonstrated that the contraction of a muscle was an electrical phenomenon and that a wave of electrical *negativity* or an *action potential* passes down the nerve. Thus electricity was established as the basis of normal nerve function.

Most of the early work on animal electricity was performed on dissected frogs. However in 1870 Gustav Fritsch and Edward Hitzig, who were both German physicians, demonstrated using dogs that certain areas of the cerebral cortex are sensitive to electrical stimulation, producing contractions of specific muscle groups. However they did not demonstrate that the cortex produces electricity. What they did do was to corroborate the views of Hughlings Jackson (1869/1958), an English neurologist who was one of the first to argue for distributed rather than local processing of complex mental functions. His hypothesis was based on his astute observations of many epileptic patients. He noted that patient's symptoms could be confined to a particular body area, and

that the surgical removal of a section of the cortex might yield no discernible effects. In fact on the subject of the relationship between brain lesions and speech he was able to write:-

> *"To locate the damage which destroys speech and to localise speech are two different things".*

From experiments with rabbits Richard Caton in 1875 was able to show that the cortex itself produces electricity. He wrote:-

> *"In every brain hitherto examined, the galvanometer has indicated the existence of electric currents";* moreover *"the electric currents of the grey matter appear to have a relation to its function", (Caton, 1875).*

Caton had thus discovered the technique of electroencephalography and the concept of evoked potentials, although it was not until 1929 that Hans Berger discovered a recording technique that did not require the opening of the skull. Instead electrodes were applied to the surface skin of the head and variations in electrical potential were recorded. This technique has enabled many subsequent discoveries to be made about the nature of the brain.

2.2.4 Functional Organisation

The French anatomist Pierre-Paul Broca was able to show that complex mental functions were *localised* in a particular area of the cortex, and that there are radical differences of function between the left and right cerebral hemispheres (Broca, 1863). His results were based on clinical fact, unlike the postulations of Franz Gall's phrenology. Gall's work was based on the assumption that the exterior features of the skull were a faithful representation of the surface of the cortex, and that these features corresponded to faculties that were particularly well developed in his subjects. Thus it was either by luck or deep intuition that Gall placed verbal memory and language in the frontal region where we accept them today (see Figure 2.2).

The locationist doctrine persisted with the belief that memory could be stored at specific locations, until questioned by Karl Lashley (1924,1950). Lashley

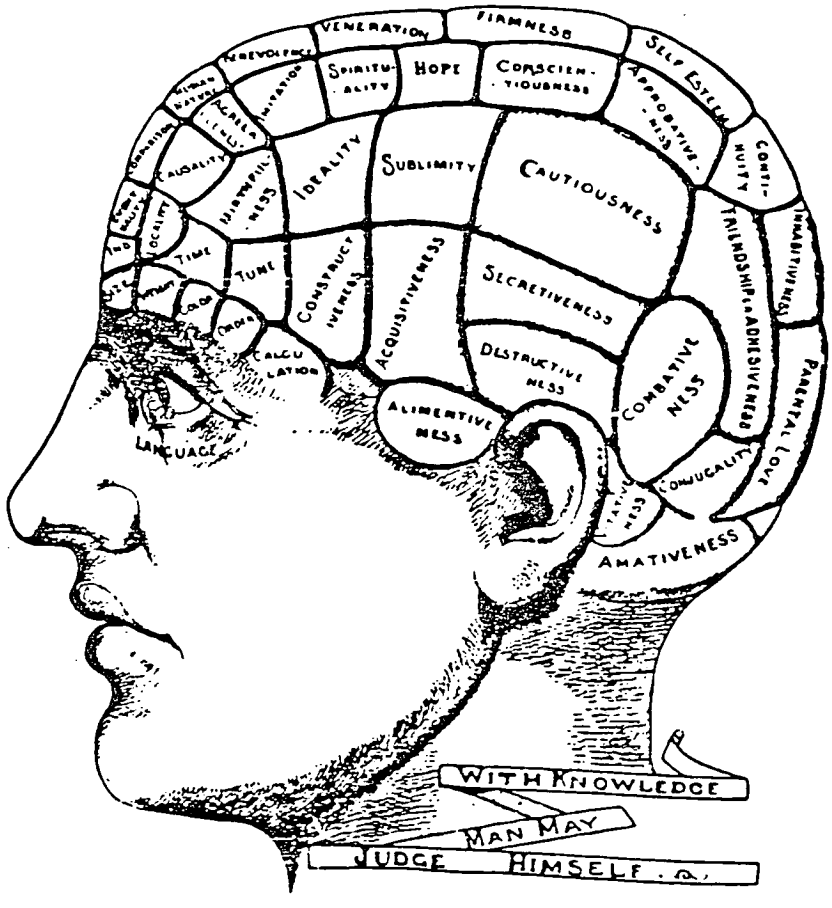

Figure 2.2 A typical phrenological map.

performed many experiments with rats, removing parts of their cortex to determine which part was responsible for a previously trained brightness-discrimination or maze habit. These experiments failed to locate an area responsible for the habit. Lashley concluded that a memory is probably established diffusely throughout all regions of the cortex (or maybe subcortex), and that it is multiply duplicated elsewhere involving huge numbers of neurons. Thus a single neuron would be involved in thousands of memories, with a memory being represented by a particular pattern of cortical excitation. Lashley's insistence that:-

> *"there are no special cells reserved for special memories" (Lashley, 1950).*

captures the idea of distributed representations perfectly.

2.2.5 Ionic Workings of Neurons

In the nineteenth century German physiologists elaborated and refined the notion that the nervous system transmitted electrical messages. With the advent of electronic measuring instruments in the early part of this century, the *all-or-nothing* principle of nervous transmission was established, i.e. if a nerve is stimulated either all of the impulse occurs, or if the stimulus is not strong enough to reach a threshold then nothing occurs. However, it was not until 1952, that Hodgkin and Huxley were able to give a complete demonstration of the ionic workings of the nerve impulse, based on a series of remarkable experiments recording electrical changes from inside the squid's giant axon (Hodgkin & Huxley, 1952). Their findings had a tremendous impact, especially as to their universal nature. This allowed the propagation of the nerve impulse, whether in a squid's giant axon, a rats sciatic nerve, or a neuron of the cerebral cortex to be explained by similar, if not identical basic mechanisms.

2.3 THE NEURON

The basic element of the nervous system is the neuron, which is essentially a communications device, receiving, integrating and sending signals. There are about 1000 types of neuron, but most have certain general signal processing characteristics in common which are represented in the classical neuron model. A classical neuron as depicted in Figure 2.3, consists of a cell body, or soma, and projections extending out from the soma.

The soma contains the genetic and metabolic machinery that is necessary to keep the cell alive, and possesses the ability to generate electrical activity, usually in the form of a voltage pulse (action potential). The projections are usually distinguished as dendrites, or axons. The dendrites, which are usually branched, receive signals from impinging axons. The axon is the principle output channel of the cell, and carries an action potential from the soma to the synapse. This potential is then transmitted either chemically or electrically across the synaptic gap to the dendrites of other cells. Eventually, this either causes the impinged cell body to fire (i.e. creates an action potential) or inhibits it from firing. Exactly how neurons communicate is a particularly complex process and only a brief summary is given here. For a more extensive background see Junge, (1981); Kuffler, Nicholls & Martin, (1984); Kandel & Schwartz, (1981).

Across each cell membrane there is a potential difference of about 50-90*mV*. This is known as the resting potential, and is due to differing concentrations of sodium Na^+ and potassium K^+ ions on each side of the membrane. On one side of the membrane there is an excess of Na^+ and on the other side an excess of K^+. If the membrane allowed them to pass freely, the electrical currents created by their movements in opposite directions would cancel each other out. But the membrane acts as a selective filter. At rest only K^+ ions can pass, and not Na^+ ions. Thus, an electromotive force develops, directly related in value and sign (negative inside) to the ratio of K^+ concentrations on each side of the membrane.

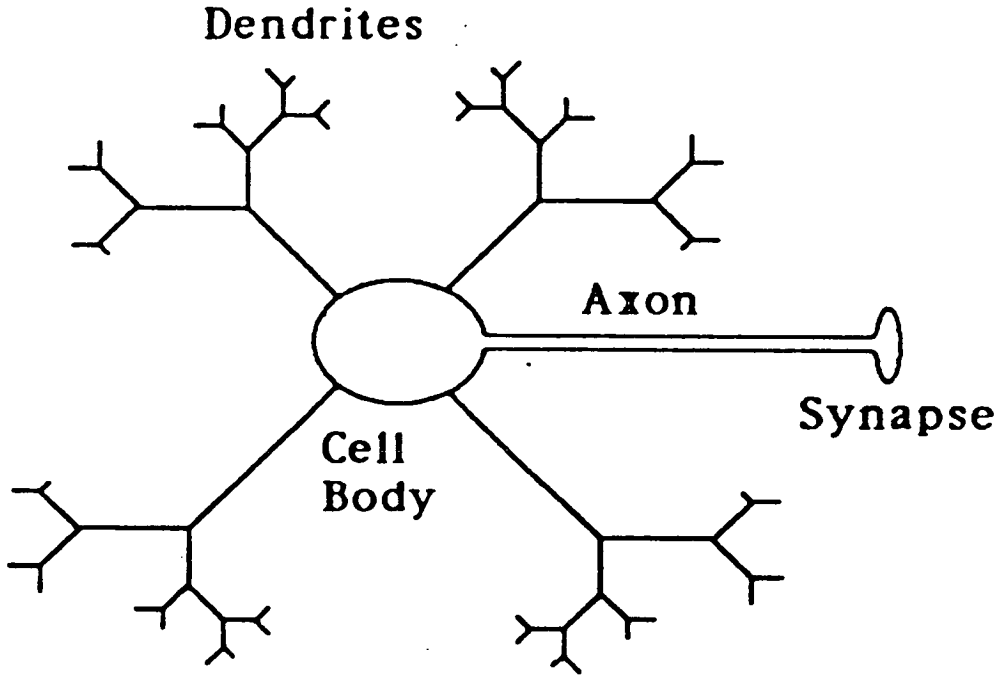

Figure 2.3 The classical neuron.

These differences in ion concentration are maintained by a protein referred to as an *enzyme pump*, which crosses the membrane, captures ions on one side and transports them back to the other. This transfer is accomplished against a gradient with energy harnessed from ATP (adenosine triphosphate), a substance produced by cell metabolism. The enzyme pump or ATPase, breaks down the ATP molecules and uses the energy so released to transport Na^+ and K^+ across the membrane. Hence ATP provides the energy needed to form a difference in ion concentrations across the membrane, and the membrane spontaneously converts this charge gradient into an electrical potential. At rest the enzyme pump and cell metabolism maintains a permanent electrochemical potential, and this potential can be used freely to produce action potentials.

The initiation of a nerve impulse occurs when an incoming signal depolarises (reduces the membrane potential from its resting value towards zero) the membrane of the axon hillock (the region of the neuron where the axon emanates from the soma). When it reaches a threshold value, (about $+10mV$), voltage-sensitive Na^+ channels in the axon membrane open, allowing Na^+ to rush into the cell (making the inside electrically more positive). This induces changes in more Na^+ channels, and allows even more Na^+ to enter the cell, and thus produces a self-generating or avalanche effect.

Since the mean open time of the channel is only about $0.7msec$, the increase in permeability to Na^+ of any part of the membrane is very brief. As the membrane potential reverses its sign and reaches a value of about $+55mV$, Na^+ conductance is suddenly inactivated. Channels selective to K^+ temporarily open and K^+ begins to move out of the cell, initiating the restoration of the resting potential. This precisely timed sequence of membrane events lasts about a millisecond or so, and is depicted in Figure 2.4.

During the brief interval when the membrane is permeable to Na^+, the inward Na^+ current spreads along the membrane causing depolarisation in the neighbouring downstream areas of the membrane. This in turn gives rise to a new regenerative impulse which consequently depolarises its downstream

neighbourhood membrane and so on down the length of the axon. The action potential is a triggered, all-or-nothing event which has a distinct threshold, and once initiated, its amplitude and duration are not determined by the amplitude and duration of the initiating event. Variations in signal can be produced by altering the frequency of spikes in a train, or by producing special patterns in a train of impulses through the combined use of hyperpolarising and depolarising currents. Frequently neurons show a low rate of spontaneous spiking (spiking without externally induced depolarisation) and this base rate of firing is increased by depolarising currents and decreased by hyperpolarising currents.

After each action potential, there is a period during which the axon cannot spike, this is called the refractory period. This occurs because the signal channels have to be reconfigured, Na^+ has to be pumped out, and the neuron membrane has to regain its balance of electric potentials.

Neurons communicate with each other via synapses. Usually an axon will form a synapse on a dendrite or on the somas of other neurons, but it may form a synapse on other axons, and in some cases dendrites form a synapse on other dendrites and on somas. Impulses reaching a synapse set up graded electrical signals in the dendrites of the neuron in which the synapse impinges, the interneural transmission being sometimes electrical and sometimes by diffusion of chemicals.

At electrical synapses, currents generated by an impulse in the presynaptic nerve terminal spread directly to the next neuron through a low resistance pathway. In the chemical synapse, a fluid filled gap between presynaptic and postsynaptic membranes prevents a direct spread of current. Instead, action potentials arriving at a chemical synapse cause vessicles containing chemical neurotransmitters to migrate to the synapse membrane and release their contents to the synaptic gap. The neurotransmitters diffuse across the synaptic gap, although some are lost from the gap. The molecules that arrive at the postsynaptic membrane interact with it to modify its membrane potential producing either an excitatory or an inhibitory synaptic potential. A particular neuron will only fire an electrical impulse

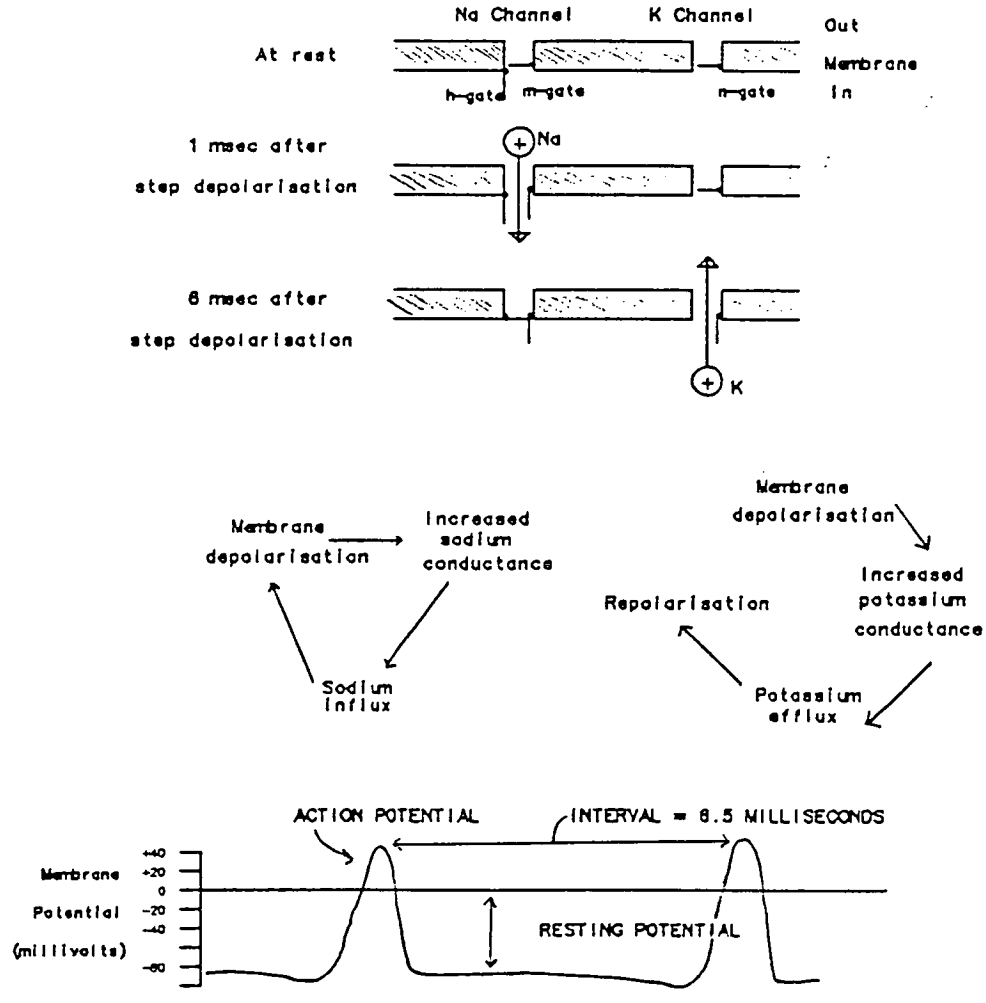

Figure 2.4 Membrane events.

along its axon if sufficient impulses reach the endbulbs impinging on its dendrites in a short period of time (period of latent summation). The impulses may actually help or hinder firing i.e. they are either excitatory or inhibitory.

As information transfer is achieved by a simple pulse train form, negative influences are obtained by having an inhibitory connection. There are thus usually two opposing paths of signal, one excitatory and the other inhibitory. The human brain has approximately 10^{12} neurons , with the number of synapses on each neuron varying from $10^3 - 10^5$. It is estimated that there are some 10^{15} connections in the human nervous system. If a suitable positive weight is assigned to each excitatory synapse, and a negative weight to each inhibitory synapse, we can say that a neuron fires only if the total weight of the synapses which receive impulses in the period of latent summation exceeds its threshold. A summary of neuron features is presented in Table 2.1.

FEATURE	ACTION POTENTIAL	SYNAPTIC POTENTIAL
Amplitude	$70 - 110mV$	$100\mu V - 10mV$
Duration	$1 - 10msec$	$5msec - 20mins$
Summation	All-or-none	Graded
Signal	Depolarising	Hyperpolarising or depolarising
Propagation	Active	Passive

Table 2.1 Summary of neuron features.

2.4 GENERAL NOTATION FOR NEUROMORPHIC MODELS

Despite being disparate, the neuromorphic models introduced in the following chapter have many features in common, and may be described as special cases of a general framework. Although various general characterisations have been

attempted, Kohonen (1977, 1984), Amari (1977a), Feldman & Ballard (1982), the framework utilised here is adapted from that of Rumelhart, Hinton & McClelland, (1986). The principle components used within this framework to define the models are shown in Table 2.2.

1. Processing units
2. State of activation
3. Output function
4. Pattern of connectivity
5. Propagation rule
6. Activation rule
7. Learning rule
8. Environment

Table 2.2 Principle components of neuromorphic systems.

Figure 2.5 illustrates the basic components of a neuromorphic system. This depicts a set of processing units, where at each point in time unit u_i has a state (of activation) denoted $s_i(t)$. This activation value is passed through function f_i to produce an output value $o_i(t)$. This output value is then passed through unidirectional links to other units in the system. Each connection has associated with it a real number called the weight, designated w_{ij}. This determines the degree of effect which the first unit has on the second unit. All the inputs are combined (usually additively), together with the unit's current activation function, using function F to give the new activation function of the unit.

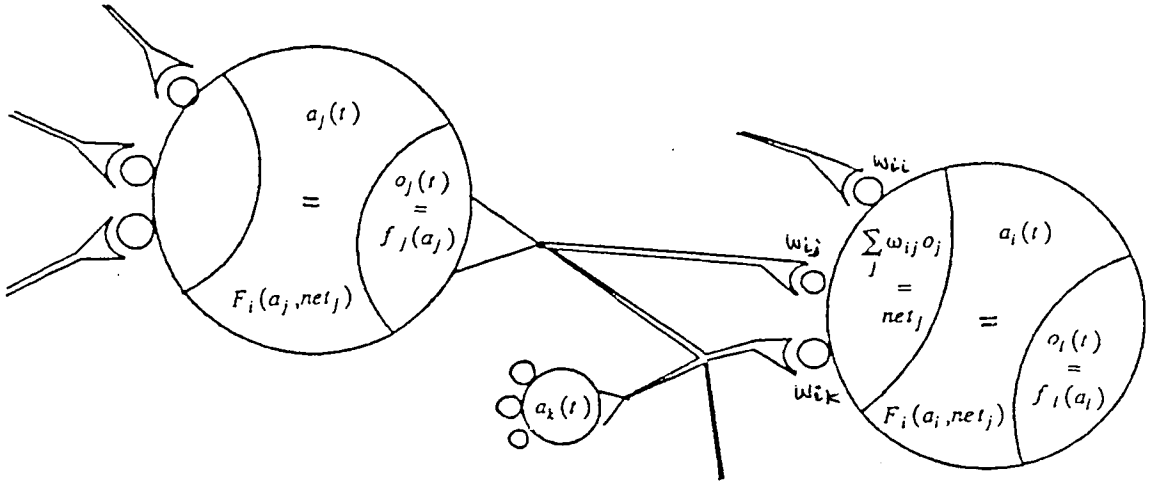

Figure 2.5 Basic components of a neuromorphic system.

2.4.1 Processing units

In some models processing units may represent conceptual objects, e.g. features letters, words or concepts, or in others they are simply abstract elements over which meaningful patterns can be defined. N is defined as the total number of units, with the i^{th} unit being designated u_i. All the processing of a neuromorphic model is carried out by these units; there is no overseer. Each unit receives input from its neighbours, and as a function of the inputs it receives computes an output value, which it sends to its neighbours. This system is inherently parallel, in that many units can carry out their computations simultaneously. Units can be grouped into three types, viz. input, output and hidden units. Input units receive signals from the system's environment, output units send signals out of the system, and hidden units have no environmental contact, i.e. their input and outputs are fully within the system. Input and output units may also be collectively called visible units.

2.4.2 State of Activation

The activation levels which units are allowed to accommodate depends upon the particular model. Activation levels may be continuous or discrete, bounded or unbounded. When activation levels are restricted to discrete values they are most often binary. In some models they are restricted to the values 0 and 1, where 1 is usually taken to mean that the unit is active, and 0 to mean that the unit is inactive. In other models the values are restricted to (+1, -1). The state of a system at time t is represented by a vector of N real numbers $s(t)$, representing the pattern of activation over the set of processing units. The activation of processing unit u_i at time t is designated $s_i(t)$.

2.4.3 Output Function

Units interact by transmitting signals to their neighbours. The strength of the signals and therefore the degree to which they interact is a function of their degree of activation. Each unit u_i has an output function $f_i(s_i(t))$, which maps the current state of activation $s_i(t)$ to an output signal $o_i(t)$.

$$o_i(t) = f_i(s_i(t)) \qquad (2.1)$$

In some cases f is the identity function $f(x) = x$. More often however f is some sort of threshold function so that a unit has no affect on another unit unless its activations exceeds a certain value. Sometimes the function f is stochastic, here the output depends in a probabilistic fashion on its activation values.

2.4.4 Pattern of Connectivity

The pattern of connections between units is dependent on the learning history of the system and determines how the system will respond to arbitrary inputs. When each unit provides an additive contribution to the input of units to which it is connected, the total input to the unit is the weighted sum of the separate inputs from each of the individual processor units. In such cases, the total pattern of connectivity can be represented by specifying the weights for each of the connections in the system. A positive weight represents an excitatory input, and a negative weight represents an inhibitory input. This pattern of connectivity is represented by a matrix W, in which each entry w_{ij} represents the strength and sense of connection from unit u_j to unit u_i. Thus the weight w_{ij} is positive if unit u_j excites unit u_i; negative if unit u_j inhibits unit u_i; and it is 0 if u_j has no direct connection to unit u_i. The absolute value of w_{ij}, specifies the strength of connection.

In some models more complex inhibition/excitation combination rules are required, and it is convenient to have separate connectivity matrices for each law of connection. Thus we can represent the patterns of connectivity by a set of

connection matrices W_i, one for each type of connection. The pattern of connectivity is particularly important, as it is this pattern which determines what each unit represents.

2.4.5 Rule of Propagation

This rule takes the output vector $\mathbf{o}(t)$, representing the output values of the units and combines it with the connectivity matrices to produce a input vector for each type of input *i* to the unit, $\mathbf{net}_i(t)$. The value net_{ij} is the net input of connection type *i* to unit *j*. The propagation rule is generally straightforward. For a system with two types of connections, inhibitory and excitatory, the net excitatory input is usually the weighted sum of the excitatory inputs to the unit. This is given by the vector product $\mathbf{net}_e = W_e\mathbf{o}(t)$. Similarly the net inhibitory effect can be written as the vector product $\mathbf{net}_i = W_i\mathbf{o}(t)$. With more complex patterns of connectivity, more complex rules of propagation are required.

2.4.6 Activation Rule

This rule (function F) combines the net inputs of each type impinging on a particular unit together with the current state of the unit to produce a new state of activation. When all connections are of one type, and F is the identity function $\mathbf{s}(t+1) = W\mathbf{o}(t) = \mathbf{net}(t)$. If F is a threshold function the net input must exceed some value before contributing to the new state of activation. Often the new state of activation depends on the previous state as well as the current input. In general therefore, we have:-

$$\mathbf{s}(t+1) = \mathrm{F}(\mathbf{s}(t), \mathbf{net}(t)_1, \mathbf{net}(t)_2 \ldots) \qquad (2.2)$$

The function is usually assumed to be deterministic. Thus, for example, if a threshold is involved it may be that $s_i(t) = 1$ if the total input exceeds some threshold value and equals 0 otherwise. Other times it is assumed that F is

stochastic. Whenever $s_i(t)$ is assumed to take continuous values it is common to assume that F is a kind of sigmoid function. In this case an individual unit can saturate and reach a maximum or minimum value of activation.

2.4.7 Learning Rule

Learning in a network is accomplished by modifying the pattern of connectivity as a function of experience. There are basically three kinds of modification:-

1. Development of new connections
2. Loss of existing connections
3. Modification of strength of existing connections

Very little work has been done on (1) and (2), although recently Le Cun, Denker & Solla (1990) have presented a scheme that allows the selective deletion of existing connections. This technique, which they call Optimal Brain Damage (OBD), permits the deletion of weights that have the least effect on the training error. Using this scheme they were able to show that OBD can be used as an automatic network minimisation procedure and as an interactive tool to suggest better architectures.

Schemes (1) and (2) above may also be considered a special case of (3), where the strengths of connections are modified through experience. Virtually all learning rules where the strengths of connections are modified are a variant of the Hebbian learning rule. Hebb (1949) suggested, without physiological evidence that:-

> *"when an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells, such that A's efficiency, as one of the cells firing B is increased" (Hebb, 1949).*

Thus if a unit u_i receives an input from another unit u_j, then if both are highly active the weight w_{ij}, from u_j to u_i should be strengthened. This can be more generally stated as:-

$$\Delta w_{ij} = g(s_i(t), t_i(t))h(o_j(t), w_{ij}) \qquad (2.3)$$

where $t_i(t)$ is a kind of teaching input to u_i. In other words the change in connection from u_j to u_i is given by the product of function $g()$, which depends on the activation of u_i, and its teaching input t_i and another function $h()$, which depends on the output value of u_j and the connection w_{ij}. The simplest form of Hebbian learning requires no teacher and the functions g and h are simply proportional to their first arguments. Here

$$\Delta w_{ij} = \eta s_i o_j \qquad (2.4)$$

where η is a constant of proportionality representing the learning rate. There are many variations on this general rule, some of which are described later.

2.4.8 Environment

In neuromorphic models, the environment is represented as a time-varying stochastic function over the space of input patterns, i.e. at any point in time there is some probability that any of the possible set of input patterns is incident on the input units. Typically, the environment is characterised by probability distributions over the set of possible input patterns, independent of past inputs and past responses of the system. Certain models are restricted in the kinds of patterns that they are able to learn: some being able to learn to respond correctly only if the input vectors form an orthogonal set; others if the inputs form a linearly independent set of vectors; whereas other models are able to learn to respond to essentially arbitrary patterns of inputs.

2.5 INTERPRETATION OF NEURAL VARIABLES

Table 2.3 summarises the mappings between neural variables and the mathematical and conceptual world. As outlined in Section 2.3 neurons are complex biochemical entities yet only a few facts about neurons and neuroanatomy have been needed to establish a base for neuromorphic model theory. Models such as the Boltzmann Machine and Hopfield network can only be regarded as highly stylised interpretations of the biological neuron. These

models use simple binary units to represent neurons. Action potentials are emitted in bursts with the information in the axonal signal being believed to reside in the pulse frequency of the burst (Perkel & Bullock 1969). Thus, the signal can be represented by a real number in a limited interval. In the Boltzmann Machine and Hopfield model this is modelled as a binary number. Hence the two possible states for units in these models may be regarded as representing a neuron firing or not firing. It is interesting to note that the properties of the action potential indicate that communication between neurons is essentially accomplished using pulse code modulation. This has architectural advantages for hardware implementations of neural networks.

The connections between units are analogous with the synaptic contacts of neurons, and the weight of the connection analogous with the synaptic strength. The energy gap for a binary unit has a role that is similar to that played by the membrane potential for a neuron, as both are the sum of the excitatory and inhibitory inputs and both are used to determine the output. Sections 3.2.3 and 3.3.5 provide more information concerning the relationship of the Hopfield model and Boltzmann Machine to the brain.

NEURAL	MATHEMATICAL	CONCEPTUAL
neuron	unit	hypotheses
average generating potential	activation	stimulus trace or short term memory
average firing frequency	output value	degree of confidence
spread of depolarisation	spread of activation	propagation of confidence
synaptic contact	connection	long term memory trace (LTM)
excitation/inhibition	positive/negative weight	positive/negative LTM trace
approximate additivity of depolarisation	summation of inputs	approximate additivity of evidence
spiking thresholds	activation spread threshold	independence from irrelevant information

Table 2.3 Interpretation of neural variables (adapted from Smolensky 1986a)

2.6 EVOLUTION OF NEUROMORPHIC MODELS

2.6.1 McCulloch and Pitts

McCulloch and Pitts (1943) approximated the brain as a set of Boolean devices, and performed the first analysis of neural networks with fixed connectivities. The threshold logic unit (TLU) or linear threshold unit is a particular type of the McCulloch and Pitts neuron. It has n inputs $x_1, \ldots x_n (n \geq 1)$, and one output

o. It is characterised by $n + 1$ numbers, its threshold θ, and the real value weights $w_1, \ldots w_n$ where w_i is associated with x_i. The unit operates on a discrete time scale $t = 1, 2, 3, 4, \ldots$, the firing of its output at time $\tau + 1$ being determined by the firing of its input at time τ, and will only fire if the total weight of the inputs stimulated at time t exceeds the threshold of the neuron.

$o(t) = 0$ $\quad i$ does not fire at time t

$o(t) = 1$ $\quad i$ does fire at time t

o may be an axonal output or a synaptic input of a neuron. The above rule may be expressed as:-

$$o(\tau + 1) = 1 \text{ only if } \sum_i w_i x_i(\tau) \geq \theta \qquad (2.5)$$

$w_i > 0$ *corresponds to an excitatory synapse (input).*

$w_i < 0$ *corresponds to an inhibitory synapse (input).*

The TLU divides the *n*-dimensional space of possible input vectors into two regions separated by a hyperplane. One region is associated with an output value 1, and the other with an output value 0. The value of the weights determines the orientation of the hyperplane. This model is perhaps the best known and arguably the most influential model of the nervous system. With it McCulloch & Pitts were able to show that it is possible to program finite networks of threshold neurons to produce input/output relations of arbitrary complexity by fixing all the connectivities in an appropriate way. Having shown how neuron-like networks could compute, the next problem was to understand how such networks could learn.

2.6.2 The Single-layer Perceptron

The combination of the McCulloch & Pitts simplified model of the neuron with Hebbian learning by changes in the synaptic junctions between neurons (Hebb, 1949) resulted in the development of the single-layer Perceptron (Rosenblatt,

1959). This model generated much interest at first because of its ability to learn to recognise simple patterns. Essentially the Perceptron consists of threshold logic units with connection weights and threshold values that may be fixed or adapted using a number of different learning algorithms. Figure (2.6) depicts a single-layer Perceptron that classifies an analogue input vector into two classes denoted A and B. The goal of learning is to minimise the error between the desired output value of the system $d(t)$ and the computed value $o(t)$. The magnitude of this error is given by $e(t) = d(t) - o(t)$. The output is calculated as follows:-

$$o(t) = \sum_{i}^{N-1} w_i(t)s_i(t) - \theta \qquad (2.6)$$

The single node computes a weighted sum of the input elements, subtracts a threshold θ and passes the result through a hard limiting non-linearity such that the output value $o(t)$ is either +1 (class A) or -1 (class B). The fixed-increment algorithm updates $w_i(t)$ according to

$$w_i(t+1) = w_i(t) + \eta e(t)s_i(t) \qquad (2.7)$$

$$0 \leq i \leq N-1$$

$$d(t) = \begin{cases} +1 \text{ if input class A} \\ -1 \text{ if input class B} \end{cases}$$

The parameter η is a positive constant that controls the stability and rate of learning. The error may be thought of as a surface over the weight space, the space of possible values for the weight vector W. Vector s can then be defined as a vector in this weight space which points in the direction of steepest descent for the error. Thus, the algorithm takes a fixed-size step in the direction of steepest descent. If training instances are linearly separable the fixed increment algorithm converges in a finite number of steps. This algorithm can be improved in a number of ways by choosing how far in the direction of gradient to move at each step.

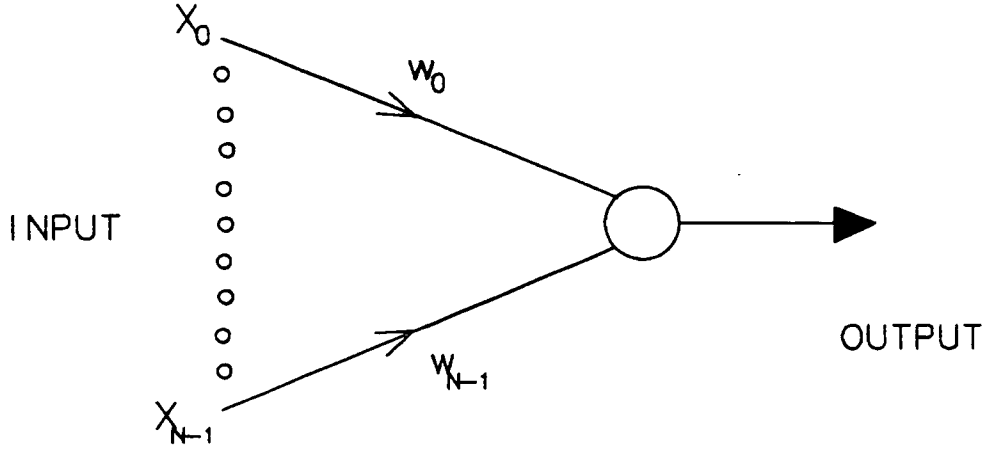

Figure 2.6 Single-layer perceptron.

The LMS (least mean square) or Widrow-Hoff rule is identical to the Perceptron convergence procedure (equation 2.7) except that in the LMS rule the hard limiting non-linearity is made linear or is replaced by a threshold logic non-linearity. For linear activation functions the Widrow-Hoff or delta rule minimises the squares of the differences between the actual and desired output values summed over the output units and all pairs of the input/output vectors. This rule was first used in the ADALINE (adaptive linear neuron) of Widrow & Hoff (1960), who were pursuing engineering applications of trainable TLU's.

The tremendous interest in Perceptrons was only dampened when Minsky and Papert (1969) produced a mathematical analysis of the computing powers of the single-layer Perceptron highlighting its limitations. The main limitation of the Perceptron convergence procedure (or Widrow-Hoff rule) is that it cannot be applied to devices in which there is more than one-layer of modifiable weights between the input array and the output unit. In other words the procedure cannot cope with hidden units.

Rosenblatt made the following interesting statement at the Conference on Mechanisation of Thought Processes (1959):-

> *"computers seem to share two main functions with the brain, a) decision making, based on logical rule, and b) control, again based on logical rule. The human brain performs these functions together with a third: interpretation of the environment. Why do we hold interpretation of the environment to be so important? The answer, I think, is to be found in the laws of thermodynamics. A system with a completely self contained logic can never spontaneously improve its ability to organise and to draw valid conclusions from information." (Rosenblatt, 1959).*

Rosenblatt appeared to be saying that there were some things a computer couldn't do, but that the brain and Perceptrons could do because of their statistical properties. But a computer program can be written to simulate the behaviour of statistical perceptrons, and indeed Rosenblatt was one of the pioneers in the digital simulation of this type of problem.

What Rosenblatt was perhaps really saying was that using the symbol manipulating capabilities of the computer to directly simulate the logical processes involved in decision making, theorem proving and other intellectual

activities of this sort would be inadequate to mimic the power of the human brain. The task he thought, could only be accomplished if computers simulated Perceptrons.

2.6.3 Grossberg

Grossberg (1982) is credited with producing the first coherent mathematical theory of brain functions. He claims to explain learning, perception and behaviour with a small number of mathematical laws and organising principles. Grossberg investigated various output functions and deduced that linear functions tend to amplify noise; slower than linear tend to generate an asymptotic uniform distribution; and that faster than linear select only those features of highest amplitude. He argues that the output function must be a sigmoid, providing a threshold below which noise is suppressed, and above which features can be enhanced. Of the many activation rules that Grossberg has presented, the majority are of the form

$$s_j(t+1) = \alpha s_j(t) + (\beta - s_j(t))net_{aj}(t) + (\gamma - s_j(t))net_{ij}(t) + I_j(t) \qquad (2.8)$$

where:

α is the decay rate

β represents the maximal degree of excitation of the unit

γ is much smaller than β and represents the maximal amount the unit can be inhibited below the resting value of 0.

I_j represents the input sources from outside the network. These are usually from other neurons or sensory transducers.

In this rule the excitatory and inhibitory inputs appear separately in the activation rule.

Grossberg has also studied many learning schemes, the most studied being:

$$\Delta w_{ij} = \eta s_i(o_j - w_{ij}) \qquad (2.9)$$

which is a variation of Hebb's law (Grossberg, 1976).

Grossberg has developed his *adaptive resonance theory* (ART), which he originally introduced in Grossberg (1976), and presented two classes of networks, ART1 and ART2 (Carpenter & Grossberg 1987, Grossberg 1987). These networks (ART1 networks are for binary-valued inputs and ART2 networks are for continuous-valued inputs) are trained without supervision and form categories for the input data with the coarseness of the categories being determined by the value of a particular parameter known as *attentional vigilance.* ART networks have applications in pattern classification and the categorisation of data.

2.6.4 Hopfield Model

Hopfield (1982) presented a model that is reminiscent of the Ising model from theoretical physics (Newell & Montroll, 1953). This net can be used as an associative memory or in the solution of optimisation problems. The models use binary state units, which are updated one at a time. The output of each unit is fed back to all other units via weights. During operation, an unknown pattern is presented to the net, forcing the output of the net to match the unknown pattern. The net iterates in discrete time steps until the outputs no longer change on successive iterations. The pattern specified by the unit outputs after convergence is the net output.

Hopfield and others (Cohen & Grossberg, 1983) have proven that this net converges when the weights are symmetric (i.e. $w_{ij} = w_{ji}$) and unit outputs are updated asynchronously. Hopfield (1982) defined an energy function for the net such that minima of energy correspond to stable firing patterns for the network. In addition an information storage algorithm was presented which allows a network of N units to store up to about 0.15N uncorrelated patterns. The optimal storage capacity for an N unit network however, is 2N uncorrelated patterns, indicating that the Hopfield model is not an efficient content-addressable memory.

2.6.5 Boltzmann Machine

The concept of minimising an energy function to arrive at particular stable states played a significant part in the development of the Boltzmann Machine (Hinton et al. 1984). This model employs binary units connected with symmetric weights. The update rule which is employed switches each unit into which ever of its two states minimises its contribution to the global energy of the system. Because the connections are symmetric the differences between the energy of the whole system with the i^{th} unit on and its energy with the i^{th} unit off can be determined locally. A probabilistic decision rule is used to prevent the network from becoming stuck at local minima that are not globally optimal. This rule allows occasional jumps to higher energy states. To reliably find good minima, large jumps in energy are allowed at first, and then these are slowly reduced. This method is analogous to annealing and is known by the term *simulated annealing*. The particular formulation of the Boltzmann Machine leads to a general learning rule which modifies connection strengths between units in such a way as to allow the network to develop an internal model which captures the underlying patterns of its environment. The Boltzmann Machine learning algorithm was the first supervised learning procedure presented that allowed hidden unit learning. Smolensky (1984) also investigated a similar scheme to that used in the Boltzmann Machine which he called *harmony* theory.

2.6.6 Multi-layer Perceptron

The recent development of learning algorithms (Rumelhart, Hinton and Williams, 1985) for the Multi-layer Perceptron has allowed many of the limitations of the single-layer Perceptron to be overcome. Although these algorithms cannot be proven to converge as with the single-layer Perceptron, they have been shown to be capable of solving many problems of interest. The abilities of the Multi-layer Perceptron stem from the non-linearities used within each node. If the nodes were linear elements, then a single-layer net with suitably chosen weights could exactly duplicate those calculations performed by the multi-layer net. The back-propagation learning algorithm allows feed-forward nets of units with

continuous differentiable non-linear activation functions to be trained. The algorithm is basically a generalisation of the LMS algorithm, which uses a gradient search technique to minimise a cost function equal to the mean square difference between the desired and the actual net outputs. Application of the rule involves two phases. During the first phase the input is presented and propagated forward through the network to compute the output value o_j for each unit. This output is then compared with the target value t_j, resulting in an error term (t_j-o_j) for each output unit. The second phase involves a backward pass through the network during which the error term is passed to each unit in the network and the appropriate weight changes are made.

2.7 SUMMARY

The preceding sections have provided a brief introduction to neuromorphic models. The origins of these models have come from insights gained into the workings of the brain. The considerable extent of this work extends back to ancient times. Indeed, it was not until 1824 that the nerve cell made its first entry into scientific literature. Further discoveries ensued and by 1952 it was possible to demonstrate the complete ionic workings of the nerve impulse. This important finding allowed the propagation of the nerve impulse, whether in a squid's giant axon or a neuron of the cerebral cortex, to be explained by similar, if not identical mechanisms.

The structure of the cerebral cortex was established as consisting of a very large number of neurons interacting primarily through the activation and inhibition of one another's activity. Each neuron is highly interconnected, with perhaps tens of thousands of interconnections. Furthermore, the processing speed of each neuron was discovered to be slow, and measured in milliseconds rather than the picoseconds common in today's computers. However, such a neural system does allow computations that are faster than is possible with even the largest and fastest of todays computing machines.

Such observations inspired numerous parallel processing models consisting of interconnected elementary processing elements. Much interest was generated in the studies of a model introduced by Rosenblatt in the 1950's, the Perceptron, because it was able to learn to recognise simple patterns. However, a critique of Perceptrons by Minsky & Papert (1969) dampened this tremendous interest by showing the limitations on the power of the single-layer Perceptron.

The problem, as noted by Minsky & Papert (1969) is that whereas there is a very simple guaranteed learning rule for all problems that can be solved without hidden units, there is no equally powerful learning rule for networks with hidden units. One response to this is to attempt to develop a learning procedure capable of learning an internal representation adequate for performing the task at hand. One such development has been the Boltzmann Machine, which makes use of stochastic binary units. It is so called because the equilibrium behaviour of the network is described by the Boltzmann distribution, providing a simple relationship between the energy of a state and its equilibrium probability. Later, the back-propagation learning rule was introduced, which is in essence a generalisation of the Perceptron learning rule for multi-layer networks. However, the back-propagation algorithm is not guaranteed to find the optimal set of weights for a problem since the learning is a process of gradient descent on an *error surface* that may contain local minima. The Boltzmann Machine learning algorithm, provides a formally guaranteed procedure for performing gradient descent in a global error measure.

With the back-propagation algorithm, the error signals returning from the downstream layer provide indirect information about conditions of the other units in the layer, but the Boltzmann Machine is not limited to that type of architecture. The Boltzmann Machine requires symmetric links, an appropriate decision rule, and the absence of connections from a unit to itself, but makes no other restrictions on the topology of the network employed. Visible units can be connected to each other, hidden units can be connected to other hidden units and the units can be arranged in layers. The formal proof of the Boltzmann Machine learning algorithm is completely insensitive to such issues.

For these reasons the Boltzmann Machine algorithm has been adopted in the work considered here for the processing of speech pathology data, and its development is presented in the next chapter. Details of the operation of the Boltzmann Machine and the learning procedure are also discussed in full. For completeness, details of the back-propagation algorithm are included along with its relationship to the Boltzmann Machine.

3 THERMODYNAMIC NEUROMORPHIC MODELS

3 THERMODYNAMIC NEUROMORPHIC MODELS

3.1 INTRODUCTION

The cerebral cortex consists of approximately 3×10^{10} neurons interconnected at synapses within a complex arrangement of axons, dendrites and cell bodies. Although the exact *modus operandi* of brain cell metabolism is not fully understood, the information carried by a single neuron appears to be very simple, and is transmitted by a primitive pulse train (action potential) propagating along the axons. About two-thirds of the cortical neural cell population are pyramidal cells. However, we can assume (after Braitenburg, 1978), that the state of the cortex as a whole is fully determined by the states of the pyramidal cells. The remaining cells play the role of inter-neurons, modifying the effective interactions between the pyramidal cells. Such a system may be represented as a large assembly of identical elements, each of which is characterised by an internal state, its firing activity and its connections to each other at synapses:-

> *"this system is reminiscent of those studied in statistical mechanics and the question naturally arises whether the analogy can be pursued far enough for the techniques of statistical mechanics to be applicable. If so, it would permit the introduction in neural network theories, of such concepts as the order parameters, phase transitions, correlation functions (the evoked potentials) or the use of tools such as the partition function, mean field theories etc ..." (Peretto, 1984).*

Statistical mechanics deals with systems comprising of a large number of elementary units. Because of the large number of units involved it is not possible to follow their individual trajectories so a statistical approach is used to extract the properties of the macroscopic system from microscopic averages. It is the science of thermodynamics which provides the broad framework for describing the relationships between the microscopic and macroscopic properties of such systems.

Various writers have profitably used the spin system analogy to describe the problems of neural assemblies. Little (1974) explored an analogy between the existence of persistent states (short term memory) in simple neural networks and the occurrence of long range order in Ising spin systems. This model was able to account for the action potential patterns observed in small neural assemblies such as the pyloric system of lobsters (Little & Shaw, 1978; Thompson & Gibson, 1981). In a landmark paper, Hopfield (1982), introduced an asynchronous neural model of content addressable memory based on an analogy with spin glasses. Many papers followed, building on this new theoretical approach to the analysis of neural network models. Later, Hopfield (1984) introduced networks of graded response neurons; Peretto (1984) assessed the possibilities of applying statistical mechanics to the functioning of large neural networks; Hinton, Sejnowski & Ackley (1984) introduced the use of noise to find a global minimum; Smolensky (1984), Smolensky & Riley (1984) investigated a similar scheme called "harmony" theory; Kinzel (1985) discussed learning and pattern recognition in spin glass models; Amit, Gutfreund and Sompolinsky (1985a, 1985b) gave a statistical analysis of the associative model based on the equilibrium theory; Toulouse G., Dehane S., & Changeux J-P. (1986) presented a spin glass model of learning by selection.

Of particular interest here are the models devised by Hopfield, Hinton et al. and Smolensky because they allow a rigourous mathematical description using statistical mechanics techniques. Details of the operation of the Hopfield model and the analogy with Ising spin systems are discussed in this chapter together with a biological interpretation of their findings. The addition of the concept of simulated annealing to the Hopfield model is also discussed along with learning procedures in the Boltzmann Machine. Smolensky's work on Harmony theory is also discussed, and brief details of both Cauchy and Gaussian Machines is provided. Finally, despite being a deterministic solution to learning in multi-layer nets, the back-propagation algorithm is discussed along with its relationship to the Hopfield and Boltzmann Machine models.

3.2 HOPFIELD MODEL

3.2.1 Introduction

In 1982 Hopfield introduced a parallel network of neuron-like elements for building content-addressable memories. This model was based on an analogy with the models used in the theoretical analysis of certain materials called spin glasses. These are magnetic materials which have a random orientational ordering (glass) of magnetic moments (spins). The spin sites are randomly interconnected by positive and negative competing interactions.

> *"Any physical system whose dynamics in phase space is dominated by a substantial number of locally stable states to which it is associated can therefore be regarded as a general content-addressable memory. The physical system will be a potentially useful memory if, in addition, any prescribed set of states can readily be made the stable states of the system." (Hopfield, 1982).*

The key question posed by Hopfield was whether these neuromorphic systems would exhibit useful computational properties such as stable content-addressable memories and an ability to create categories of generalisation. This issue has also been examined by Smith & Davidson (1962), Harmon (1964), Little (1974), Amari (1977b) and Amari & Akikazu (1978) amongst others.

A general content-addressable memory (CAM) is capable of retrieving an entire memory item on the basis of sufficient partial information. If, for example, using an illustration similar to that given by Hopfield (1982), the item stored in memory is:-

" Newell G. F. & Montroll E. W. Rev. of Mod. Phys. 25, 353-389 (1953)"

the *key* input "& Montroll (1953)" might be sufficient to retrieve the complete item, provided that the key only occurs once in the CAM. An ideal CAM is also error-correcting and would be capable of retrieving this particular item from the input "Mintroll, (1953)" provided that the CAM did not actually contain an entry

for "Mintroll, (1953)". Only relatively simple forms of CAM have been implemented in computer hardware, with more complex tasks like error correction of the accessing information usually undertaken in software.

Little & Shaw (1978) and Kohonen (1984) have also proposed using parallel networks of neuron-like elements as content-addressable memories. These models use the concept of storing memory states as fixed points of the dynamics of the neural network. These fixed points are intended to be attracting, so any initial configuration of neurons sufficiently close to the memory state will be attracted onto that memory state. The set of all initial configurations of neurons which lead to a given memory state is called the *basin of attraction* of that memory state. It is desirable to have a well-behaved basin of attraction around each memory state because if the basin becomes too complicated, the CAM will not be uniformly robust in its error correction.

Hopfield's (1982) model consisted of binary units that were symmetrically connected and updated one at a time with a deterministic update rule. This update rule behaves in such a way as to reduce (or at worse not increase) an overall measure which he called "energy" (because of the analogy with physical systems). Consequently, repeated iterations are guaranteed to find an energy minimum.

Hopfield thought in terms of energy, because his networks behave very much as thermodynamic systems, which seek minimum energy states. This analogy of networks falling into energy minima in a manner similar to physical systems has provided an important conceptual tool for analysing neuromorphic processing mechanisms. This energy measure is essentially a cost function which reflects (the negation of) the degree to which the desired constraints are satisfied. These constraints are:

(i) constraints imposed by other units
(ii) the *a priori* strength of the hypothesis (captured by adding the bias)
(iii) direct evidence (input value times the activation value of the unit)

States to which the network converges (when outputs no longer change on successive iterations) are local minima of this cost function. This means that the networks are performing optimisation of a well-defined function. Unfortunately, there is no guarantee that the network will find the best minimum. Hopfield's original networks were prone to this problem with local "energy minima". With binary units, if the net input to a unit is positive it takes on its maximum value; if it is negative, the unit takes on its minimum value (otherwise its value remains unchanged). Binary units are more prone to local minima because the units do not get the opportunity to communicate with one another before committing to one of their two states. Hopfield (1984), has since presented a version in which units take on a continuum of values to help deal with the problem of local minima in his model.

However, finding the local minimum which is closest to the initial state is useful if the minima are used to represent "items" in a memory, and the initial states are queries to memory which may contain missing or erroneous information. The network then simply finds the minimum that best fits the query. Thus, this network may be used as an associative memory or for solving some types of optimisation problem.

3.2.2 Hopfield Model

The model postulated by Hopfield (1982) has its origins in McCulloch and Pitts (1943) and Hebb (1949). It is capable of storing information, as well as performing certain computational tasks such as error correction and nearest neighbour searches. Each unit *i* in the network may be in one of two states $s_i = +1$ or $s_i = 0$, (see Figure 3.1). Hopfield uses the states +1 and -1 though, because his model was derived from physical systems called spin glasses in which spins are either *up* or *down*. Provided the units have thresholds, models that use +1 and -1 can be translated into models that use 1 and 0 and have different thresholds.

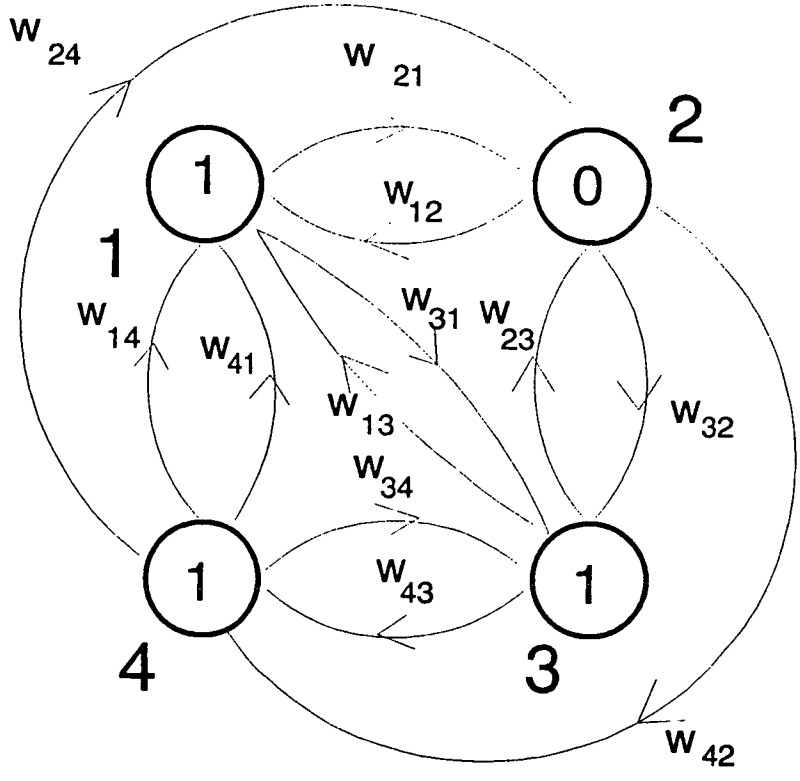

Figure 3.1 Four unit, fully interconnected Hopfield network.

The link weight between unit j and i is given by w_{ij}, with all such links being symmetrical i.e $w_{ij} = w_{ji}$. Each unit i has a fixed threshold θ_i which unless otherwise stated, is chosen to be 0. Only one unit updates its state at any instant in time, and each unit updates repeatedly with only a short finite time between updates. Hopfield (1982) and also Cohen & Grossberg (1983) showed that these conditions of symmetric weights and asynchronous updates jointly rule out the possibility of a permanent oscillation in the network and therefore guarantee that the system will settle into a single state.

Hopfield (1982) defined an energy function over the states of the units in the model as:-

$$E = -\frac{1}{2} \sum_{j \neq i} \sum w_{ij} s_i s_j \qquad (3.1)$$

The factor of one half appears because the summation counts each link twice, once from i to j and once from j to i. The energy of a global state depends on the states of the individual units and the values of the link weights. Because the connections are symmetrical, the difference between the energy of the whole system with the i^{th} unit *off* and its energy with the i^{th} unit *on* can be determined locally by the i^{th} unit. Thus the *energy gap* or change in E, due to Δs_i is given by:-

$$\Delta E = -\Delta s_i \sum_{j \neq i} w_{ij} s_j \qquad (3.2)$$

The updating rule requires that each unit (selected at random) is switched to whichever of its two states yields the lower total energy given the current states of the other units. Therefore the rule for minimising the energy contributed by a unit is to adopt the *on* or +1 (true) state if its total input from the other units equals or exceeds its threshold. If the total input from the other units is less than the threshold the *off* or 0 (false) state is adopted. Thus the state at site i is determined by:-

$$s_i' = H\left\{\sum_{j \neq i} w_{ij} s_j\right\} = \begin{cases} +1, & \text{if } \sum_{j \neq i} w_{ij} s_j > \theta_i \\ 0, & \text{if } \sum_{j \neq i} w_{ij} s_j < \theta_i \end{cases} \quad (3.3)$$

where H is the Heaviside step function. In the +1, -1 case the step function is replaced by a *sign* function. The state of the net changes until it enters a state of minimal energy in the sense that no change in any one of the variables s_i will lower the value of E. There may be a number of different such states, which are known as local minima. Global minimisation is not guaranteed.

The update rule thus involves the link weights directly connected to a unit, and the states of all the units at the other ends of those links. This places practical limitations on the number of units possible in the numerical simulation of fully interconnected networks. Simulations of some very large fully interconnected networks using an ICL Distributed Array Processor (DAP) have been described by Bruce et al. (1986) and Forrest (1988) amongst others. The DAP consists of a 64×64 array of bit-serial processing elements whose single instruction multiple data (SIMD) parallelism is apparently readily exploited for these simulations. Forrest (1988) presents results from numerical simulations carried out for networks of 512, 1024 and 2048 units on the DAP. A unit update rate of over 1700 units per second for a 512 unit network was claimed.

In using the Hopfield model for computation, certain units are designated as inputs and these have their values clamped so that the updating rule is not applied to them. Updating is then repeatedly applied to all the other units allowing the network to settle to its local energy minimum. The result may then be determined from the designated output units. This vector of stable states constitutes a stored item in memory. The basic operation of the network is to converge to a stable state provided it is initialised with a nearby state vector (in the Hamming sense).

Since there is no algorithm specifying which unit changes at the next time step, the corresponding input function may yield different values depending on the sequence of choices made. For a given search problem, weights are chosen for the network so that E is a measure of the overall constraint violation. The Hopfield

(1982) update rule is then used to maximise the constraint satisfaction. This, as noted previously only allows local minima to be found and does not give a global minima.

The information storage algorithm for determining the weights w_{ij} adopted by Hopfield is that identified with Hebb (1949). To store a set of *P* nominated patterns on a given number of units N $\{s_i^c = 0, 1; c = 1, \ldots P; i = 1, \ldots, N\}$ as stable states, w_{ij} the symmetric weight matrix is defined as:-

$$w_{ij} = \sum_{c=1}^{P} (2s_i^c - 1)(2s_j^c - 1) \qquad (3.4)$$

but with $w_{ji} = 0$. This scheme is based on the sum of the outer products of the *P* pattern vectors. The symmetry and zero-diagonal nature of the matrix **W** imply the existence of an energy function which is monotonic decreasing if nodes are updated serially. If w_{ji} is non-zero and too large a step is taken, it is then possible for the energy to increase at each step and the model become unstable.

Since the simplest identification of states stored in memory is that they correspond to the fixed points of the dynamics of the net, memory states may be regarded as local minima of this energy surface/function. These fixed points are supposed to be attracting, so that any initial configurations of neurons sufficiently close to a memory state are called the basin of attraction. An item is recalled by specifying enough of its content to ensure that the net is initially in the basin of attraction of the energy minimum which corresponds to that item. It is also desirable to have a well-behaved basin of attraction around each memory state; if the basin is too complicated, the CAM will not be uniformly robust in its error correction. To ensure well defined "memory states" for each nominal vector provided requires that $P \ll N$.

The Hopfield model, with the Hebbian rule is relatively inefficient in terms of its storage capacity. If too many patterns are stored, the basin of attraction becomes too complicated, and the net may converge to a novel spurious pattern different from all exemplar patterns. Such a spurious pattern will produce a "no match"

output when the net is used as a classifier. Computer simulations by Amit, Gutfreund & Sompolinsky (1987), have shown that it will only store a maximum of about $P = 0.145N$ uncorrelated patterns on an N-node network. It has been shown that the optimal storage capacity for an N-node network is $2N$ for uncorrelated patterns, (Gardner & Derrida, 1988; Baldi & Venkatesh, 1987) and more if the patterns are correlated (Gardner, 1987). The number of classes is thus typically kept well below $0.145N$. For example a Hopfield net for only 10 classes might require more than 70 nodes and more than roughly 5,000 connection weights.

A second limitation of the Hopfield net is that an exemplar pattern will be unstable if it shares many bits in common with another exemplar pattern. Here an exemplar is considered unstable if it is applied at time zero and the net converges to some other exemplar. Hopfield (1982) states that for $N = 100$, a pair of random memories should be separated by 50 ± 5 Hamming units so as not to be confused. (The Hamming distance between two binary states is defined as the number of places in which the digits are different). Hopfield also studied the case when $N = 100$ and the number of memories stored $P = 8$. In this case P consisted of seven random memories with the eighth made up with Hamming distances of only 30, 20 or 10 from one of the other seven memories. Hopfield reported that at a distance of 30, both similar memories were usually stable; at a distance of 20, the minima were usually distinct but displaced and at a distance of 10, the minima were often fused. It is difficult to make a direct comparison to the "& Montroll (1953)" example of Section 3.2.1 as content-addressability depends on the number of units in the network and the number of patterns stored. The author is not aware of a general result for the percentage of common bits allowed before the model becomes unstable. The problem of having too many bits in common however can be eliminated and performance improved by a number of orthogonalisation procedures (Wallace 1985, Grant 1986).

3.2.3 Relationship to the Brain

The Hopfield model is a highly stylised interpretation of the biological neuron. Most neurons are capable of generating a train of action potentials when the average potential across their membrane is held well above its normal resting value. The mean rate at which action potentials are generated as a function of the mean membrane potential is depicted in Figure 3.2, and can be seen to be sigmoidal.

Perkel & Bullock (1969) were able to show that the biological information sent to other neurons often lies in a short-time average of the firing rate. When this is so, the details of individual action potentials can be neglected and Figure 3.2 may be regarded as a smooth input-output relationship. The linear associative networks of Rosenblatt (1962) and Kohonen (1977) emphasise the linear central region of Figure 3.2. However, the collective effects in particle dynamics produced by particle interactions come from a non-linear dependence of forces on positions of the particles. Thus, Hopfield replaced the linear input-output relationship with a step function as shown in Figure 3.2.

Delays in synaptic transmission and in the transmission of impulses along axons and dendrites produce a delay between the input of a neuron and the generation of an effective output. These delays are modelled by a single parameter, known as the stochastic mean processing time.

Synapses are activated by arriving action potentials. The input signal to a cell *i* can be taken to be $\sum_j w_{ij} s_j$ where w_{ij} represents the effectiveness of the synapse. Figure 3.2 thus depicts an input-output relationship for a neuron. It is assumed that w_{ij} is produced by previous experience.

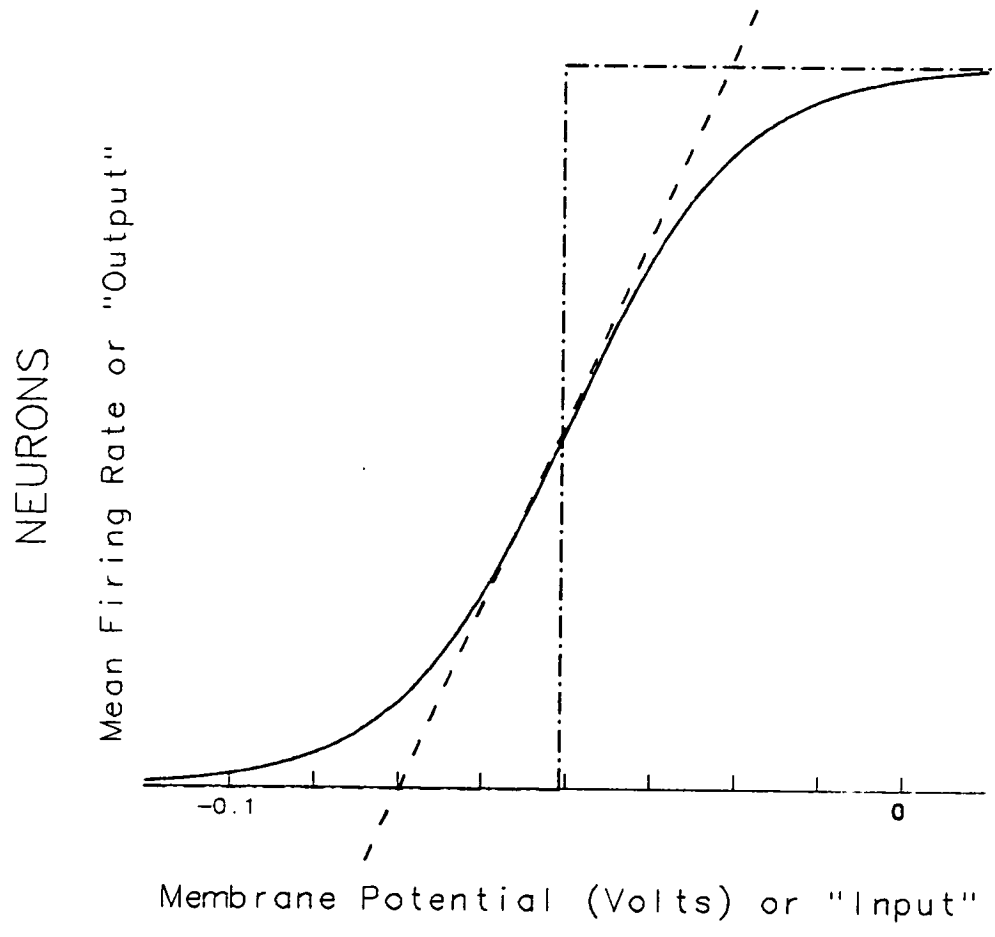

Figure 3.2 Firing rate versus membrane voltage for a typical neuron (solid line), dropping to 0 for large negative potentials and saturating for positive potentials. The broken lines show approximations used in modeling. (From Hopfield, 1982).

3.2.4 Ising Spin Analogies

For reasons of completeness a brief introduction to Ising Models is given so that the analogy established by Hopfield may be observed. Ising models are based on a simple model of ferromagnetism proposed by Ising in his doctorate dissertation of 1925. These models have been the subject of considerable investigation since then. The model is based on the view that ferromagnetism is caused by an interaction between the spins of certain electrons in the atoms making up a crystal. The Ising model is not considered to be a very realistic model of ferromagnetism, but it is equivalent to a very good model of binary substitutional alloy and an interesting model of a gas or liquid.

Each unit *i* of a system comprising of a large number *N* of elementary units, has a spin co-ordinate which can take only two values $\sigma_i = \pm 1$. The spin is considered to be either *up* or *down*. The Ising spin models have no intrinsic dynamics thus once given the set *I* of the internal states, it remains unchanged forever. The existence of an unspecified mechanism which allows spins to flip must be assumed if the system is to evolve towards an equilibrium state. Equilibrium is fully determined by the rules the spin directions have to follow.

A quantity called the molecular field h_i is associated with every site *i*. This is a linear function of the internal states σ_i of the surrounding spins $h_i = \sum_j J_{ij}\sigma_j$.

The coefficients J_{ij} are called the exchange interactions. The internal state σ_i eventually flips so as to fulfil the alignment condition $h_i\sigma_i > 0$. If the interactions J_{ij} are local, an extensive quantity *H(I)* can be found:-

$$H(I) = -\sum_i h_i\sigma_i = -\sum_i \sum_j J_{ij}\sigma_i\sigma_j \qquad (3.5)$$

The equilibrium properties of the Ising spin models are then given by:-

$$\rho(I) = Z^{-1} \exp -\beta H(I) \qquad (3.6)$$

β is a noise source parameter and H is known as the Hamiltonian of the system. (In physics, the Hamiltonian determines which states are most probable:- the states with lowest energy are most probable at all temperatures, and states of high energy have negligible probability except at high temperatures).

By adding an external field h_i^0 to the molecular field, the term $-\sum_i h_i^0 \sigma_i$ is introduced in the Hamiltonian given by Equation (3.5) above.

H remains an extensive quantity (i.e. a function of the state) for fully interconnected systems if one assumes that the long range actions J_{ij} scale as N^{-1};

$$H(I) = -\frac{1}{N}\sum_{ij} J_{ij}\sigma_i\sigma_j - \sum_i h_i^0\sigma_i \qquad (3.7)$$

Now Hopfield defines the input to a unit i as:-

$$net_i = \sum_j w_{ij}s_j \qquad (3.8)$$

where w_{ij} is the synaptic efficiency between unit j and unit i. The state of unit i is $s_i=1$ if $net_i > \theta_i$, $s_i=0$ if $net_i < \theta_i$. θ_i is the threshold potential of unit i.

By allowing $\sigma_i = 2s_i - 1$ i.e. $\sigma_i = +1$ when $s_i = 1$, $\sigma_i = -1$ when $s_i = 0$ then $\sigma_i(s_i - \theta_i) > 0$ or:-

$$\sigma_i h_i > 0 \qquad (3.9)$$

with:

$$h_i = \sum_j J_{ij}\sigma_j + h_i^0$$

$$J_{ij} = \frac{w_{ij}}{2}; \qquad h_i^0 = \sum_j \frac{w_{ij}}{2} - \theta_i$$

The analogy established by Hopfield, between this description of a neural network and the Ising spin model may be observed. Note that the model does not involve any specific dynamical mechanism.

An extensive quantity, an Hamiltonian can therefore be associated to the Hopfield neural model. This is:-

$$H(I) = -\sum_i h_i \sigma_i = -\sum_{ij} J_{ij}\sigma_i\sigma_j - \sum_i h_i^0 \sigma_i \qquad (3.10)$$

Both σ_i and σ_j both belong to the same set *I*. In general $J_{ij} \neq J_{ji}$ but the $\sigma_i\sigma_j$ and the $\sigma_j\sigma_i$ terms can be grouped so that the Hamiltonian (3.10) can be rewritten as:-

$$H(I) = -\sum_{<ij>} J'_{ij}\sigma_i\sigma_j - \sum_i h_i^0 \sigma_i \qquad (3.11)$$

with $J'_{ij} = J'_{ji} = (J_{ij} + J_{ji})$ and the summation is carried out on neuron pairs $<ij>$.

The interactions J_{ij} can *a priori* be of either sign and therefore the Hamiltonian (Equation 3.11) is an Ising spin glass Hamiltonian. In spin glass models, one often assumes that the interaction distribution is symmetrical and its average is zero. The biological counterpart is an equal number of excitatory and inhibitory synapses. Also it is believed that most neurons work in the vicinity of $h_i^0 = 0$ to enhance their sensitivity. This corresponds to zero field spin glasses.

In neural networks the synaptic efficiencies are not random parameters, they are assumed to result from a learning procedure. According to Hebb (1949) one has:-

$$w_{ij} = J'_{ij} = \sum_{c=1}^{P} \sigma_i^c \sigma_j^c \qquad (3.12)$$

where $I^c = \{\sigma_i^c\}, c = 1, \ldots, p$ are P learned patterns. Hopfield uses Monte-Carlo calculations to look for the minima of H and has reported that $P \ll N$ if the minima are to be close to the learned states.

In neural assembly modelisations, it is assumed that an unspecified noise source exists which permits the neurons to change their internal states. This is exactly as for Ising spin statistics, which demands an undetermined mechanism that allows the spins to flip. The equilibrium probability distribution is given by Equation (3.6) and *H(I)* is given in Equation (3.11) where β is a noise source parameter.

3.3 BOLTZMANN MACHINE

3.3.1 Introduction

Hopfield (1982) showed how networks of binary symmetric units could be used as memories that were stored as local minima (see Section 3.2). For hard optimisation problems however, the system must try to escape from local minima in order to find the configuration that is the global minimum given the current input.

One standard technique used in solving hard optimisation problems is that of gradient descent. Here the values of the variables in the problem are modified in whatever direction reduces the cost function or *energy*. For difficult problems, gradient descent becomes stuck at local minima that are not globally optimal. This is an inevitable consequence of only allowing downhill moves. If jumps to higher energy states occasionally occur, it is possible to break out of local minima, but it is not obvious how the system will then behave and it is far from clear when uphill steps should be allowed.

Kirkpatrick, Gelatt and Vecchi (1983) proposed a stochastic approximation method for solving combinatorial optimisation problems. It was based on an algorithm invented by Metropolis et al. (1953) that simulates the behaviour of

many-particle systems in thermal equilibrium. This stochastic method used the physical analogy of annealing to guide the use of occasional uphill steps. This search method is called *simulated annealing*.

A simple modification was made to Hopfield's updating rule by Hinton, Sejnowski & Ackley (1984) that allowed parallel networks to implement simulated annealing and hence find the global minimum. Hinton et al. named these networks *Boltzmann Machines* because at thermal equilibrium the relative probability of two global states is determined solely by their energy difference and follows a Boltzmann distribution. This simple relationship made it possible to derive a learning algorithm for the Boltzmann Machine that provably optimises a global measure of how well the network is performing its task. Thus the Boltzmann Machine is capable of learning the underlying constraints that characterise a domain simply by being shown examples from the domain. The network modifies the strengths of its connections so as to construct an internal model that produces examples with the same probability distribution as the samples it is shown. Then, when shown any particular example, the network can *interpret* it by finding values of the variables in the internal model that would generate the example. When shown a partial example, the network can complete it by finding internal variable values that generate the partial example and use them to generate the remainder.

3.3.2 Minimising Energy

The structure of a Boltzmann Machine is similar to that of a Hopfield network. Given a set of units which take on binary values $s_i = 0$ or $s_i = 1$, connected with symmetric weights w_{ij} the overall energy of a particular configuration is given by:-

$$E = -\sum_{i<j} w_{ij} s_i s_j + \sum_i \theta_i s_i \qquad (3.18)$$

where θ_i is the threshold of the i^{th} unit. Because of the symmetric connections, a local decision can be made as to whether or not a unit should be in the *on* or *off* state to minimise this energy. If the unit is *off* (0), it contributes nothing to the above equation, but if it is *on* (1) it contributes:-

$$\Delta E_i = \sum_j w_{ij} s_j - \theta_i \qquad (3.19)$$

In order to minimise the overall energy, then a unit should turn *on* if its input exceeds its threshold and *off* otherwise. This is the Hopfield rule for binary threshold units.

The effect of θ_i on the global energy or on the energy gap of an individual unit is equivalent to the effect of a link with strength $-\theta_i$ between unit *i* and a special unit that is by definition held in the *on* state. This *true unit* simplifies the computations by allowing the threshold of a unit to be treated in the same manner as the links. The value $-\theta_i$ is called the *bias* of unit *i*. If a permanently active *true unit* is assumed to be part of every network, then Equations (3.18) and (3.19) can be written as:-

$$E = -\sum_{i<j} w_{ij} s_i s_j \qquad (3.20)$$

$$\Delta E_i = \sum_j w_{ij} s_j \qquad (3.21)$$

3.3.3 Simulated Annealing

The simple deterministic algorithm, as used in the Hopfield model does not allow the global minimum to be found because it suffers from the standard weakness of gradient descent methods and becomes stuck in local minima that are not globally optimal. This as previously observed is not a problem in Hopfield's system because local energy minima of his network are used to store *items*. If the system started near some local minimum, the desired behaviour is to fall into that minimum and not to find the global minimum. For optimisation tasks, however, the system must try to escape from local minima in order to find the configuration that is the global minimum given the current input. A simple way to get out of local minima is to occasionally allow jumps to configurations of higher energy.

An algorithm with this property was introduced by Metropolis et al. (1953) to simulate the behaviour of many-particle systems in thermal equilibrium. In each step of this algorithm, the state s_i of particle i is given a small random displacement and the resulting change, ΔE_i in the energy of the system is computed. If $\Delta E_i \leq 0$, the displacement is accepted, and the configuration with the displaced atom is used as the starting point for the next step. The case $\Delta E_i > 0$ is treated probabilistically and the probability that the configuration is accepted is:-

$$P(\Delta E_i) = e^{-\frac{\Delta E_i}{k_B T}} \qquad (3.22)$$

where k_B is Boltzmann's constant, and T is temperature. Random numbers uniformly distributed in the interval (0,1) are a convenient means of implementing the random part of the algorithm. One such number is selected and compared with $P(\Delta E_i)$. If it is less than $P(\Delta E_i)$, the new configuration is retained, if not, the original configuration is used to start the next step. By repeating the basic step many times, the thermal motion of atoms in thermal contact with a heat bath at temperature T is simulated. This choice of $P(\Delta E_i)$ has the consequence that the system evolves into a Boltzmann distribution, (Metropolis et al., 1953; Kindermann & Snell, 1980).

As T goes to 0, this distribution will tend to an impulse (or set of impulses) corresponding to the state (or states) of minimum energy, that is, to the value of s that minimises $E(s)$ globally.

One serious difficulty, however, is that attaining thermal equilibrium might take a very long time at low temperatures. Kirkpatrick, Gelatt & Vecchi (1983) used another physical analogy to guide the use of occasional uphill steps. To find a very low energy state of a metal, the best strategy is to melt it and then to slowly reduce its temperature. This process is called annealing, and so they named their search method *simulated annealing*. Thus the system to be optimised is first *melted* at a high effective temperature, then the temperature lowered by slow

stages until the system *freezes* and no further changes occur. This method has been used successfully to design electronic systems and on travelling salesman problems (Kirkpatrick et al., 1983).

Thus, the solution to the problem of local energy minima can be solved in essentially the same way that flaws are dealt with in crystal formation by annealing. This is a process whereby a material is heated and then cooled very slowly. The idea is that as the material is heated, the bonds between the atoms weaken and the atoms are free to reorient relatively freely. They are in a state of high energy. As the material is cooled, the bonds begin to strengthen, and as the cooling continues the bonds eventually become sufficiently strong so that the material freezes. To minimise the occurrence of flaws in the material, it must be cooled slowly enough so that the effects of one particular coalition of atoms has time to propagate from neighbour to neighbour throughout the whole material before the material freezes. The cooling must be especially slow as the freezing temperature is approached. During this period the bonds are quite strong so that the clusters will hold together, but they are not so strong that atoms in one cluster might not change state so as to align with those in an adjacent cluster, even if means moving into a momentarily more energetic state. In this way annealing can move a material toward a global energy minimum.

Boltzmann Machines (Hinton, Sejnowski & Ackley, 1984) are networks with a binary Hopfield net structure that use as their update rule a form of the Metropolis algorithm that is suitable for parallel computation. If the energy gap between the 1 and 0 states is ΔE_i then regardless of the previous state the probability of a unit's next state being *on* is given by:-

$$p_i = \frac{1}{1 + e^{\frac{-\Delta E_i}{T}}} \qquad (3.23)$$

where T is a parameter which acts like temperature of a physical system. The time required to reach *thermal equilibrium* grows extremely rapidly as T is lowered. A more practical way of zeroing in on the state of least energy is to start

at a high temperature and gradually lower it. This corresponds to annealing a physical system (Kirkpatrick et al. 1983). Early in the search only large energy differences are significant, and the system quickly makes a coarse attempt at the problem, avoiding states of extremely high energy. As the system cools down, smaller energy differences become significant, and more and more states are avoided as the search focuses on states with energies close to the minimal value. If the cooling is done gently, the state of maximal energy should be found in much less time than by giving T a constant low value.

The idea of implementing constraints as interactions between stochastic elements was proposed by Moussouris (1974) who discussed the identity between Boltzmann distributions and Markov random fields. The idea of using simulated annealing in parallel networks has been investigated independently by several groups. Geman & Geman (1984) established limits on the allowable speed of the annealing schedule, and showed that simulated annealing can also be very effective for removing noise from images. They were able to prove that a necessary and sufficient condition for convergence to the global minimum based on strictly local sampling requires that the time schedule of changing the fluctuation variance, described in terms of the artificial cooling temperature $T_a(t)$, be inversely proportional to a logarithmic function of time, given a sufficiently high initial temperature T_0.

$$T_a(t) = \frac{T_0}{\log(1+t)} \qquad (3.24)$$

Unfortunately, the value of T_0 for which Geman & Geman were able to guarantee convergence is in general very high, so that convergence becomes impractically slow.

Hinton & Sejnowski (1983) showed that the use of binary stochastic elements could solve some problems that plague other relaxation techniques, in particular the problem of learning the weights. Smolensky (1984) investigated a similar scheme he called *Harmony*. The essential update rule employed in all of these models is probabilistic and is given by:-

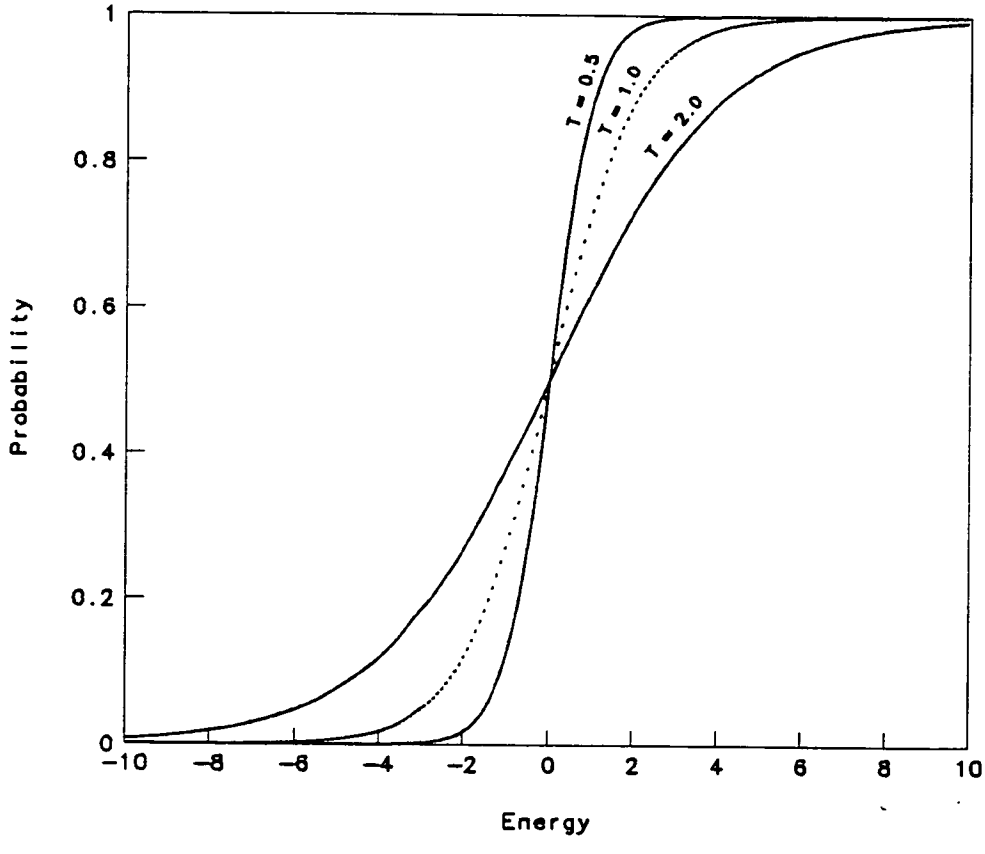

Figure 3.3 Probability values as a function of net input and temperature.

$$\text{probability}(s_i = 1) = \frac{1}{1 + e^{\frac{net_i}{T}}} \qquad (3.25)$$

where T is the temperature. It is not accidental that these three models all choose the same update rule. This rule is drawn directly from physics and there are important mathematical results that, in effect, guarantee that the system will end up in a global minimum if the system is annealed slowly enough.

Figure 3.3 shows the probability values as a function of net input and the temperature. Several observations can be made. First if the temperature is 0, the unit takes on its maximum and minimum values with equal probability. Second, if the net input is large enough, the unit will always take on its maximum value no matter what value the temperature is; and if the net input is sufficiently negative, the unit will take on its minimum value no matter what the temperature. Third, as the temperature approaches 0, the function becomes deterministic and takes on its maximum value if the net input is positive and its minimum value if the net input is negative. The *zero temperature* case is identical to the Hopfield binary unit model.

Simulated annealing with the Boltzmann Machine requires a cooling schedule inversely proportional to log time and is computationally slow. Another approach to simulated annealing is the Cauchy Machine (Szu, 1986) which uses the properties of the Cauchy distribution to perform fast simulated annealing with a cooling schedule inversely proportional to time (see Section 3.4).

3.3.4 Boltzmann Machine Learning Algorithm

At thermal equilibrium at temperature *T*, the Boltzmann distribution gives the relative probability that the system will occupy state A, as against state B:-

$$\frac{P_A}{P_B} = e^{-\frac{(E_A - E_B)}{T}} \qquad (3.26)$$

where P_A is the probability of being in the A^{th} global state, and E_A is the energy of that state.

> *"The Boltzmann distribution has some particularly interesting mathematical properties and it is intimately related to information theory. In particular, the difference in the log probabilities of two global states is just their energy difference (at a temperature of 1). The simplicity of this relationship and the fact that the equilibrium distribution is independent of the path followed in reaching equilibrium are what make Boltzmann Machines interesting." (Hinton et al., 1984).*

By using the decision rule in Equation (3.25) and running the network freely without any input from the environment until it reaches thermal equilibrium at some finite temperature the relationship between a global state and its energy is given by Equation (3.26). It is therefore possible to control the probabilities of global states by controlling their energies. Because the energy is a linear function of the weights (Equation 3.18) this leads to a simple relationship between the log probabilities of global states and the individual connection strengths:-

$$\frac{\delta \ln P_\alpha^-}{\delta w_{ij}} = \frac{1}{T}(s_i^\alpha s_j^\alpha - p_{ij}^-) \qquad (3.27)$$

where s_i^α is the binary state of the i^{th} unit in the α^{th} global state and P_α^- is the probability at thermal equilibrium, of global state α of the network when none of the visible units are clamped (the lack of clamping is denoted by the superscript $^-$. p^-_{ij} is the probability of finding the i^{th} and j^{th} units on together when the system is at equilibrium. Equation (3.27) shows that the effect of a weight on the log probability of a global state can be computed from purely local information because it only involves the behaviour of the two units that the weight connects. This makes it easy to manipulate the probabilities of global states provided the desired probabilities are known (Hinton & Sejnowski, 1983).

The units of the Boltzmann Machine partition into two groups of units, anon-empty set of visible units and a possibly empty set of hidden units. External input drives the visible units into various possible states with various probabilities. The hidden

units are never clamped by the environment, and can be used to *explain* underlying constraints in the input patterns that cannot be represented by pairwise constraints among the visible units.

Unfortunately it is normally unreasonable to expect the environment to specify the required probabilities of entire global states of the network. The task that the network must perform is defined in terms of states of the visible units, and so the environment or teacher only has direct access to the states of these units. The difficult learning problem is to decide how to use the hidden units to help achieve the required behaviour of the visible units. A learning rule which assumes that the network is instructed from outside on how to use *all* of its units is of limited interest because it evades the main problem which is to discover appropriate representations for a given task among the hidden units.

In statistical terms, there are many kinds of statistical structure implicit in a large ensemble of environmental vectors. The separate probability of each visible unit being active is the first-order structure and can be captured by the thresholds of the visible units. The $\frac{v^2}{2}$ pair-wise correlations between the v visible units constitute the second-order structure and this can be captured by the weights between pairs of units. All structure higher than second-order cannot be captured by pairwise weights between visible units. For example, if the ensemble consists of vectors (1 1 0), (1 0 1), (0 1 1), and (0 0 0), each with probability 0.25. There is clearly some structure here because four of the eight possible 3-bit vectors never occur. However the structure is entirely third-order. The first-order probabilities are all 0.5, and the second-order correlations are all 0, so in considering only these statistics, this ensemble is indistinguishable from the ensemble in which all eight vectors occur equiprobably.

The Widrow-Hoff rule or perceptron convergence procedure (Rosenblatt, 1962) is a learning rule which is designed to capture second-order structure and it therefore fails on the above example. If the first two bits are treated as an input

and the last bit is treated as the required output, the ensemble corresponds to the exclusive-OR (XOR) function which is one of the examples used by Minsky and Papert (1969) to show the strong limitations of single-layer perceptrons.

In one sense it is true that networks with pairwise connections can never capture higher than second-order statistics. In another sense it is false. By introducing extra units which are not part of the definition of the original ensemble, it is possible to express the third-order structure of the original ensemble in the second-order structure of the larger set of units. In the example given we can add a fourth component to get the ensemble {(1101),(1010),(0110),(0000)}. It is now possible to use the thresholds and weights between all four units to express the third-order structure in the first three components. i.e. an extra "feature detector" is introduced which in this example detects the case when the first two units are both on. Then it is possible to make each of the first two units excite the third unit, and use strong inhibition from the feature detector to overrule this excitation when both of first two units are on. The difficult problem in introducing the extra unit is deciding when it should be on and when it should be off, i.e. deciding what feature it should detect (there are six different ways of using the extra unit to solve the task).

The weights in the network should be chosen so that the hidden units represent significant underlying features that bear strong, regular relationships to each other and to the states of the visible units. The hard learning problem is to determine what these features are, i.e. to find a set of weights which turn the hidden units into useful feature detectors that explicitly represent properties of the environment which are only implicitly present as higher order statistics in the ensemble of environmental vectors.

Another view of learning is that the weights in the network constitute a generative model of the environment. The object then is to find a set of weights so that when the network is running freely, the patterns of activity that occur over the visible units are the same as they would be if the environment was clamping them. The number of units in the network and their interconnectivity define a space of

possible models of the environment, and any particular set of weights defines a particular model within this space. The learning problem is to find a combination of weights that gives a good model given the limitations imposed by the architecture of the network and the way it runs.

Making certain assumptions permits derivation of a measure of how effectively the weights in the network are being used for modeling the structure of the environment, and it is also possible to show how the weights should be changed to progressively improve this measure. Firstly, assume that the environment clamps a particular vector over the visible units and keeps it there long enough for the network to reach thermal equilibrium with this vector as a boundary condition (i.e. to "interpret it). Secondly, assume (unrealistically) that there is no structure in the sequential order of the environmentally clamped vectors. This then means that the complete structure of the ensemble of environmental vectors can be specified by giving the probability, $P^{+}(V_{\alpha})$, of each of the 2^{v} vectors over the *v* visible units. $P^{+}(V_{\alpha})$ does not depend on the weights in the network because the environment clamps the visible units.

A particular set of weights can be said to constitute a perfect model of the structure of the environment if it leads to exactly the same probability distribution of visible vectors when the network is running freely with no units being clamped by the environment. Because of the stochastic behaviour of the units , the network will wander through a variety of states even with no environmental input and it will therefore generate a probability distribution, $P^{-}(V_{\alpha})$ over all 2^{v} visible vectors. This distribution can be compared with the environmental distribution, $P^{+}(V_{\alpha})$. In general, it will not be possible to exactly match the 2^{v} environmental probabilities using the weights among the *v* visible and *h* hidden units because there are at most $(v+h-1)\frac{(v+h)}{2}$ symmetrical weights and *(v + h)* thresholds. However, it may be possible to do very well if the environment contains regularities that can be expressed in the weights. An information theoretic measure (Kullback, 1959) of the distance between the environmental and free-running probability distributions is given by:-

$$G = \sum_{\alpha} P^{+}(V_{\alpha}) \ln \frac{P^{+}(V_{\alpha})}{P^{-}(V_{\alpha})} \qquad (3.28)$$

where $P^{+}(V_{\alpha})$ is the probability of the α^{th} state of the visible units in *phase*$^{+}$ when their states are determined by the environment, and $P^{-}(V_{\alpha})$ is the corresponding probability in *phase*$^{-}$ when the network is running freely with no environmental input.

G is never negative and is only zero if the distributions are identical. *G* is actually the distance in bits from the free running distribution to the environmental distribution (using base 2 logarithms).

It is possible to improve the network's model of the structure of its environment by changing the weights so as to reduce *G*. To perform gradient descent in *G* it is important to know how *G* will change when a weight is changed. But changing a single weight changes the energies of a significant number of all the global states of the network, and it changes the probabilities of all the states in ways that depend on *all* the other weights in the network.

To minimise *G*, Hinton et al. (1984) show that from Equations (3.26) and (3.18) the partial derivative of *G* is:-

$$\frac{\delta G}{\delta w_{ij}} = -\frac{1}{T}(p^{+}_{ij} - p^{-}_{ij}) \qquad (3.29)$$

where p^{+}_{ij} is the average probability of two units being in the *on* state when the environment is clamping the states of the visible units, and p^{-}_{ij} is the corresponding probability when the environmental input is not present and the network is free running. (Both these probabilities must be measured at equilibrium). To minimise *G*, it is sufficient to observe p^{-}_{ij} and p^{+}_{ij} when the network is at *thermal equilibrium* and to change each weight by an amount proportional to the difference between these two probabilities:-

$$\Delta w_{ij} = \epsilon(p^{+}_{ij} - p^{-}_{ij}) \qquad (3.30)$$

where ϵ scales the sign of each weight change. The change in a weight depends only on the behaviour of the two units it connects, even though the change optimises a global measure, and the best value for each weight depends on the values of all the other weights. Interestingly, it does not matter whether the weight is between two visible units, two hidden units, or one of each.

If there are no hidden units, it can be shown that *G*-space is concave (when viewed from above) so that simple gradient descent will not get trapped at local minima. With hidden units, however, there can be local minima that correspond to different ways of using the hidden units to represent the high-order constraints that are implicit in the probability distribution of environmental vectors.

Once *G* has been minimised the network will have captured as well as is possible the regularities in the environment. There are a number of variations that can be made to the learning algorithm and indeed various techniques that can be applied to speed up performance of the algorithm. Some of these are discussed in Chapter 4.

3.3.5 Relationship to the Brain

The membrane potential of a neuron may be regarded as being similar to the energy gap for a binary unit. Both are the sum of the excitatory and inhibitory inputs and both are used to determine the output state. However, neurons produce action potentials rather than binary outputs. When an action potential reaches a synapse, the signal it produces in the postsynaptic neuron rises to a maximum and then exponentially decays with the time constant of the membrane (typically around 5 msec for neurons in the cerebral cortex).

The summed input from all the recently active binary units is represented by the energy gap. If the average time between updates is identified with the average duration of a postsynaptic potential, then the binary pulse between updates can be considered as an approximation to the postsynaptic potential. Although the shape of a single binary pulse differs significantly from a postsynaptic potential,

the sum of a large number of stochastic pulses is independent of the shape of the individual pulses and depends only on their amplitudes and durations. The binary approximation may therefore be reasonable for large networks having the large fan-ins typical of the cerebral cortex (around 10,000).

The membrane potential of a neuron is graded, but if it exceeds a fairly sharp threshold an action potential is produced followed by a refractory period lasting several msec during which another action potential cannot be generated. If Gaussian noise is added to the membrane potential, then even if the total synaptic input is below the threshold, there is a finite probability that the membrane potential will reach the threshold.

The amplitude of the Gaussian noise determines the width of the sigmoidal probability distribution for the neuron to fire during a short time interval, and it therefore plays the role of temperature in the model. A cumulative Gaussian is a very good approximation to the required probability distribution. It has been shown using intracellular recordings from neurons that there is stochastic variability in the membrane potential of most neurons, which is due in part to fluctuations in the transmitter released by presynaptic terminals.

In a Boltzmann Machine all connections are symmetrical. It is very unlikely that this assumption is strictly true of neurons in the cerebral cortex. However, if the constraints of a problem are inherently symmetrical and if the network on average approximates the required symmetrical connectivity, then random asymmetries in a large network will be reflected as an increase in the Gaussian noise in each unit.

3.4 CAUCHY MACHINE

One of the major problems confronting simulated annealing is convergence speed. An algorithm that uses Cauchy noise and has a cooling schedule that is inversely linear in time has been present by Szu (1986). This fast simulated annealing (FSA) is a semi-local search and consists of occasional long jumps.

With a cooling schedule that is inversely linear in time it is fast compared with classical simulated annealing (CSA) which is inversely proportional to the logarithmic function of time.

$$T_c(t) = \frac{T_0}{(1+t)} \tag{3.31}$$

A distributed Cauchy Machine is an architecture that can execute a parallel version of the fast simulated annealing algorithm. Binary unit output and analogue unit input are used with the Cauchy noise version of the Metropolis acceptance function, as well as the Cauchy noise generating random states.

The output s_i is set to one with the following acceptance criteria:

$$P(net_i) = \frac{1}{2} + \frac{1}{\pi} arctan\left(\frac{net_i}{T}\right) \tag{3.32}$$

where T and net_i are the (artificial) temperature parameter and the total input to the i^{th} unit respectively. If the uniform random number $N\ [0\text{-}1]$ is smaller than $P(net_i)$, the output s_i is set to one. T is updated according to and output s_i is set to zero otherwise.

3.5 GAUSSIAN MACHINE

A neuron in a Gaussian Machine model (Akiyama et al., 1989) has graded output responses like a Hopfield Machine (Hopfield, 1984) and behaves stochastically like a Boltzmann Machine so that the system can escape from local minima under appropriate system parameters. In Gaussian Machines, the output function of a neuron is deterministic, just as in the Hopfield Machine, but the output value is influenced by random noise added to each input, and as a result forms a probabilistic distribution. The Gaussian Machine model includes the McCulloch-Pitts model, the Hopfield Machine and the Boltzmann Machine as special cases of its definition.

The Gaussian Machine is so called because its significant property is derived from a Gaussian distribution of random noise added to the neural input of the model. Each unit u_i receives the output value o_j from the other neurons through the input links. The output values o_j of u_i are graded values with a range $0 < o_j < 1$. A synaptic weight w_{ij} is defined for each interconnecting link from u_j to u_i. The neuron also has an input bias θ_i. Thus the net input net_i to u_i is:-

$$net_i = \sum_{j=1}^{N} w_{ij} o_j + \theta_i + \epsilon \qquad (3.33)$$

where ϵ is the error (per unit time) on the input caused by random noise. This noise term ϵ is essential to the Gaussian Machine because it breaks the determinism of each neuron and helps the system to escape from local minima.

The unit u_i is activated by the net input net_i. The activation value s_i of net_i is changed in accordance with the difference equation:

$$\frac{\Delta s_i}{\Delta t} = -\frac{s_i}{\tau} + net_i \qquad (3.34)$$

where τ is the time constant of the neuron. The asynchronous update schedule of Equation (3.34) has the advantage of damping oscillation, as reported by Hopfield (1982). Δt has the range of $0 < \delta t \leq 1$ in consideration of convergency.

The output value o_i is determined by a sigmoid function:-

$$o_i = f(s_i) = \frac{1}{2}\left(\tanh\left(\frac{s_i}{s_0} \right) + 1 \right) \qquad (3.35)$$

where s_0 is the reference activation level, which defines the gain of the curve. If s_0 approaches zero, then the function becomes the unit step function in the same manner as the McCulloch-Pitts threshold model. Equation (3.35) is identical to that of the Hopfield Machine.

The net input net_j of the Gaussian Machine is always influenced by random noise $\in$. The error is propagated to the activation value s_j and then affects the determination of the output value o_j. Alspector (1989) used a similar technique to this in an electronic model of a 6 neuron network.

The noise $\in$ obeys a Gaussian distribution with mean of zero and variance σ^2. The deviation σ depends on a parameter T known as temperature and defined as:-

$$\sigma = kT \tag{3.36}$$

where the constant k is given by $k = \sqrt{8/\pi}$.

The convergence behaviour of a Gaussian Machine depends on w_{ij} and biases θ_i. To describe the convergence behaviour of a system, Hopfield (1984) presented the following energy function:-

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}o_i o_j - \sum_{i=1}^{N}\theta_i o_i + \frac{1}{\tau}\sum_{i=1}^{N}\int_{1/2}^{o_i} g^{-1}(o)do \tag{3.37}$$

where the third term disappears when s_0 approaches zero.

Hopfield has shown that the energy function E monotonically decreases with time when the matrix w_{ij} is symmetric, all principal diagonal elements w_{ii} are zero, and there is no noise. In a Hopfield Machine, the allowing of vague decisions based on a continuous output values contributes to faster convergence. However, it cannot escape from local minima because of its determinism.

A neuron of a Gaussian Machine changes its state stochastically with E tending to decrease. Thus the Gaussian Machine model can escape from a local minimum when the noise is sufficiently large.

The McCulloch-Pitts model is characterised by a binary output, deterministic decision rule and instantaneous activation in time. If s_0=0 and T=0, the output function f becomes the unit step function and the decision is completely deterministic.

The Hopfield Machine is characterised by a graded output, deterministic decision rule and continuous activation in time. The Hopfield Machine is identical to the Gaussian Machine model when $T=0$ because the Gaussian Machine has the same output function controlled by s_0.

The Boltzmann Machine model is characterised by a binary output, stochastic decision rule and instantaneous activation in time. The activation value s_j is equal to net_j in this model and is calculated by Equation (3.33). The selection of binary output is described probabilistically. The output value o_j is set to 1 with the following probability

$$P(o_i = 1) = \frac{1}{1 + \exp^{(-\Delta E_i / T)}} \qquad (3.38)$$

where ΔE_i is identical to both net_j and s_j, and T is the temperature parameter in the Boltzmann Machine. The cumulative Gaussian distribution is a sigmoid function and fits very well to the curve of the probability function on the Boltzmann Machine model. The two curves have the same gradient at s_j (or ΔE) = 0. This can be achieved by setting the coefficient $k = \sqrt{8/\pi}$. The difference between the curves never exceeds 0.002 (Akiyama et al., 1989), and has the same behaviour as a Boltzmann Machine at temperature T.

3.6 HARMONY MACHINE

> *"...the privileged unconscious phenomena, those susceptible of being conscious, are those which... affect most profoundly our emotional sensibility ... Now, what are the mathematic entities to which we attribute this character of beauty and elegance ... ? They are those whose elements are harmoniously disposed so that the mind without effort can embrace their totality while realising the details. This harmony is at once a satisfaction of our aesthetic needs and an aid to the mind, sustaining and guiding ... Figure the future elements of our combinations as something like the unhooked atoms of Epicurus ... They flash in every direction through the space ... like the molecules of a gas in the kinematic theory of gases. Then their mutual impacts may produce new combinations."* (Poincaré, 1913).

The basic mathematics of harmony theory is rather similar to that of the Boltzmann Machine, although the structure and motivation are different. Both the Harmony Machine and Boltzmann Machine use bidirectional, symmetric links and discrete, stochastic units. Unit behaviour is governed by the sigmoid function with a model parameter controlling the *temperature* of the units. The energy function of a Boltzmann Machine corresponds to the negative of the *harmony function* defined over states of a Harmony Machine. The dynamics of the system can be viewed as minimising energy, or, equivalently, as maximising harmony.

Whereas the Boltzmann Machine can be considered as an arbitrarily interconnected set of homogeneous units, Harmony theory presupposes two distinct layers of units (Figure 3.4). A harmony network consists of a lower layer of *representational feature* units (which contains the visible units) and an upper layer of *knowledge units* (hidden units). The feature units take on activation values ± 1, whereas the knowledge units take on values 0 and 1. The feature units may be regarded as corresponding to the featural description of a situation. In a complete description, each feature is either present (+1) or absent (-1). The knowledge units on the other hand, may be thought of as bits of knowledge about what configurations of features go together. Knowledge units may be either active or inactive. When a knowledge unit is active, it can be viewed as concluding that the configuration of input features it is looking for is present in the environment. When a knowledge unit is inactive it can be viewed as concluding that the evidence does not warrant such an assertion. All connections in a harmony model are symmetric, and all connections are between features and knowledge units. Thus, a given feature may either excite a knowledge unit that is consistent with it, inhibit a knowledge unit that is inconsistent with it, or have no effect on a knowledge unit to which the feature is irrelevant. Similarly, knowledge units specify certain configurations of features that are consistent with the knowledge represented by that unit. Thus a knowledge unit may activate those features that are consistent with the unit, inhibit those that are inconsistent, or not connected with those that are irrelevant to the contents of that unit. Neither features nor knowledge units are directly connected to one another. All connections in the system are ± 1.

However, each knowledge unit has a strength designated σ. The strength corresponds to the degree that the *knowledge* unit in question insists that the features to which it is connected are present at the input.

Harmony theory is so named because, for any configuration of input features the system finds the configuration of knowledge units that is maximally consistent, or harmonious, with the featural constraints. The configurations of active knowledge units may be seen as an *interpretation* of the input features.

In addition to creating an interpretation of a set of input features, the knowledge units themselves can complete missing features in a way that is maximally consistent with those features that are fixed (clamped) and the set of knowledge units. This is the so-called completion problem.

A state of harmonium is determined by the values of the lower and upper level nodes. Such a state is determined by a pair (**r**, **s**) consisting of a representation vector **r** and an activation vector **s**. A harmony function assigns a real number $H_K(\mathbf{r}, \mathbf{s})$ to each such state. The harmony function has as parameters the set of knowledge vectors and their strengths: $\{(\mathbf{K}_\alpha, \sigma_\alpha)\}$.

The basic requirement on the harmony function *H* is that it be additive under decompositions of the system (in physics, one says that *H* must be an extensive quantity). This means that if a network can be partitioned into two unconnected networks, the harmony of the whole network is the sum of the harmonies of the parts.

The harmony function *H* can be written in the following way:-

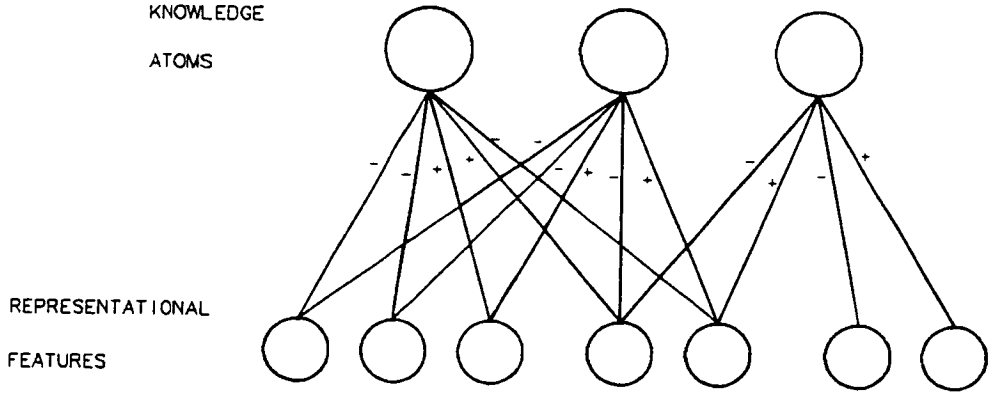

Figure 3.4 A graphical representation of a Harmony Machine. The nodes denote stochastic processors, and the links denote communication lines.

$$harmony = \sum_i \sigma_i s_i h_i \qquad (3.39)$$

Here *i* ranges over the knowledge units and h_i is a measure of the degree to which the current set of feature values is consistent with knowledge unit *i*. The variable σ_i is a strength or importance value associated with unit *i*. The variable h_i is given by

$$h_i = \frac{\sum_j r_j k_{ij}}{n_i} - \kappa \qquad (3.40)$$

Here *j* ranges over features, r_j is the activation of representational feature *j*, and n_i is the number of non-zero connections to unit *i*. The variable k_{ij} is given by

$$k_{ij} = \left\{ \begin{array}{c} 1 \text{ if positive connecion} \\ -1 \text{ if negative conection} \\ 0 \text{ if no connection} \end{array} \right\} \qquad (3.41)$$

In other words, the total harmony is given by the sum of contributions of each of the knowledge units. If a knowledge unit is not activated ($s_i = 0$), there is no contribution. If it is active ($s_i = 1$), then it contributes an amount that is proportional to the product of its importance, σ_i, and a term representing the consistency of that unit with the current pattern of activation among the representational features. This consistency term, h_i, is the proportion of relevant features that are consistent minus the proportion that are inconsistent, less a constant κ. Consider first the case in which κ is 0. In this case, turning on unit *i* will contribute a positive amount to the overall harmony of the system whenever the number of consistent features exceeds the number of inconsistent features. If κ is near 1, then it will contribute to the overall harmony only when all, or nearly all, of its features match the template for the unit.

Once a node has computed the difference in harmony ΔH between its two possible states, the likelihood ratio for adopting the two states is $e^{\frac{\Delta H}{T}}$. Converting this to the absolute probability of changing value gives:-

$$prob(change) = \frac{e^{\frac{\Delta H}{T}}}{1 + e^{\frac{\Delta H}{T}}} \qquad (3.42)$$

where T is the computational temperature. The mapping with statistical physics allows harmony theory to exploit a computational technique for studying thermal systems that was developed by Metropolis et al. (1953). This technique uses stochastic or Monte-Carlo computation to simulate the probabilistic dynamical system under study.

The relationship between probability and harmony is mathematically identical to the relationship between probability and (minus) energy in statistical physics: the Gibbs or Boltzmann law. This is the basis of the isomorphism between cognition and physics exploited by harmony theory. In statistical physics, H is called the *Hamiltonian function*; it measures the energy of a state of a physical system. In physics T is the temperature of the system. In harmony theory , T is called the computational energy of the cognitive system. When the temperature is very high, completions with high harmony are assigned estimated probabilities that are only slightly higher than those assigned to low harmony completions; the environment is treated as more random in the sense that all completions are estimated to have roughly equal probability. When the temperature is very low, only the completions with highest harmony are given non-negligible estimated probabilities.

Although Smolensky briefly describes a learning algorithm for the Harmony Machine, much of his work (Smolensky, 1986) has been in gaining analytical and empirical insights into the behaviour of the machine, given a fixed set of link weights, during a single (annealing) pattern completion.

3.7 DETERMINISTIC MODEL

3.7.1 Multi-layer Perceptron

Multi-layer Perceptrons are feed-forward networks that possess a layer of input units, one or more layers of hidden units, and a layer of output units. These layers

of hidden units are not directly connected to both the input and output nodes. The hidden units in any given layer are not connected to each other. The hidden units and output units have real-valued outputs between 0 and 1 determined by the sigmoid function of net input.

Multi-layer Perceptrons overcome many of the difficulties of the Single-layer Perceptrons, but were generally not used in the past because effective training algorithms were not available. This has recently changed with the development of a new training algorithm called *back-propagation* (Rumelhart, Hinton & Williams, 1986). Although it cannot be proven that this algorithm converges, it has been shown to be successful for many problems (Rumelhart et al., 1986).

To use the network to implement an associative memory, given a set of input vectors and corresponding output vectors, the network must learn to produce the appropriate output vector for a given input vector. The Multi-layer Perceptron is operated by clamping an input vector into the input units and applying the decision rule to each layer of units in turn from the first hidden layer to the output layer. Since the desired output vector is known, each output unit can compute an error derivative based on the difference between its actual state and the state specified by the output vector. Since the state of an output unit is completely determined by its input (these units are not stochastic), and the sigmoid function is differentiable, it is easy to compute the derivative of the error with respect to the unit's total input. That quantity can be propagated backwards along the links into the unit (and scaled by the link weights), to produce error derivatives for the units in the last layer of hidden units. This process can be repeated recursively to produce error derivatives for each of the layers in turn back to the first layer of hidden units. The link weights are incrementally adjusted in proportion to the partial derivatives of the global error with respect to each weight in the network.

The back-propagation algorithm is a generalisation of the LMS or Widrow-Hoff algorithm (Widrow & Hoff, 1960; Widrow & Stearns, 1985). It uses a gradient search to minimise a cost function equal to the mean square difference between the desired and the actual net outputs. The desired output of all units is typically

low (0 or < 0.1) unless the unit corresponds to the class the current input is from. In that case it is *high* (1.0 or > 0.9). The net is trained with supervision. Weights and node offsets are initially set to small random values and all training data is then presented repeatedly. Weights are adjusted after every trial until the cost function is reduced to an acceptable value or remains unchanged. An essential component of the algorithm is the iterative method that propagates error terms back from units in the output layer to units in lower layers.

The hidden units in any given layer communicate with each other via their effects on the next layer. The error derivative at a unit depends both on downstream influences and on that unit's activity. A unit's activity, in turn, depends on the activities of all the units in the previous layer, and so the error derivative computed by one unit in the previous layer depends in part on the activities of all the other units in that layer. The formal derivation of the learning algorithm avoids dealing with these interactions by computing the appropriate change for a given weight under the assumption that all the other weights remain constant. Although steps can be taken to minimise this factor, in general, as the hidden unit interactions become more significant (for example when the number of hidden units in a layer is increased), a smaller learning rate must be chosen, which increases the learning time.

It is possible to define an *iterative* version of the back-propagation algorithm, which relaxes the feed forward restriction on the network topology. Instead of propagating errors back through the layers of feedforward network, errors can be propagated backwards through the *sequence of states* generated over time by an arbitrary network (Almeida, 1987a; Almeida, 1987b; Pineda, 1987).

Multi-layer Perceptrons which have been trained with back-propagation have been found to perform well in many applications including problems related to speech synthesis and recognition (Sejnowski & Rosenberg, 1986; Peeling & Moore, 1987; Elman & Zipser, 1987) and on problems related to visual pattern recognition (Rumelhart, Hinton & Williams, 1986). Back-propagation can be applied both to linear problems currently handled by Gaussian Classifiers and to

non-linear problems where it is difficult to specify the optimal classifier or filtering technique. The generally high performance found for the back-propagation algorithm is somewhat surprising, considering that it is a simple gradient search technique that may find local minima in the LMS cost function instead of the desired global minimum. The problem of multiple local minima corresponding to very different network performance levels has not been observed frequently for classification problems. One difficulty, however, has been that many presentations of the training data are frequently required for convergence. This is important during training but does not effect response time during classification. It currently sets a practical limit to the size of nets that can be trained but is not a severe problem for single-layer nets or multiple-layer nets with restricted connectivity. As an indication of the practical limit to the size of network and corpus of training data Elman & Zipser (1987) describe numerical simulations on a Cray XMP-4 for a network consisting of an input layer of 640 units, a hidden layer of 8 units, and an output layer of 640 units. The corpus used contained approximately 140,000 different input patterns, with each pattern consisting of 640 numbers. The network was trained on this corpus for 1,000,000 learning cycles. No indication of the program times was given.

3.7.2 Relationship to Hopfield and Boltzmann Machines

If the interconnection weights of a Multi-layer Perceptron are made symmetrical and the *self-feedback* weights w_{ji} forced to be zero and no hidden units are allowed, feedback perceptrons become formally equivalent to Hopfield networks with graded neurons (Hopfield, 1984). Therefore, back-propagation can be viewed as a learning rule for graded Hopfield networks.

In addition, a feedback perceptron with symmetrical weights and diagonal elements w_{ji} that are all zero and steep sigmoids, has approximately the same energy minima as a Boltzmann Machine with the same weights. Almeida (1987a) suggests that back-propagation can very probably be used to train Boltzmann Machines, i.e. to adapt their weights in such a way that they have a minimum of energy at the desired location, for each input pattern.

3.8 SUMMARY

> *"All of this will lead to theories [of computation] which are much less rigidly of an all-or-none nature than past and present formal logic. They will be of a much less combinatorial, and much more analytical, character. In fact, there are numerous indications to make us believe that this new system of formal logic will move closer to another discipline which has been linked in the past with logic. This is thermodynamics, primarily in the form it was received from Boltzmann, and is that part of theoretical physics which comes nearest in some aspects to manipulating and measuring information." John Von Neumann, Collected Works Vol5, p304.*

It was the Hopfield (1982) model which stimulated the current surge of interest in the relationship between neural networks and statistical mechanics. Hinton et al (1984) applied simulated annealing to Hopfield networks and proved a learning algorithm which allowed hidden units to capture underlying features.

There are two key insights behind the Boltzmann Machine. The first is Hopfield's result that networks with symmetric weights (and some other restrictions; see Section 3.2) can be viewed as minimising a globally defined *energy function*; the second is that with an appropriate choice of stochastic decision rule, the equilibrium behaviour of the network will be described by the Boltzmann distribution, providing a simple relationship between the energy of a state and its equilibrium probability. The Boltzmann Machine learning algorithm compares the statistics gathered at equilibrium when the visible units are externally driven and when the whole network is allowed to free-run, and adjusts the link weights to reduce the observed differences.

The features which make the Boltzmann Machine attractive for the application with which this work is concerned are that there are no restrictions on the network topology, that there is a formal proof for the convergence of the learning procedure and that the learning procedure is able to handle hidden units. Other attractions include the fact that the learning algorithm is based only on information available locally at the connection. This is particularly important for possible electronic implementations (Alspector & Allen, 1987).

4 SIMULATION ENVIRONMENT

4 SIMULATION ENVIRONMENT

4.1 INTRODUCTION

The simulation environment described in this chapter for the numerical simulation of Boltzmann Machines was developed from an initial serial implementation on a DEC VAX 11/750 to a pseudo-parallel implementation on a Masscomp[2] Vector Accelerator system. The Vector Accelerator implementation is described in this chapter and was in fact used for the majority of the experimental work described in the later chapters.

Figure 4.1 depicts a block diagram of the simulation environment. The crux of the system is the learning module, which along with the search module are the only parts of the simulation system that use the Vector Accelerator. The learning module is used for training Boltzmann Machines while the search module is used for testing Boltzmann Machines. The graphical weight display module allows the graphical presentation of connectivity information learned using the learn module. The error module allows the error measures obtained during learning to be smoothed, if required, and displayed graphically. The data preparation module provides various utilities for preparing the training and testing data for application to the Boltzmann Machine simulations.

In creating a simulation environment for Boltzmann Machines for the work of this thesis, it was important that flexibility be maintained without unduly compromising the speed of learning operation. The required flexibility was ensured by catering for the following factors in the simulator:-

1) networks of varying numbers of input, hidden and output units
2) the implementation of different connection strategies, e.g. fully connected, no connections between input and output units etc.
3) easy modification of the annealing and learning parameters
4) repeatability
5) ability to graphically display connectivity information

---

2 Masscomp is a trademark of the Massachusetts Computer Corporation.

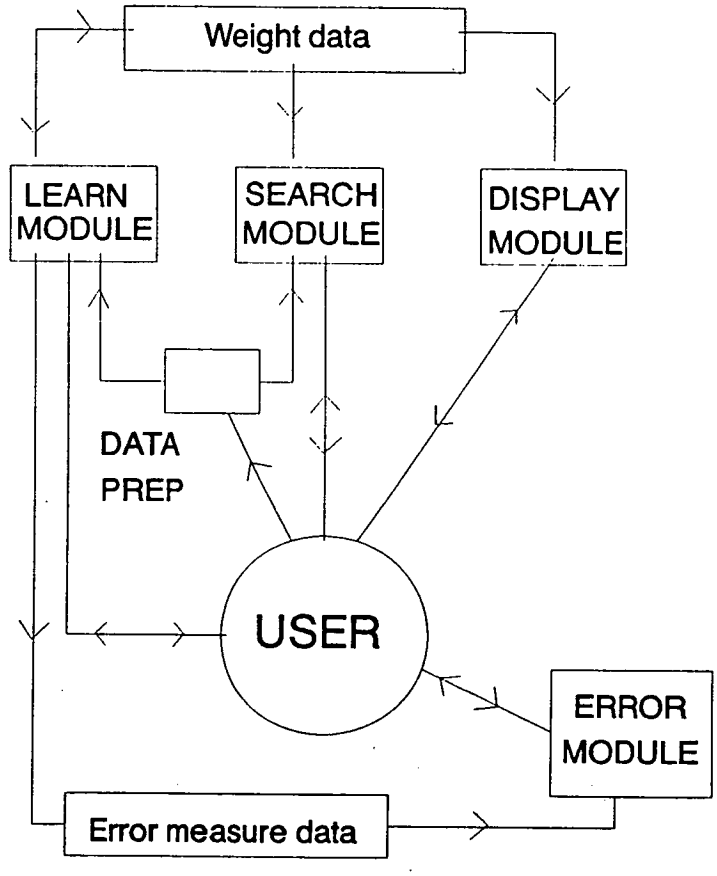

Figure 4.1 Block Diagram of Simulation System

Speed of learning operation was maximised by implementing the simulator on a Vector Accelerator system. This required that the units be stored as a vector and the connection weights as a matrix. Calculating energy difference values (which is the most time consuming aspect of simulation) was then easily performed using a dot product calculation. This approach allows higher speeds of operation than could have been obtained by creating a data structure called a *unit* and then incurring performance penalties due to the time taken for pointer manipulation.

This chapter provides a brief overview of the hardware platform used to host the simulation environment, which because of its unique architecture, directly influenced the approach taken in developing the simulation software. The particular structure of the Boltzmann Machines adopted in this work and the Boltzmann Machine learning algorithm itself are then discussed. Following this, the methodology behind the software development is described along with details of the software implementation of the Boltzmann Machine learning algorithm. The performance of the learning simulator is also discussed. The search module and graphical weight display modules are then briefly described along with a section discussing the interpretation of weight maps. Finally, elementary examples of the learning simulator's operation are given for the classic exclusive-OR (XOR) learning problem and the 4-2-4 encoder problem.

4.2 VECTOR ACCELERATOR OVERVIEW

The (final) hardware platform used for the Boltzmann Machine simulations was a Masscomp MC5700[3] minicomputer. This was configured with four central processor units (CPUs), 12 Mbytes of memory, one Vector Accelerator and one lightning floating point board. In addition, each of the four processors was equipped with a 68881 floating point processor. The operating system installed was Masscomp RTU[4] (Real-Time Unix). Also installed was Network File System (NFS) which allowed access to a very large file system. A single high-resolution

---

3 MC5700 is a trademark of the Massachusetts Computer Corporation

4 RTU is a trademark of the Massachusetts Computer Corporation

graphics terminal was also available on this powerful multi-user machine. Despite being essentially a parallel machine with four CPUs the key element here as regards the simulators is the Vector Accelerator. Figure 4.2 depicts a block diagram of the Vector Accelerator and Host. For clarity, only one Host CPU is shown.

The Masscomp Vector Accelerator is architecturally distinct from traditional array processor designs in two important ways. It is tightly coupled with the host, and the Vector Accelerator's local data memory is very fast. Many traditional array processors are independently programmed, and have large main and secondary storage so that they are almost self-sufficient. These machines are usually attached to the host over a lower-speed peripheral bus. The slow data transfer speeds over the bus create the need for a large memory in the array processor. In the absence of a large memory, the efficiency of the processor would be reduced by the slowness of data transfer to and from it.

The Masscomp Vector Accelerator does not have local program memory. Programs are not written to run on the Vector Accelerator's memory, but written to execute on the host and use the Vector Accelerator as a dedicated resource. Because data between the host and Vector Accelerator memory is transferred using the Vector Accelerator's direct memory access (DMA) facility, blocks of data are transferred at high speeds (12.47Mbytes/second for vector memory load operations and 15.30 Mbytes/second for store to host operations). An application program directly executes the instructions that transfer data between the Vector Accelerator and program address space. By contrast, many traditional array processors are virtually distinct computers, having incidental interactions with the host processor.

The Vector Accelerator is controlled by the host and dedicated to a process. In other words it may only be assigned to one process at a time. It communicates with the host via a Memory Interconnect (MI) bus (see Figure 4.2). The Vector Accelerator includes:-

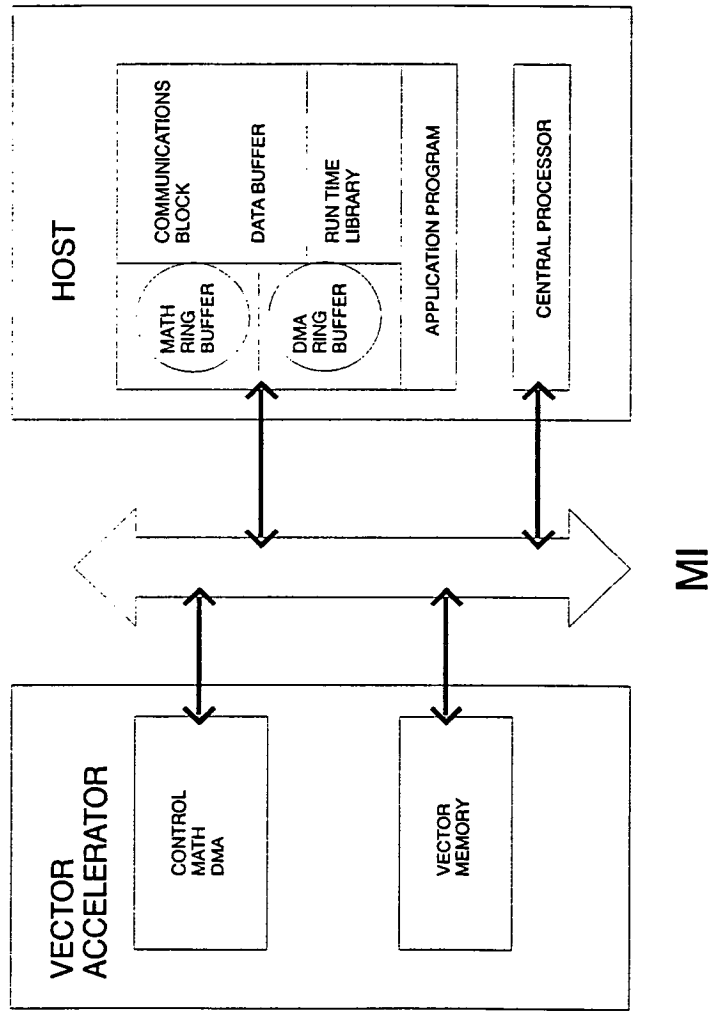

Figure 4.2 Vector Accelerator and Host Block Diagram

1) Control processor
2) DMA processor
3) MATH processor
4) Vector memory

The control processor has two functions. It mediates interaction between the host and the Vector Accelerator and also controls and schedules the MATH and DMA Processors.

The DMA processor (with some assistance from the Control Processor) handles direct memory access. It loads data from Host memory into Vector Memory; likewise, it stores data from Vector memory into Host memory. Although the Vector Accelerator operates on floating point numbers data can be transferred in either integer or floating point format.

The MATH Processor handles arithmetic processing by performing floating point operations on data in Vector Memory. If the data has been loaded as integer data, it must be converted to floating point data prior to issuing a MATH instruction. This can be done by the MATH processor under explicit program control. Except for a few exceptions the MATH processor adheres to the IEEE floating point standard.

The addressable Vector Memory consists of 32K (32768) 32-bit locations. However, locations 31742 - 32767 are reserved areas for constants and a MATH scratch pad, leaving locations 0 - 31741 available to the user.

The user application program, the Vector Accelerator driver and Run Time Library (RTL) all run on the host processor. The Vector Accelerator driver is that part of the operating system that deals exclusively with the Vector Accelerator. Functions of the Vector Accelerator driver fall into the following major categories:-

1) Accelerator allocating (for exclusive use) and de-allocating for the user process.
2) Initialising the Vector Accelerator.

3) Mapping Vector Accelerator memory, Communications Block and Ring Buffers into user virtual address space.
4) Handling interrupts generated by the Vector Accelerator
5) Establishing and delivering asynchronous system traps for the user process.

The Vector Accelerator DMA processor deals only with Vector memory and Host physical memory. Data structures and buffers declared in the application program are allocated to the data segment of the user process. If this data segment is not locked into host physical memory, it may not be present when needed for Vector Accelerator load and store operations. Thus, the data segment of the user program is usually locked into physical memory during Vector Accelerator allocation.

The Vector Accelerator Run Time Library (RTL) is a set of functions callable from C and FORTRAN. The RTL routines normally interact with other parts of the system through packet loading operations or through host operations. The packet loading routines place one or more packets (of instructions) in the ring buffers to instruct the Vector Accelerator to perform specific functions. The host routines operate exclusively on the host and do not cause the Vector Accelerator to perform an operation. Functions performed by the host routines include re-arranging vectors in host memory, establishing error handling and waiting for selected ring buffers to empty.

There are three basic steps that have to be undertaken in using the Vector Accelerator:-

1) load all required data before starting the MATH operation
2) compute the results before starting the store operation
3) store all required results before starting a host operation on those results

As each RTL routine is called in the application program, it places one or more instruction packets in either the MATH ring buffer or the DMA ring buffer, or both.

Synchronisation and wait routines in the RTL are used to control these operations. The synchronisation routines place a command packet into one of the ring buffers to control Vector Accelerator activity. When one of these packets is encountered, the Vector Accelerator suspends further processing of command packets in that ring buffer until all operations up to and including a specified command packet in the other ring buffer have completed. The wait routines block host application program activity until a synchronisation event from the Vector Accelerator occurs.

The application program must explicitly place each vector in Vector Memory, and must explicitly reserve enough space for each one. A start-up overhead is associated with Vector Accelerator MATH and DMA functions. This is the amount of time it takes the Vector Accelerator to decode a DMA or MATH command packet and start the actual transfer or operation. As the vector length grows, the relative importance of start-up overhead diminishes.

To get the maximum performance from using the Vector Accelerator it is important to minimise data transfers between the host memory and Vector Accelerator memory and to keep the vector lengths as long as possible to reduce the effect of start-up overheads. In addition, many MATH processor operations on the Vector Accelerator are optimised for a stride of one. Hence vectors need to be structured to have a stride of one in Vector Accelerator Memory.

Using the Vector Accelerator effectively requires the careful synchronisation of several independently executing processors:- the host CPU, the DMA processor, and the MATH processor. The design of the system allows the overlap of host CPU, DMA and MATH operations for optimum performance.

4.3 BOLTZMANN MACHINE LEARNING ALGORITHM

4.3.1 Network Structure

As discussed in Section 3.3.4 the units of a Boltzmann Machine can be divided into two groups of units, viz. a non-empty set of visible units and a possibly empty set of hidden units. The Boltzmann Machine learning algorithm can be used as

a content addressable memory if some of the visible units are forced into particular states representing a *search* key. The remaining visible units adopt states that are compatible with the key as determined by the training distribution. Alternatively, the visible units can be divided into an input set and an output set and the network asked to learn a collection of probability distributions, conditional on the states of the input units. The first formulation may therefore be considered as a process for completing a pattern which may be substantially incomplete and the second formulation may be considered as a process for classifying a pattern.

Thus the input/output model of the Boltzmann Machine appears to be more appropriate to pathological/healthy speaker discrimination and pathology discrimination than models consisting of isomorphic visible units, and hidden units. In addition the input/output model allows a more practical implementation of the learning algorithm for large machines since there is no requirement for the machine to learn the structure between the input units as they will always be clamped. Hence only the input/output model of the Boltzmann Machine was implemented in the numerical simulator. Figure 4.3 shows a topological representation of a fully interconnected input/output model of the Boltzmann Machine.

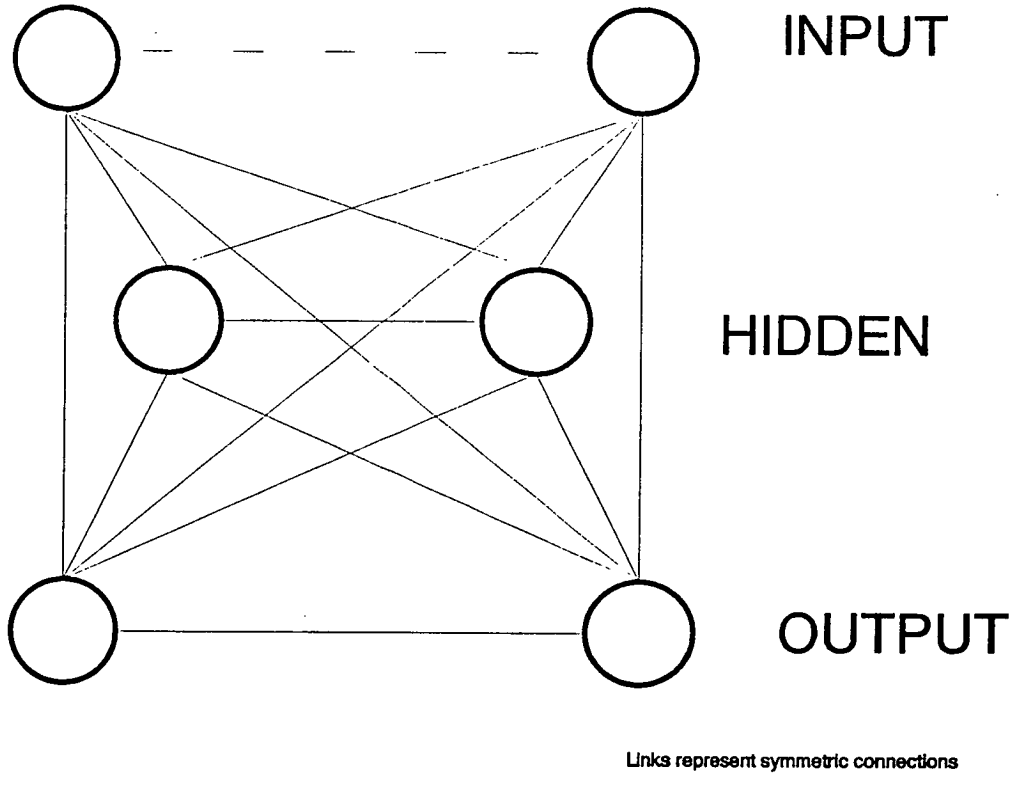

Figure 4.3 Structure of a Fully Interconnected Input /Output Model of the Boltzmann Machine

4.3.2 Learning Overview

Each Boltzmann Machine learning cycle has essentially three steps for training a model with exemplars consisting of input patterns and the corresponding desired output pattern:-

1) Clamp input and output units. *(Clamp phase)*

   p_{ij}^{+} is estimated by clamping the input units and output units with the exemplars. For each set of input and output patterns the machine is annealed and statistics gathered for each connection weight indicating how often the units at either end of the link are *on* together.

2) Clamp input units only. *(Free phase)*

   p_{ij}^{-} is estimated in a similar scheme to the above. However in this case the output units are unclamped and are free to change state. This step is referred to as the *free* phase of the learning procedure. It should be noted that for the input/output model of the Boltzmann Machine this means that only the output units are free to change state. In the visible/hidden unit model all the visible units are allowed to change state in the free phase.

3) Adapt the weight values.

   The sign of the partial derivative of G is found for all the connection weights and the weight values are then incremented if the sign is positive or decremented if the sign is negative by a constant value.

4.3.3 Unit State Update

Reference is made in subsequent sections to *units of time* for the annealing schedule and collection of co-occurrence statistics. In the Boltzmann Machine context one unit of time is defined as the time required for each unit to be given, on average, one opportunity to change its state.

Boltzmann Machine theory dictates that each unit be updated asynchronously. However, it was found empirically that satisfactory operation could be obtained for suitably large machines if the units were in fact updated synchronously. Synchronous operation, where the units change state at the same time

depending on the previous set of states, allows efficient use of the Vector Accelerator and consequently allows a higher speed of simulation than could be achieved by asynchronous operation with the Accelerator.

The Boltzmann Machine employs stochastic binary units based on a sigmoid function given by:-

$$P(s_i = 1) = \frac{1}{1 + e^{-\Delta E_i / T}} \qquad (4.1)$$

Where ΔE_i is the difference in energy for the *on* and *off* states of the i^{th} unit. A scaling constant T, the *temperature*, determines how smooth the curve is. In the limit $T \rightarrow 0$, the Boltzmann Machine decision rule becomes the deterministic binary rule; as $T \rightarrow \infty$, the decision rule becomes a completely random choice. A cumulative Gaussian distribution is a sigmoid function that fits very well to the probability function curve given by Equation (4.1).

The amplitude of the Gaussian noise determines the width of the sigmoidal probability distribution. If the standard deviation of the Gaussian distribution is chosen suitably the two curves have the same gradient at $\Delta E = 0$ and the curves never differ by more than about 1%, (see Figure 4.4).

Using Equation (4.1) the state of a unit may be determined by first computing the probability that the unit should be in the *on* state. To determine whether to set the unit's state to 1 or 0 a random variable uniformly distributed between 0 and 1 is consulted. If this random number is less than the probability of the unit being in the *on* state the unit adopts the *on* state, otherwise it is set to the *off* state.

However, using the Gaussian approximation the local decision rule becomes:-

$$\text{If} \quad \Delta E + N(0, \sigma) > 0 \quad \text{then set} \quad s_i = 1 \quad \text{else} \quad s_i = 0 \qquad (4.2)$$

where $N(0, \sigma)$ is a sample from a Gaussian distribution of mean 0 and standard deviation σ. The relationship between temperature and standard deviation to ensure that the gradient of the curves for the cumulative Gaussian distribution and the probability function at $\Delta E = 0$ are identical is:-

$$\sigma = \sqrt{\frac{8}{\pi}} T \qquad (4.3)$$

Figure 4.4 depicts the curves for the probability function and the cumulative Gaussian function with $T = 1.0$ and $\sigma = \sqrt{\frac{8}{\pi}}$

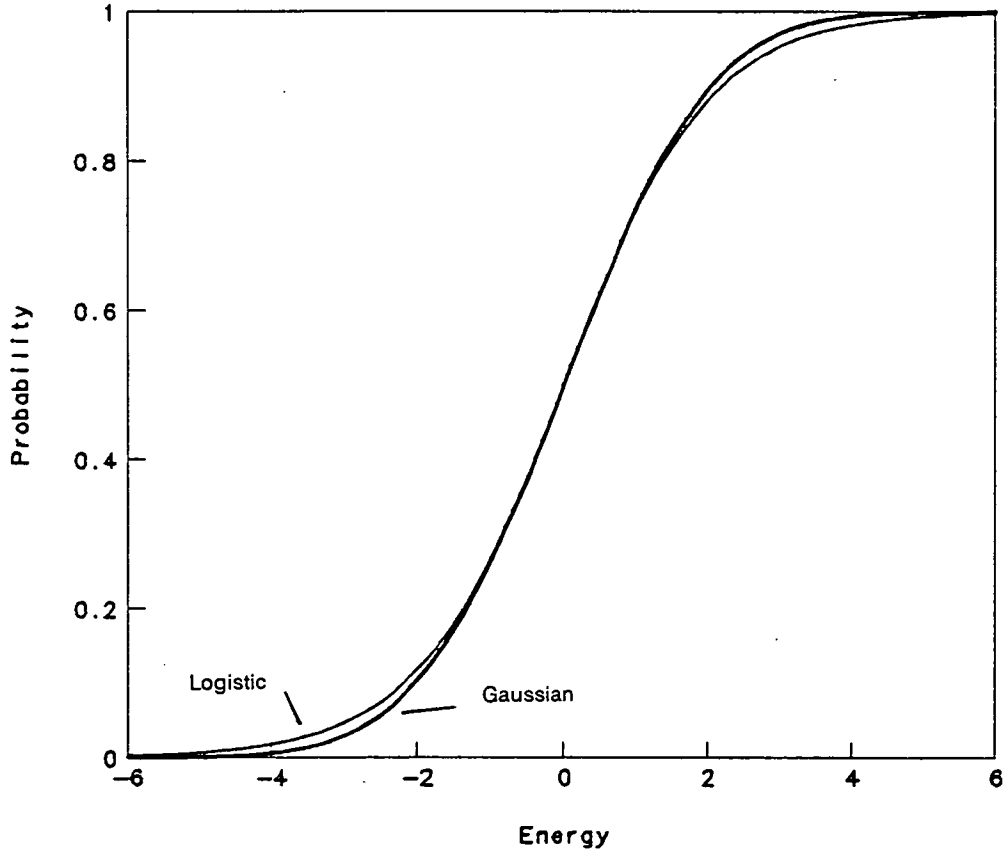

Figure 4.4 Superimposed curves for probability function and cumulative area under a Gaussian. For the probability function curve $T = 0$. For the cumulative Gaussian curve the Gaussian distribution used has a mean of 0 and standard deviation of $\sqrt{8/\pi}$.

4.3.4 Simulated Annealing

Geman & Geman (1984) were able to show that a system converges to the globally optimal state if temperature is lowered at the rate:-

$$T = \frac{C}{\log(n+1)} \tag{4.4}$$

where n is the number of iterations, and C is a constant. The value of C for which Geman & Geman were able to guarantee convergence is in general very high so that convergence becomes impracticably slow. However, it has been found experimentally that annealing schedules approximating the above can produce reasonable convergence behaviour, Hinton, Sejnowski & Ackley (1984).

The starting temperature can be selected using the mean of the absolute value of the energy difference ΔE that each unit has to overcome before being able to make an uphill jump. These upward moves occur when the magnitude of ΔE is within a range of about three times the temperature or about five times the deviation σ of the Gaussian noise. The starting value σ for the annealing schedule is thus chosen to be five times the average absolute energy. The schedule allows each unit to change once at each temperature, before moving to the next lower temperature. The advantage of using a starting temperature that depends on the energy barrier is that is ensures that the network will start at a sufficiently high temperature to overcome very high energy barriers created during the learning.

4.3.5 G Measure

As stated previously the Boltzmann Machine learning algorithm allows a network of binary symmetrically connected units to develop an internal model which captures the structure of its environment. An information theoretic measure, the G measure (Kullback, 1959; Renyi, 1962), of the discrepancy between the internal model and the environment can be made. For the input/output model of the Boltzmann Machine the G measure is:-

$$G = \sum_{ab} P^{+}(I_a \wedge O_b) \ln \frac{P^{+}(O_b \mid I_a)}{P^{-}(O_b \mid I_a)} \qquad (4.5)$$

where $P^{+}(I_a \wedge O_b)$ is the probability of the a^{th} state of the input units and the b^{th} state of the output units when all visible units are clamped by the environment, $P^{+}(O_b \mid I_a)$ is the probability of the b^{th} state of the output units given the a^{th} state of the input units when all visible units are clamped by the environment, $P^{-}(O_b \mid I_a)$ is the corresponding probability with only the input units clamped. The structure referred to in Equation 4.5 is depicted in Figure 4.3.

G is zero if and only if the clamped and free phase distributions are identical; otherwise it is positive. Learning is accomplished by reducing G and thus the G measure may be used to monitor how learning is proceeding during training.

4.3.6 Co-occurrence Statistics

Learning is accomplished by reducing G, which is achieved by modifying the link weights, since these determine $P^{-}(O_b \mid I_a)$. Because of the relationships that hold at thermal equilibrium with the relative probabilities of two global states following a Boltzmann distribution, the partial derivative of G with respect to each of the weights can be calculated:-

$$\frac{\delta G}{\delta w_{ij}} = -\frac{1}{T}(p_{ij}^{+} - p_{ij}^{-}) \qquad (4.6)$$

where p_{ij}^{+} is the probability of units *i* and *j* being on together when all the visible units are clamped and p_{ij}^{-} is the corresponding probability with only the input units clamped. Hence the performance of the machine in each of the clamping modes is compared and the link weights altered according to the findings. Thus for each mode the network is annealed to a low temperature for each set of training data and statistics are gathered for pairs of units being on together.

4.3.7 Weight Adaptation

The direction in which to change the link weights is known once the partial derivative of G (Equation 4.6) has been found, but the magnitude of the weight change required is not specified. The strategy adopted in this work was to use a constant step for all the links. Thus, rather than change w_{ij} by an amount proportional to $(p_{ij}^{+} - p_{ij}^{-})$, it is incremented by a fixed weight step if $p_{ij}^{+} > p_{ij}^{-}$ and decremented by the same amount if $p_{ij}^{+} < p_{ij}^{-}$. There are a number of ways of implementing the change in weight. The weight step can be fixed throughout the learning or the magnitude of the step may be decreased as a function of the mean error as the network converges:-

$$\Delta w_{ij} = (10\bar{\xi} + 1) \times SIGN(p_{ij}^{+} - p_{ij}^{-}) \qquad (4.7)$$

where $\bar{\xi}$ is the mean output bit error. This approach has the advantage in that it does not require the machine to select a representation that requires a very precise co-ordination between the numerical values of the weights. Problems can occur if some weights become very large as this tends to make it difficult for the network to achieve equilibrium due to the energy barriers becoming too large. To prevent the weights from becoming too large, the weights may be decayed by subtracting from each weight a fraction of its value Derthick (1984). This also ensures that arithmetic overflow does not occur and cause the learning process to abort.

4.4 SOFTWARE DEVELOPMENT METHODOLOGY

4.4.1 Introduction

The software development process for the various modules in the simulator followed the steps shown below:-

1) requirements analysis
2) design specification
3) implementation
4) unit test, integration and system test

The software tools used comprised solely of a 'C' language compiler used in conjunction with the *make* utility, a debugger, a performance profiler, and a special library of routines to allow synchronisation checking to be undertaken. Regrettably, no source code control system or computer aided software engineering (CASE) tools to assist in the requirements analysis and design specification phases were available for use.

4.4.2 Requirements Analysis

The first step in the software development process was to construct a requirements specification for the Boltzmann Machine simulation environment. One of the key requirements was that flexibility be maintained without unduly compromising the speed of the learning operation. To achieve the desired flexibility it was deemed necessary to accommodate the following capabilities in the simulator:-

1) networks of varying numbers of input, hidden and output units
2) the implementation of different connection strategies, e.g. fully connected, no connections between input and output units etc.
3) easy modification of the annealing and learning parameters
4) repeatability
5) ability to graphically display connectivity information

The hardware platforms available in the Research Group in which the Boltzmann Machine studies were carried out included a DEC VAX 11/750, two Masscomp MC5500[5] machines with Array Processors and latterly one Masscomp MC5700 with Vector Accelerator. The use of the Masscomp Array Processor and Vector

---

5 MC5500 is a trademark of the Massachusetts Computer Corporation

Accelerator systems offered the best possible performance solutions, but required that the algorithms be structured for the unique architectures of these machines. The operating system available on these machines was Masscomp RTU (Real-time Unix).

Because the simulation environment was designed to be a research tool it was important that it be easy to modify sections of code to incorporate further features or abilities as required. Thus it was important that the code generated be easy to understand and well documented. The overall structure of the simulation system arrived at is depicted in Figure 4.1. Each of the learn, search, graphical weight display and error modules was implemented as a standalone program or programs. These programs were able to share certain common data files, containing the weight information and error measures. Only the development of the learn and search program modules is detailed in the following sections.

As mentioned in Section 4.2 the Vector Accelerator (and Array Processor) is a single-user resource. In other words only one user process could use the device at a time. Because the learning process could take up a considerable amount of compute time it was necessary to ensure that the learning could be halted from time to time so that the Vector Accelerator could be freed to allow other users on the system to use it. Because there was no queueing system arrangement for the Accelerator, a scheme was required which would allow the simulator a given amount of Vector Accelerator time before making it available to another user. In addition, while the simulator was waiting to use the Vector Accelerator whilst another user's process was locked in, it had to be able to poll for when the Vector Accelerator was released and ready for use.

Because the simulations were to be run overnight it was important that a system crash did not result in the loss of data. To overcome this the weight information was to be stored periodically. It was not deemed feasible to store the weights developed for every learning cycle due to disc storage restrictions. The storage of the weights was also required to enable details of how the weights developed during learning to be observed. An alternative approach to the above for storing

the weight information would have been to store all the weights to file after each weight update, overwriting previous weight values in the file along with the current learning cycle detail. Another approach would have been to not store any weights to file until the learning task had been accomplished or a level of classification error had been achieved. Both of these approaches would not allow the development of the weights to be examined as learning progressed.

The user interface to the learn and search programs made extensive use of the command line to specify the operating modes. Since the majority of processing work had to be undertaken overnight in *batch* mode, and indeed the processing time for most simulations could be measured in hours or tens of hours, the provision of a graphical user interface for the environment was not deemed relevant.

Parameters passed to the learning simulator via the command line were as follows:-

1) random number generator seed
2) operating mode (synchronous/asynchronous)
3) annealing schedule (fixed schedule, fixed start temperature and log or linear decay, energy based start temperature and log or linear decay)
4) network topology (10 schemes, see Table 4.1)
5) pattern clamping method (random/dot)
6) weight adaptation (fixed, scaled, decayed)
7) name of file containing the network details (i.e. number of input, hidden and output units)
8) name of file containing various annealing, co-occurrence statistic gathering and weight increment parameters
9) name of file for control data, such as the number of learning cycles, where to restart learning and how often to save the weights to file
10) name of file containing the training data parameters and training data files
11) generic name for weight, error measure and diagnostic output files

Thus all parameters required in the simulator's operation were specified through the command-line, except that is for details of the length of Vector Accelerator time required and for how long the Accelerator was to be released. This information was passed to the simulator from a reference text file which was edited to change Vector Accelerator time parameters rather than use a different parameter file.

Table 4.1 details the various network topologies that could be specified. It should be noted that no intra input unit connections existed because the input units were always clamped. The number of receptive fields could be varied from the usual one to accommodate either three (intonation/jitter/shimmer parameters) or ten fields (one field for each of the ten intonation and perturbation parameters used). To obtain these various topologies, the standard weight matrix was used. However at the end of each learning cycle, after all the weights (in the fully interconnected input/output model) had been adapted, the connection topology required was obtained by setting to zero all the weights for the connections that were not needed. Figure 4.5 indicates the general structural arrangement for networks with three receptive fields (RFTHREE1 type) whereas Figure 4.6 indicates the general structural arrangement for networks with ten receptive fields (RFTEN2 type). The scheme mnemonics are those devised by the author.

As discussed in Section 4.2 it is important when using the Vector Accelerator to minimise the number of data transfers between the host memory and Vector Accelerator memory and to keep the vector lengths as long as possible to reduce the effect of start-up overheads. It is therefore necessary that the Boltzmann Machine algorithms be implemented in a way that makes efficient use of the Vector Accelerator hardware.

Scheme	Receptive Fields	i/p to o/p connections	intra hidden connections	intra output connections
NOIO	1	N	Y	Y
ENCODER	1	N	N	Y
NOOP2OP	1	Y	Y	N
IDENT	1	N	Y	N
NOHID	1	Y	N	N
RFTHREE1	3	N	N	Y
RFTHREE2	3	N	N	N
RFTHREE3	3	N	Y	Y
RFTEN1	10	N	Y	Y
RFTEN2	10	N	N	Y

Table 4.1 Network Topologies

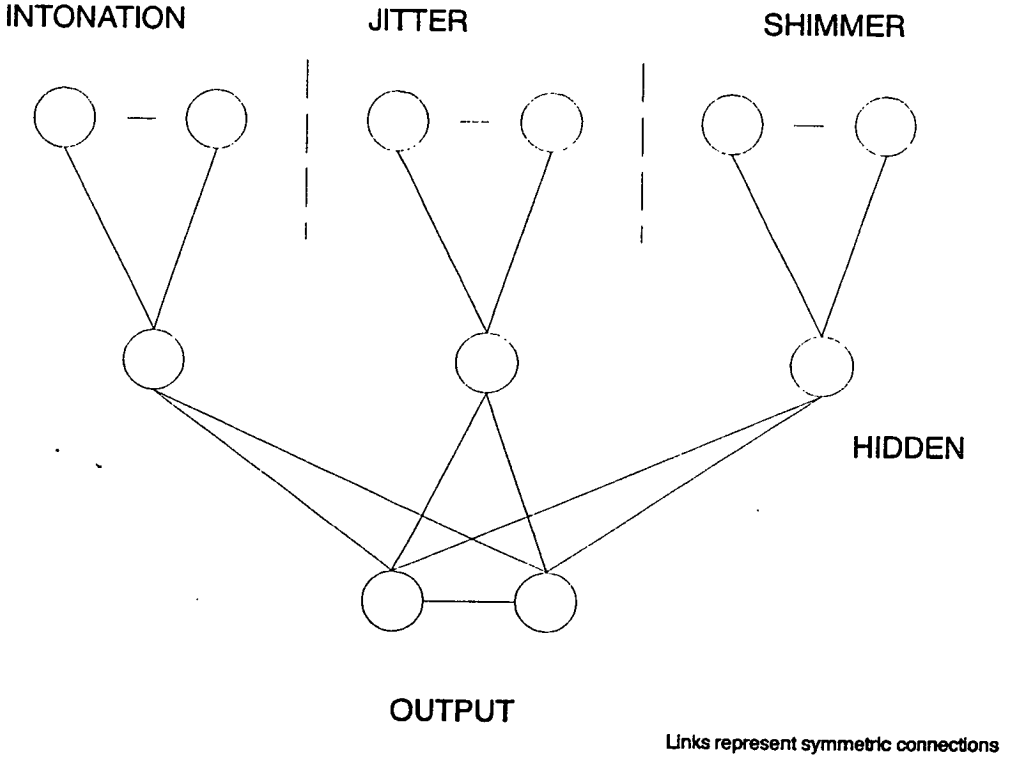

Figure 4.5 General Structure for Networks with Three Receptive Fields

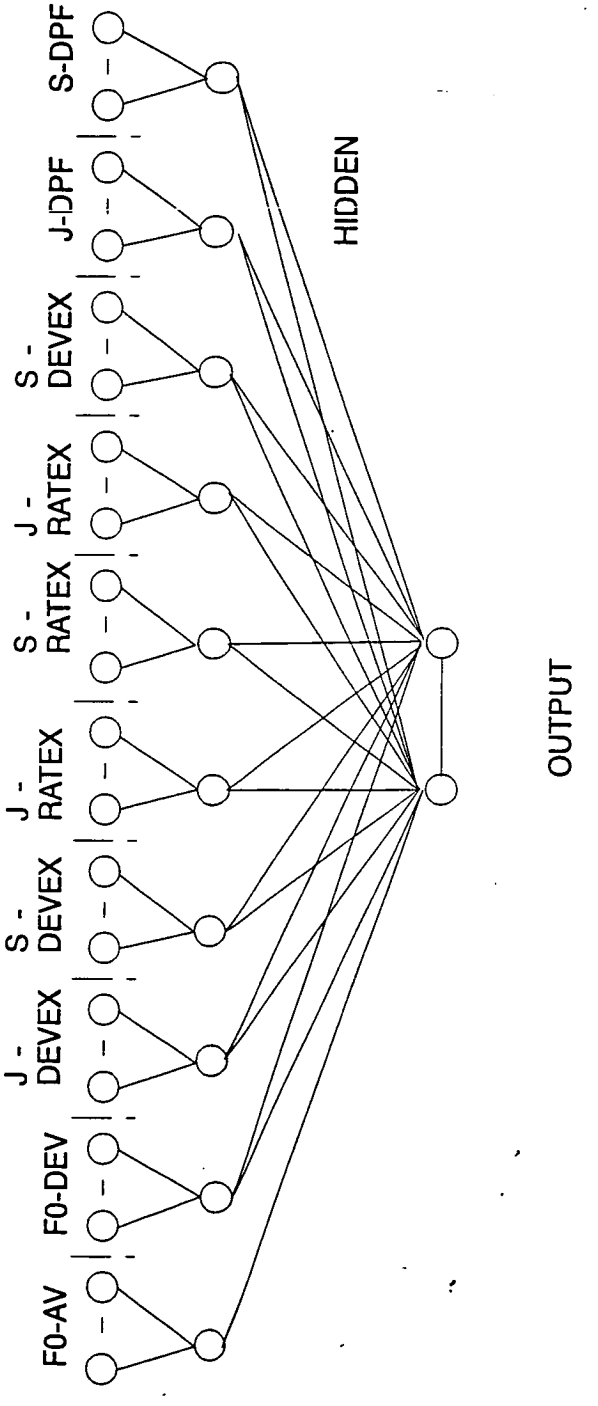

Figure 4.6 General Structure for Networks with Ten Receptive Fields

4.4.3 Design Specification

In this development phase the required functionality for the learn and search program modules was segregated into a set of twelve component modules as detailed in Table 4.2.

Module	Function	Lines	Approx. Man-weeks
Flmn.c	master learn module	475	20
Fsmn.c	master search module	289	8
Cmap.c	allocate and de-allocate VA	34	1
Csig.c	signal/interrupt handling	48	1
Fadp.c	adapt the weights	143	6
Fchst.c	change the unit states	63	4
Fclmp.c	binary clamping of patterns	56	1
Fegy.c	energy calculation	82	8
Fgms.c	error measures	45	1
Fiop.c	various file/data input/output routines	93	1
Fsts.c	collection of statistics	52	4
Frnd.c	randomise state of units	22	1

Table 4.2 Simulator Modules

In addition a small set of library routines was written to provide error handling and random number generation. Table 4.2 also provides an indication of the lines of code required for each of the component modules and the approximate programming effort exerted in achieving the final versions of the learn and search programs.

The highest level component modules are Flmn.c and Fsmn.c. These two modules call functions in the other (lower-level) modules. At this stage of the development process the external data file formats were also defined along with the key algorithms, data structures and data paths.

4.4.4 Implementation and Coding

Coding followed the software design blueprint developed in the design specification stages. As mentioned before all code was written in the 'C' programming language with numerous calls to the Vector Accelerator Run Time Library routines. The code was liberally commented throughout, and was at various stages reviewed and subjected to walk-throughs. It was not possible to obtain an independent input at these stages in attempting to locate bugs and errors.

4.4.5 Unit Test, Integration and System Test

Generally, each module was thoroughly tested using where necessary programs which supplied certain inputs and allowed the resulting outputs from the module to be validated. Once the individual modules had been tested, they were integrated into the final system. The system was then further tested with a number of cases designed to test the overall system operation. It was particularly important to make sure that testing was thoroughly carried out, as it was found at an early stage that learning can still occur in the simulator despite errors in the implementation. This meant that it was not sufficient to test the simulator only by checking that it can learn some elementary function such as the exclusive-OR or the 4-2-4 encoder problem.

The software development process cycled round the various development phases until the system had been fully integrated and debugged. The next stage was to undertake performance profiling of the system and identify which areas were causing performance bottlenecks, and which areas would produce the

largest benefits from revised approaches. Thus, the development cycle around the development phases was repeated again but with the inclusion of performance profiling.

4.4.6 Porting Array Processor Software to the Vector Accelerator

For the vectorised implementation of the Boltzmann Machine simulator only Masscomp MC5500s with Array Processors were initially available as the hardware environment. Hence initial design and development work was undertaken using the Array processor. Later when a Masscomp MC5700 with Vector Accelerator became available for use, the software was ported to this and enhanced. The Vector Accelerator system differed from the Array Processor system by having twice as much vector memory and a performance increase of approximately one order of magnitude for the Boltzmann Machine learning application. It also allowed the use of multiple Vector Accelerators, whereas the MC5500 systems only allowed a single Array Processor. However only one Vector Accelerator was in fact installed on the Masscomp MC5700.

The porting of Array Processor applications to the Vector Accelerator was fairly straightforward. Two methods were available and were determined by which of two libraries the application program was linked to. One library allowed programs to be run in VA mode while the other allowed programs to be run in AP mode. The libraries differ in how the Ring Buffer data structures and data pointers are used. An application running in the AP mode could only use one Vector Accelerator, and would use only one Ring Buffer, the MATH Ring Buffer. Since the packets execute sequentially, there is no overlap of DMA and MATH operations, and the need for DMA/MATH synchronisation is eliminated. Applications programs running in VA mode can be written to take full advantage of the dual Ring Buffers and other Vector Accelerator hardware.

The porting of the Array Processor routines to the Vector accelerator required that initially they be checked for correct synchronisation. This was achieved using a special synchronisation library to identify any problems. It is quite possible for an Array Processor application to have synchronisation errors that are never detected on that system. This is because the faulty instruction sequences, while not containing proper synchronisation requests may nevertheless preserve a correct order of processing due to accidents of the instruction set-up time, vector lengths or other incidental factors. However, porting such a program to a Vector Accelerator may bring out such hidden problems.

After checking for synchronisation problems the programs were compiled and linked with the AP mode library. The programs were then checked to see that they executed correctly and yielded the correct results, and that they did so under different system conditions. Some program synchronisation errors may or may not reveal themselves on any given execution, depending on system load and other variable factors. Thus it was basically an iterative process to correct synchronisation problems, compile and check execution of the program. Once the program was running correctly in the AP mode it was then recompiled using the VA mode library. Again correct operation was checked, and if necessary iterating as before in the event of synchronisation problems arising. Once the program was running correctly in the VA mode it was then possible to improve the performance further by modifying the code to incorporate new Vector Accelerator Run Time Library routines and longer vector lengths.

4.5 SOFTWARE IMPLEMENTATION FOR LEARNING

4.5.1 Data Structures

As outlined earlier, to achieve efficient operation of the Vector Accelerator requires that the Boltzmann Machine algorithms be structured so as to minimise the transfer of data between the Accelerator and the host. It was thus important to keep as much of the computation and data in the Vector Accelerator as possible. In addition, it was also important to make the vectors being operated

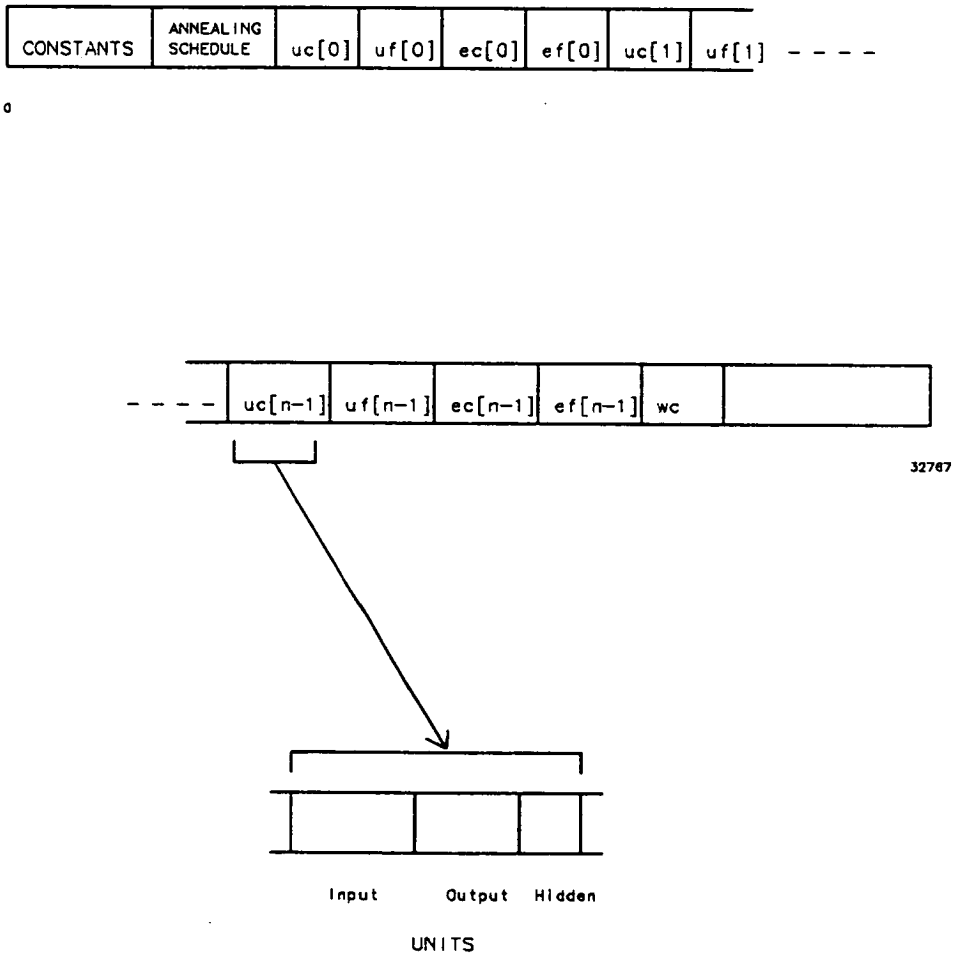

Figure 4.7 Vector accelerator memory allocation.

on by the Accelerator as long as possible. This was achieved by accommodating within Vector Accelerator memory as many pairs of Boltzmann Machines as possible. One of the machines of each pair had its input and output units clamped by one of the sets of training data, while the remaining machine of the pair had only its input units clamped with data from this set. Thus, the clamped and free phases of the learning algorithm co-exist in the simulator for as many of the training patterns as the Accelerator's memory will allow. Unfortunately, due to the nature of the computations required it was not generally possible to allow simultaneous Vector Accelerator operations over all the Boltzmann Machines in Accelerator memory. However, a limited number of operations could be performed on all the machines at once.

Figure 4.7 depicts how the Vector Accelerator memory was allocated. Locations 0 to 5 were used to store constants as shown in Table 4.3.

Location	Constant	Description
0	zero	0.0
1	one	1.0
2	+prm.wsc	constant weight increment
3	-prm.wsc	constant weight decrement
4	prm.pon	probability that an *on* unit is allowed to remain *on*
5	prm.pof	probability that an *off* unit is allowed to remain *off*

Table 4.3 Locations 0 - 5 of Vector Accelerator Memory

The position of the unit and energy vectors in the Accelerator's memory depends on:-

1) The length of the annealing schedule
2) The number of units in each network

3) The number of training pattern sets that could be applied to pairs of machines

Figure 4.7 depicts the general position of these vectors. Immediately following the constant area in Accelerator memory is the annealing schedule. The remaining Vector Accelerator memory was divided up into vectors corresponding to clamped units (uc); free units (uf); energy values for the clamped units (ec); energy values for the free units (ef) and a working area (wc). As many patterns as possible were installed into Vector Accelerator memory so that the units and energy portions of the Accelerator memory were indexed as uc[n], uf[n], ec[n], ef[n] where n is the training pattern number. The number of units in a single network is referred to as *snet,* and the number of patterns being processed in the vector Accelerator is referred to as *patns.*

Although it would be possible to shuffle various vectors around in Vector Accelerator memory and increase vector lengths, is was considered that the performance increase would not justify the time taken in shuffling the vectors, and also that this would result in code that was difficult to understand and modify.

The connectivity weight matrix for a network of N units is a symmetric zero diagonal matrix of size N × N. The zero diagonal indicates that there is no connection from a unit to itself, i.e. $w_{ii} = 0$. Also associated with each unit is a bias or threshold value. For a network of 100 units the weight matrix size is 100 × 100 which with 32-bit floating point number representation requires approximately 39 Kbytes of memory. The 100 bias values take up another 400 bytes. For a net with 200 units the storage requirements would be about 157 Kbytes. As it is required to store the weight matrices at regular intervals in order to observe the way the weights develop during training, it can be seen that a large amount of storage would be required. This storage requirement can be relaxed because the weight matrix is symmetric, it is thus only necessary to store $(N^2 + N)/2$ values (including the bias values).

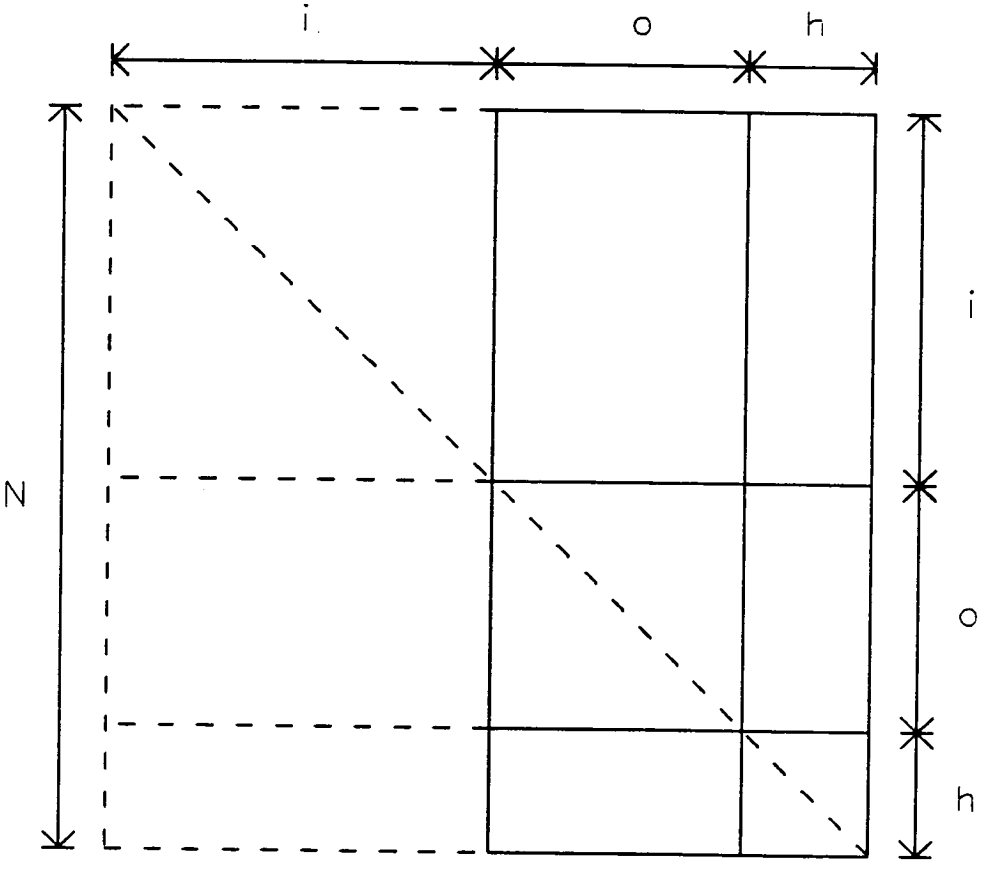

Figure 4.8 Weight matrix format.

However, the particular formulation of the Boltzmann Machine under study was the input/output model. Here the input units remain clamped throughout the learning procedure as the machine is not required to model any structure between the input units. Hence, the weight values between the input units are all zero. Thus if we have *i* input units, *o* output units and *h* hidden units, all the weights can be determined from a matrix of size $(o + h) \times (i + h + o)$, (see Figure 4.8). This is particularly significant if there are a large number of input units.

Further savings in storage space could be made by using the symmetry property as outlined earlier. However, for a large number of simulations the number of input units is likely to be larger than the number of hidden and output units. There would also be time penalties incurred if the weights were stored in a compacted form. Thus the matrix form shown in Figure 4.8 was adopted for storing the weight values. This was slightly modified in that the bias values were stored in the normally zero diagonal locations of the weight matrix.

4.5.2 Main Component Module

The component module Flmn.c handles getting the operating parameters from the command line, opening/closing files, calculating pointers, allocating storage, generating vectors of both Gaussian distributed and uniformly distributed random numbers, checking the position of learning and retrieving weights from a file if it was a continuation of an aborted learning process, obtaining control of the Vector Accelerator and locking it in to process memory, and initialising the Vector Accelerator.

As part of the Vector Accelerator sharing scheme it was necessary to specify how much Accelerator time was required before releasing the Accelerator to allow another user to have access to it. A give period of time was used rather than a given number of learning cycles since the learning cycle time varied depending on the size of the network being simulated and the quantity of training data that was being presented. Vector Accelerator use was timed using an *alarm* signal which generates an interrupt after a given time period and enables the Vector

Accelerator to be released. After waiting for a given period of time attempts were then made to gain control of the Accelerator again. This process was continued until the specified number of learning cycles had been achieved. Arithmetic exceptions (invalid operand, underflow, overflow and inexact results) in the Vector Accelerator made use of the default RTL response, which caused an error message to be printed and the process to be aborted.

For training data consisting of *n* sets of input and corresponding output patterns, 2*n* Boltzmann Machines were established in Vector Accelerator memory. Thus each of the two phases detailed above coexisted in the simulator. If the number of pairs of machines was too large for the Vector Accelerator memory the training patterns were broken up into subsets corresponding to the maximum number of pairs of machines that could be fitted into memory. These subsets were then presented sequentially to the virtual Boltzmann Machines residing in memory on completion of the learning cycle for the previous pattern subset. In this way transfers from host memory to Vector Accelerator memory were minimised, and vector lengths were kept as long as possible.

Learning was started by initialising to zero the variable handling the error measure value and installing the annealing schedule and constants into Vector Accelerator memory. The sequence of learning operations was then as follows:-

1) clear co-occurrence statistics matrices
2) randomly set the state of all the units
3) clamp input and output units in *clamp phase* machines and input units in *free phase* machines
4) anneal the machines
5) collect the co-occurrence statistics
6) adapt the weights

Repeat 1 - 6 until the specified number of learning cycles have been completed.

The method of implementing the Boltzmann Machines in the simulator means that it is not practical to test each Boltzmann Machine to ascertain whether or not

it has converged. Instead, the specified annealing schedule is followed by all the Boltzmann Machines for the same number of steps and convergence is assumed to have occurred.

4.5.3 Random numbers

Simulations require two types of distribution of random numbers:-

1) numbers uniformly distributed in the range 0 to 1 and
2) numbers drawn from a Gaussian distribution of mean 0 and standard deviation 1.

Because the generation of both these random numbers takes up considerable time, they were not calculated *on demand* as the learning progressed. Instead, a vector of 10,000 numbers was generated for each type of random number at the start of learning. During the learning, vectors of numbers were taken from random starting positions in the host vectors and transferred to the Vector Accelerator.

Masscomp RTU includes the routine rand() which provides random integers uniformly distributed in the range 0 - 32767. This was used to provide random numbers uniformly distributed in the range 0 to 1 by dividing the value returned from rand() by 32767. The method used for generating the normal (Gaussian) deviates was the Polar method (Knuth, 1969). This algorithm calculates two independent normally distributed variables given two independent variables uniformly distributed between 0 and 1.

4.5.4 Initialisation

A function rnd_u(patns) is used to randomly set the state of all the units so that there is a 50% probability of them being in the *on* state. This is achieved by loading *snet* (the total number of units in each network) normal (Gaussian) distributed numbers into the Vector Accelerator at locations uc[n] and uf[n] for n = 0 to n = patns. Thus the complete units vector area for all of the Boltzmann Machines in Vector Accelerator memory are filled with these standard deviates. The

deviates are compared to 0 and if an element is greater than 0, that element in the vector is set to 1.0 otherwise it is set to 0. The vector length for this operation is $2 \times snet \times patns$.

The component module Fclmp.c contains a number of clamping routines. Some of these are specific to the format of the data applied to the input units and are described in Chapter 6. However, one of the functions, clmp_rnd() is described here. This function sets *on* units *off* with probability *(1 - prm.pon)*, and sets *off* units *on* with probability *(1 - prm.pof).* It is used to introduce noisy clamping (if *prm.pon* and *prm.pof* are less than 1) and is a technique for preventing the weights from becoming too large. For each Boltzmann Machine in Vector Accelerator memory the function first loads the input and output patterns for the clamped units. Then the Vector Accelerator is loaded with a vector of uniformly distributed random numbers corresponding to the number of input and output units. These values are then compared so that if greater than *prm.pon* the corresponding element in *ec[0]* (used here for temporary storage) is set to zero, otherwise it is set to 1. This is repeated again but this time for whether the random values are greater than *prm.pof.* In which case the corresponding element in *ef[0]* (used here for temporary storage) is set to 1 otherwise it is set to 0. Then to actually perform the noisy clamping the states of the units are compared to 1. If they are equal to 1 they are replaced with the corresponding element from vector *ec[0]*, if not, they are replaced with the corresponding element from vector *ef[0]*. The uc[n] input pattern is then copied to the input pattern portion of *uf[n]*. This process is repeated for all of the *n* Boltzmann Machines in Vector Accelerator memory.

4.5.5 Energy Calculation

The function energy(patns) is used to determine the energy difference values for all the units in Vector Accelerator memory except that is for the input units. This is achieved by calculating the dot-product of the unit states and corresponding weights. Thus, for each unit (except the input units) the column vector of weights representing the weight values between it and every other unit in the Boltzmann Machine are loaded into the Vector Accelerator memory. This vector is multiplied

with the corresponding states of the units. The result is then summed and stored at the appropriate index in the relevant energy vector. This calculation is performed for every pair of Boltzmann Machines in Vector Accelerator memory, so that the weight column vector is only loaded into Accelerator memory once every learning cycle.

On completion of the dot-product calculations for all relevant units in all the Boltzmann Machines an adjustment has to be made to ensure that the bias value for each unit is correctly added to its energy value. At this stage, the bias value will only have been added if the unit in consideration is in the *on* state. However the bias value must be added irrespective of the state of the unit. To achieve this in Accelerator memory, the bias values are loaded from the host to Accelerator memory. The stride for the bias values on the host is effectively *snet+1,* and is transformed to a stride of 1 in the Accelerator. These bias values are then added to the corresponding energy values. Next, the bias values are multiplied by the state of the unit each one is associated with. This new vector is then subtracted from the energy values, leaving the correct energy values in the *ec* and *ef* vectors for all of the training patterns.

4.5.6 Updating Unit States

Using the timing information given by Masscomp for the Vector Accelerator the likely execution times of both the probability function and Gaussian approximation method were evaluated. Tables 4.4 and 4.5 present the corresponding execution times in microseconds of process user time for vectors of length 10 and 100.

Thus for a vector length of 10 the Gaussian method is approximately 8.8 times faster, and for a vector length of 100 it is approximately 6.0 times faster. The timings are for a system in single user mode, with routines called from the C language and all vectors having a stride of 1. It is nevertheless a good indication of the time differences likely to be encountered between the two methods. For both of the approaches, it was assumed that vectors of uniform or Gaussian distributed random numbers were available on the host.

Operation	Vector Length	
	10	100
$-\frac{1}{T} \times \Delta E$ (with -1/T as a constant)	20	29
Exponential function	447	550
Add 1	20	29
Reciprocal	115	137
Load uniformly distributed random numbers	25	55
Compare to random number	25	55
Totals	652	855

Table 4.4 Vector Accelerator Timings for Sigmoid Function Method

Operation	Vector Length	
	10	100
Load normally distributed *N(0,1)* numbers	25	55
Calculate $\sigma \times N(0,1) + \Delta E$	24	33
Compare to 0	25	55
Totals	74	143

Table 4.5 Vector Accelerator Timings for Gaussian Method

The function change(step,patns) is used to change the state of units in Vector Accelerator memory. Before commencement of this routine vectors *ec* and *ef* are assumed to contain the current energy (difference) values. The step parameter indicates the current annealing step and provides an index to the

annealing schedule held in the Accelerator's memory. The particular value σ (temperature) pointed to by the index is used to calculate Gaussian distributed numbers with a mean of 0 and standard deviation given by σ. To achieve this, for each of the patterns in Vector Accelerator memory a vector of *snet* normally distributed numbers is loaded from host memory to Vector Accelerator memory. This vector of *snet* numbers is selected from a vector of 10,000 normally distributed numbers residing in host memory. The index position within the host vector for the starting point of the *snet* length vector is selected randomly. These *snet* Gaussian numbers are then multiplied by the scalar σ indexed by the value of *step,* and within the same Accelerator routine added to the energy values.

Once this has been completed for all the Boltzmann Machines in Vector Accelerator memory, the new states of the units are determined by comparing the energy values to 0. If the energy is greater than 0 the unit is placed in the *on* state, otherwise it is placed in the *off* state. The results for this are placed back into the energy vectors and not directly into the unit vectors. If the Boltzmann Machines are to be updated synchronously the new states for the hidden and free output units are copied to the unit vectors. If however the Boltzmann Machines are being updated asynchronously, only one unit in each machine is allowed to change state. This is selected at random from the new values in the energy vectors and transferred to the unit vectors. The asynchronous mode is only intended for small networks, with no more than approximately 10 hidden/output units, with the number of training patterns restricted to about 4. As only one unit is updated at a time effective use is not made of the Vector Accelerator architecture, and learning time is much extended. No performance measures were undertaken to determine how learning time scales with the number of hidden and output units for the asynchronous mode of operation.

4.5.7 Statistics Gathering

Once the Boltzmann Machines have been annealed, they are run for a certain time and statistics are gathered indicating how often pairs of units are on together. These probabilities are defined as:-

$$p_{ij}^{+} = \sum_{clamp} s_i s_j \qquad (4.8)$$

$$p_{ij}^{-} = \sum_{free} s_i s_j \qquad (4.9)$$

There will be noise present in these estimates, which will depend on the temperature at which the statistics are gathered, (Equation 4.6). The effect of noise in the estimates can be reduced by collecting statistics for a longer time. The simulator allows statistics to be gathered for either a fixed length of time during each learning cycle or for the length of time to be increased as a function of decreasing mean output bit error.

A function collect(patns) performs these statistics collecting operations. The intermediate statistic matrices for the clamped and free machines are held on the host for the same reason that the weight matrix is stored on the host. In updating the statistics the relevant column vector of each of the statistics matrices is loaded into the Accelerator's memory. The clamped statistics are held in ec[0] and the free statistics held in ef[0]. Then for each of the patterns in the Accelerator's memory the functions in Equations 4.8 and 4.9 are performed for the relevant clamped or free phase machines. After having run through all the patterns in Accelerator memory, this information is added to the previous co-occurrence values and stored back on the host.

4.5.8 Updating the Weights

In performing weight adaptation a column vector from the clamp statistics matrix is loaded from the host to the Accelerator memory along with the corresponding column vector from the free statistics matrix. The corresponding column vector from the weight matrix is also loaded into the Vector Accelerator's memory. The free statistics vector is subtracted from the clamp statistics vector. If the result is greater than zero for any of the elements, the corresponding weight is incremented. If the result for any of the elements is less than zero the corresponding weight value is decremented. If there is no difference between

the clamp and free statistics for a particular element, no change to the corresponding weight value is made. The weight column vector is then stored back on the host. The incremental or decremental weight value used is the scalar value stored in the constant area of the Accelerator's memory.

If the architecture of the Boltzmann Machines is to be constrained in any way by for example having no connections between the input and output units or no connections between the hidden units, then at this stage, the weights for connections that are not required are set to zero.

4.5.9 Error Measures

The two error measures available are the G measure of Equation 4.5 and the number of output bits wrong, expressed as a percentage. G is zero if and only if the clamped and free phase distributions are identical; otherwise it is positive. Learning is accomplished by reducing the error measures, which may be used to monitor how learning is proceeding during training. One of these measures is calculated every learning cycle and stored to file on the host. This data is readily graphed to give an indication of how the learning proceeded.

4.5.10 Learning Time Performance

Figure 4.9 depicts performance curves for learning cycle time versus number of hidden units (h) for differing numbers of patterns. For these results the network consisted of 80 input units and 2 output units. This size of network was chosen because it allowed 8 units for each of the ten intonation and perturbation parameters and two output units each representing one of the two possible classes for pathological/healthy voice discrimination. The number of hidden units was varied so as to alter the number of weights that needed to be updated. This particular structure of network is representative of some of those studied in Chapter 6. The learning simulator was run in the synchronous mode with an annealing schedule taking 10 units of time and co-occurrence statistics being gathered for 10 units of time. The network was fully connected.

Table 4.6 shows the number of weights that are updated in fully interconnected networks with differing numbers of hidden units. It should be remembered that the input/output model is used and thus there are no interconnecting links between the input units or biases for the input units. The figures appearing in the weights column of Table 4.6 represent the number of pairs of symmetric interconnection links plus the number of bias values.

It may be seen from Figure 4.9 that an increase in the number of weights to be updated by a factor of approximately 10 (i.e. from 2 to 32 hidden units) gives a resulting increase in learning cycle time by a factor of 6 for 16 training patterns and by a factor of 5.7 for 32 training patterns.

Thus the effect of the start-up time overhead required for each Vector Accelerator MATH or DMA operations maybe observed. As the vector lengths increase, further time savings would result.

Hidden Units	Total Units	No. of Weights
2	84	330
4	86	501
8	90	855
16	98	1611
32	114	3315
64	146	7491

Table 4.6 Total Number of Weights for Different Net Sizes

Using the various approximations made to the Boltzmann Machine procedures, it was observed that there was a probability that the machine could become stuck in local minima that were not globally optimal. The computational expense required to avoid this by using slower annealing schedules and running the

simulator in the asynchronous mode would have been considerable. In many instances, the problem could be avoided by using different starting points and/or modifying the annealing schedule used, thus enabling the learning time to be kept as small as possible.

Using an annealing schedule based on energy difference values as discussed in Section 4.3.4 proved to be generally more successful than using fixed schedules, which were chosen empirically for a given problem. However, although using the energy difference approach allowed a sufficiently high starting temperature to be selected it still meant that the annealing step had to be suitably chosen. This could have been overcome by including an automatic procedure for determining when thermal equilibrium had been reached, and if this was not achieved within a given number of units of time to adjust the annealing step size and try again.

Table 4.7 shows the relative time taken by various functions in the learn simulator. All values are relative to one call of the unit update function *change*.

Function name	Action	Performance Index
change	update state of all unclamped units	1.0
energy	calculate energy for hidden and output units	21.5
clamp	clamp all input units and necessary output units	21.6
collect	collect co-occurrence statistics for all weights	28.8
wt_adapt	adapt all weights	120.8

Table 4.7 Relative Performance of Functions in the Learn Simulator

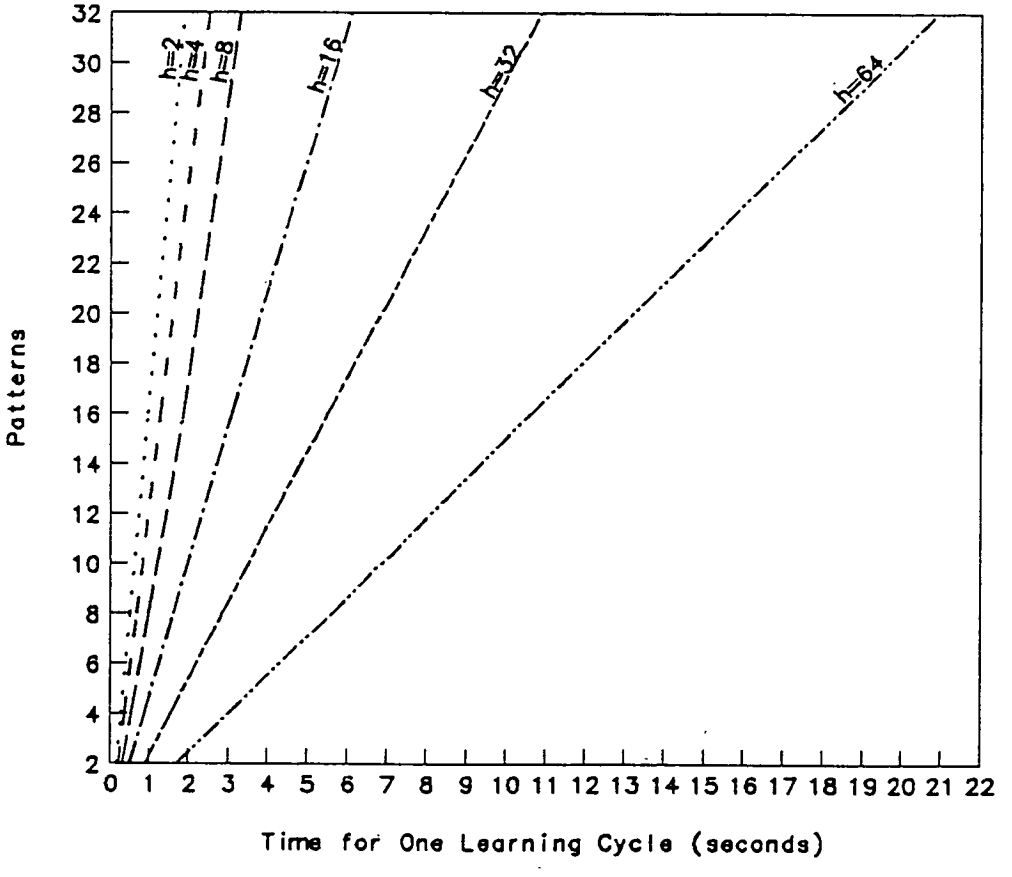

Figure 4.9 Simulator learning time performance.

4.6 SEARCH MODULE

Once a set of weights has been learned it is useful to be able determine the performance of the Boltzmann Machine using these weights with data previously unseen by the machine. From the learning results it is straightforward to determine whether or not the Boltzmann Machine has successfully learned all of the training patterns. If it has not been completely successful, it is useful to determine which of the pattern(s) caused the learning problem. The search program is used to determine this. This is essentially a cut-down version of the learning program. It simulates one Boltzmann Machine in Accelerator memory, with each of the input patterns being clamped to the input units in turn. There is no need to complicate matters here by implementing a large number of Boltzmann Machines in Vector Accelerator memory at once, because the time taken to produce a search output is just the time taken to anneal the network and collect statistics on the output units. It is necessary to collect statistics about how often the output units are *on* or *off* because as the Boltzmann Machine is not deterministic, the output units can change state and hence waver about the convergence point. Collecting the statistics enables the probability of a particular output pattern being valid to be determined.

4.7 GRAPHIC WEIGHTS DISPLAY MODULE

A program was developed to allow the graphical display of learned weights. It makes use of the Masscomp graphics package primitive routines and allows the weight displays produced to be passed to a number of device filters. The information displayed represents the strengths of connections from the hidden units and the strength of connections between the input and output units (if present). The strength of each weight is represented by the area of a box, with the presence or absence of shading of the box indicating the sign of the weight. A thresholding function is provided so that weights can be filtered out and not displayed if they are below a certain percentage of the largest weight value. The scale is indicated by a shaded square representing the largest weight value. The threshold value can also be displayed as a relevant sized square.

This weight presentation technique was adopted because it was felt that it would be easier to attempt an interpretation of the weights developed when they were presented graphically, rather than just as arrays of numbers. In addition, similar techniques are widely used amongst the neural network community. An alternative method to the one chosen, for example, is to use a topological representation as shown in Figure 4.10. Although this method makes the presentation of small networks straightforward, it soon becomes difficult to interpret as the number of connections is increased. It is not the absolute value of each weight that is important, as the value for each weight depends on the value of every other weight. It is the relative magnitudes of the weights that is important.

The program takes the weight information from the weights output file produced by the learning program. The user is prompted for the particular weight values to be displayed depending on the update time for the weights file. The weights are then displayed on the graphics screen and a graphics file is created. The graphics produced are described in more detail in the following two sections. The weight maps included in this thesis are shown only in black and white, however the program is able to produce these maps in colour, which improves their clarity.

4.8 INTERPRETATION OF WEIGHT MAPS

In the weight maps, (see Figure 4.15 for an example), each unit in the network is represented by a square outline. Within this unit outline a weight is represented by a square whose size is proportional to the magnitude of the weight and whose colour represents the sign of the weight (black or shaded for positive, white with black outline for negative). The maps are generally arranged in two sections (if appropriate). One section representing the connections made from the hidden units to all the other units, whereas the other section represents the connections made from the output units to all other units (except for the input units, as this information is to be found in the previous display section). Thus all the weights developed in a network are displayed.

For the display section detailing connections from hidden units, (this section if present is placed at the top of the page), a graphical representation of the complete network outline is repeated for each of the hidden units in the network. Thus in Figure 4.15, because the 4-2-4 encoder network has two hidden units, the complete network outline of 10 units is repeated twice at the top of the page. The particular unit whose connection weights are depicted in any of one of the network outlines is identified by a small upward pointing triangle situated beneath it. The square situated within the unit square pointed to by the triangle represents the bias value for that particular unit.

In the display section detailing the connections from the output units, the network outline is repeated for each of the output units, as before. Again because the 4-2-4 encoder has four output units, the outline of the 10 units is repeated four times in Figure 4.15. In this case it can be clearly seen that there are no connections from the output units to the input units. The only weights displayed are the bias values for each of the output units and the connections that are made between the output units.

The magnitude of the largest weight in the map is indicated in each map along with its correspondingly sized square. Because some of the weights may be very small, and hence make it difficult to determine what their sign is using the reproduction method adopted, a thresholding function is used to filter out any weights whose magnitude is less than a given percentage of the magnitude of the largest weight. Typically, the weight threshold is set to 5%. A zero weight or a weight whose magnitude is below the threshold is indicated by the unit outline being left blank inside. As mentioned in the previous section the value of each weight depends on the value of every other weight in the network, thus the relative magnitudes of the weights are important, rather than the absolute magnitude of the weights.

A positive weight (represented by a black or shaded square) is excitatory, and if one unit it is linked to is *on* causes the input to the other unit to be increased and hence make it more likely for that unit to be *on.* A negative weight (represented

by a white outline square) is inhibitory and if one unit it is linked to is *on* causes the input to the other unit to be decreased and hence make it more likely for that unit to be *off.*

4.9 XOR EXAMPLE

As explained in Section 3.3.4 the implementation of the *exclusive-or* (XOR) function cannot be achieved with a simple two-layer associative network. In such networks a set of input patterns are mapped directly to a set of output patterns at an output layer. This network has no hidden units and so there is no *internal representation.* The XOR problem is illustrated in Table 4.8. Those patterns which overlap least are supposed to generate identical output values. This problem cannot be accomplished by networks without hidden units with which to create their own internal representations of the input patterns.

Input Patterns		Output Patterns
0 0	→	0
0 1	→	1
1 0	→	1
1 1	→	0

Table 4.8 The XOR Problem

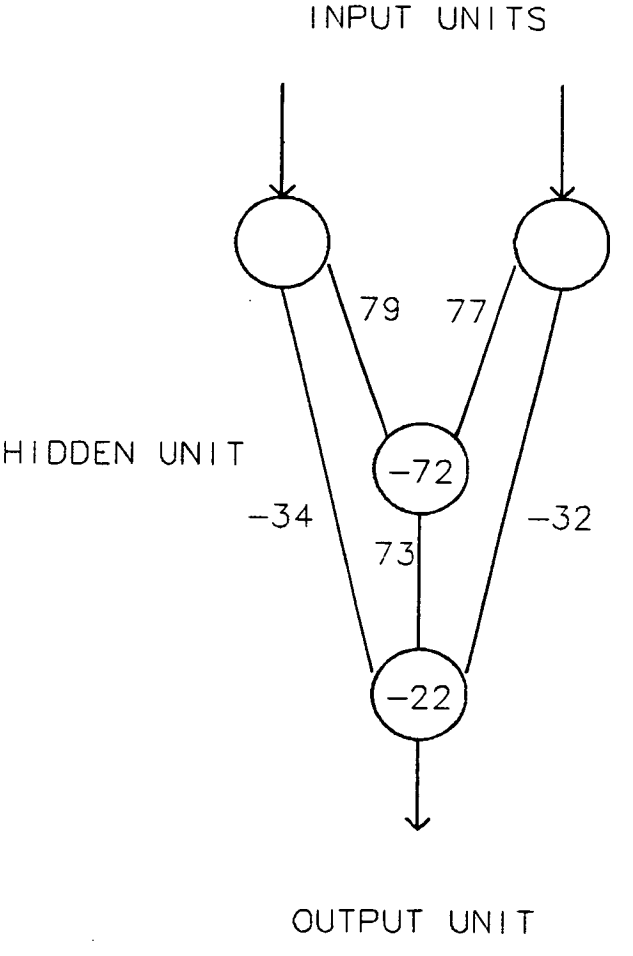

Figure 4.10 A simple XOR network with one hidden unit.

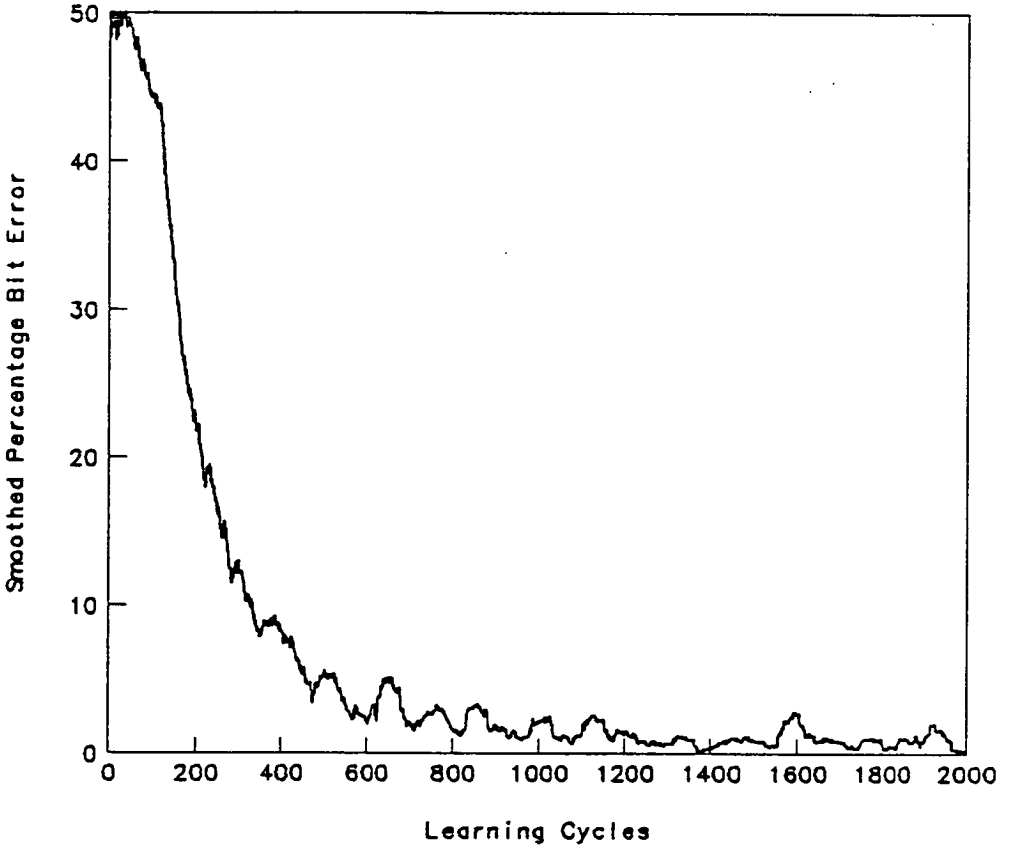

Figure 4.11 Smoothed percentage bit error for XOR learning

Figure 4.10 illustrates how the XOR problem can be solved with a single hidden unit. The weights for the solution shown are those reached after 2000 learning cycles during which each of the four stimulus patterns was presented. The annealing schedule used was logarithmic with a starting value σ (temperature) of 20 and length 10 units of time with temperature being decreased after every unit of time as follows:-

$$\sigma = \frac{0.693 \times \sigma_0}{\log(k+2)} \qquad (4.10)$$

where σ_0 is the starting temperature and $0 \leq k < 10$. Statistics were collected for 10 units of time. The weights were incremented or decremented by a value of 1.0. No noisy clamping was used and the simulator operated in the asynchronous mode.

Figure 4.11 shows the percentage bit error rate smoothed using a moving average technique for a data window of 25 learning cycles.

The graphical weight display output is depicted in Figure 4.12. The four units at the top of the figure represent all the units in the XOR network. The two input units are labelled 1 and 2. The hidden unit is shown in the middle with the single output unit at the bottom. The small triangle pointing to the hidden unit indicates that this unit is the reference unit for this particular display block. In other words, the weight indicated in the hidden unit box is its bias value. The weights in the two input unit boxes are the values of the weights between the hidden unit and each of the input units. The weight in the output unit indicates the size of the link between the hidden unit and the output unit.

The three units at the bottom of the figure indicate the relationships between the input and the output units. In this case the output unit is the reference unit and its bias value is indicated within its unit box. The weights in the input unit boxes indicate the size of the links between the output unit and the two input units. Figure 4.10 provides a topographical representation of the data shown in Figure 4.12.

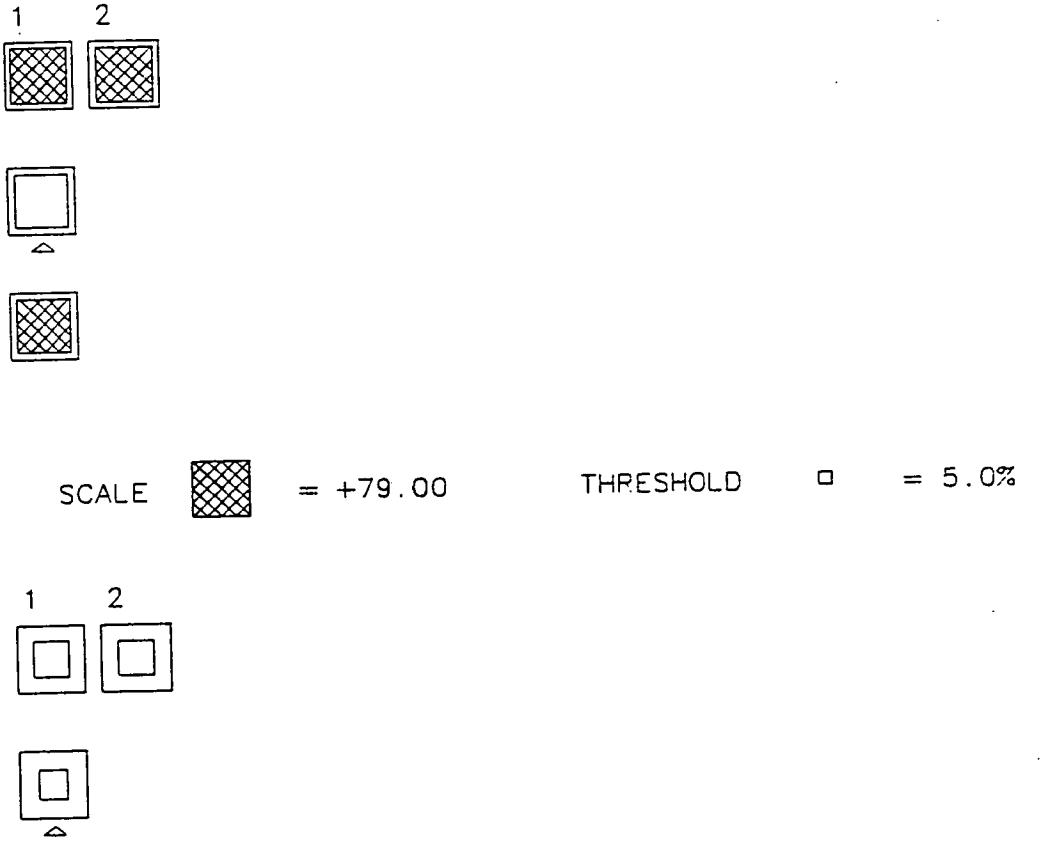

Figure 4.12 Weight map for the XOR network

The solution developed by the network uses the hidden unit to identify when there is a 1 at either or both of the inputs. This hidden unit is then effectively regarded as another input unit from the output units point of view. It is thus as if the input pattern consisted of three rather than two units. Table 4.9 depicts the effective patterns for this *three input system.*

Input Patterns		Output
Input	Hidden	Patterns
0 0	0	0
0 1	1	1
1 0	1	1
1 1	1	0

Table 4.9 The XOR problem showing hidden unit states.

The weight from the hidden unit to the output unit with the weights from the input units to the output unit and the output units bias value ensure that the output unit will not come on when both input units are on.

4.10 4-2-4 ENCODER EXAMPLE

Figure 4.13 illustrates the network topology for a 4-2-4 encoder. The network consists of four input units and four output units. In the formulation depicted here the input and output unit groups each have only one unit on at a time so that there are only four different states for each group. The input and output units are not connected directly but both are connected to two hidden units. To permit perfect communication between the input and output units $h \geq \log_2\left(\frac{v}{2}\right)$ where h is the number of hidden units and v is the number of visible (input + output) units.

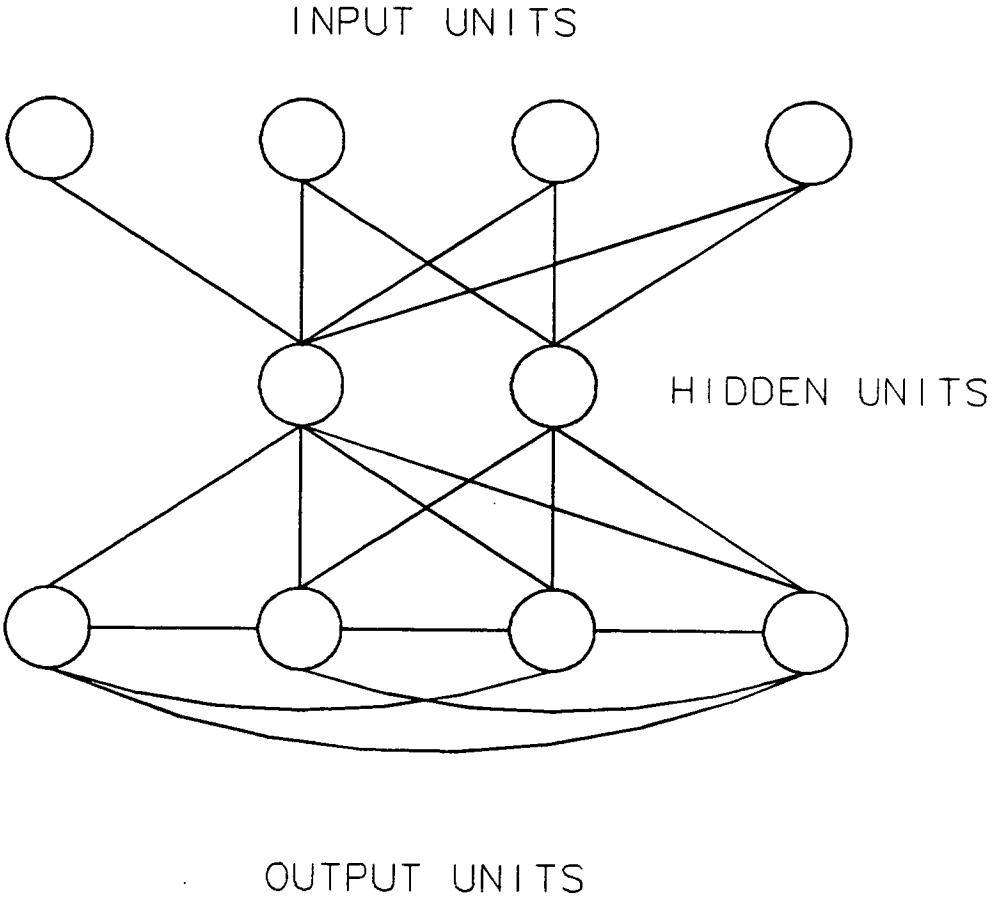

Figure 4.13 4-2-4 Encoder Connections

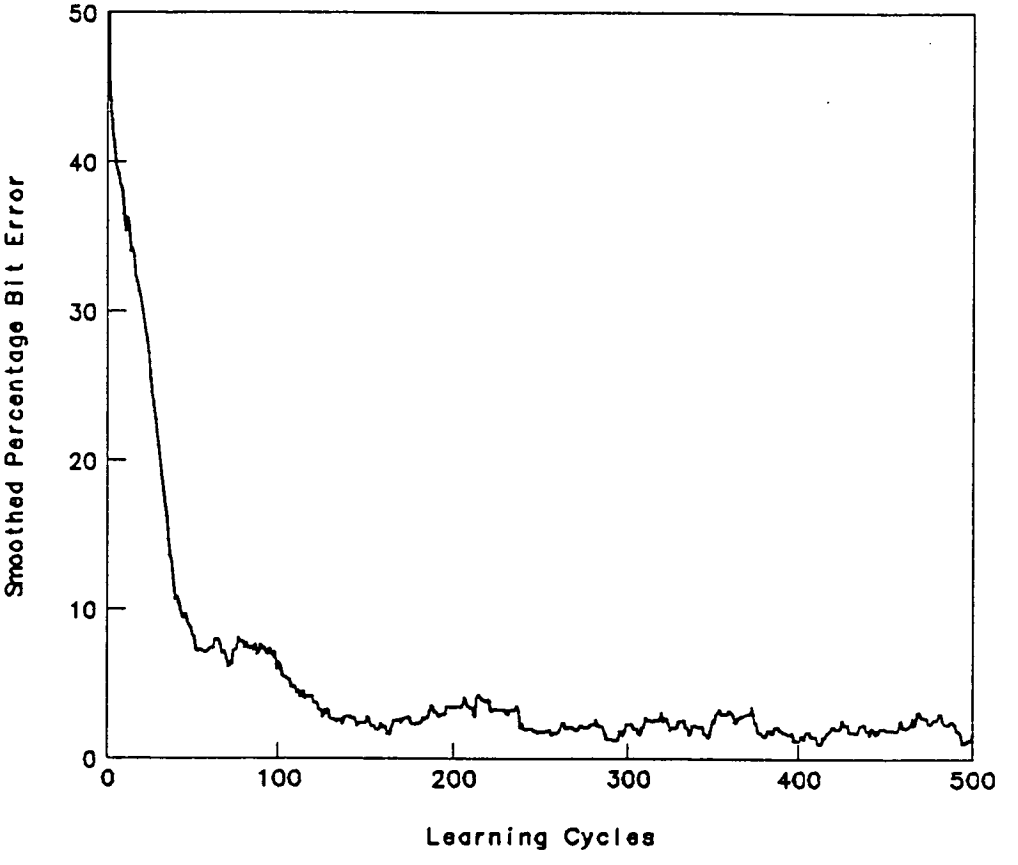

Figure 4.14 Smoothed percentage bit error for encoder learning

The environment patterns consisted of four equiprobable vectors of length four which specify that one unit in the input units and the corresponding unit in the output units should be on together, with all other visible units off. The input units are only connected to the hidden units. The output units are connected internally and each is also connected to the hidden units. The hidden units are not connected to each other.

The annealing schedule followed for this problem used a starting value σ of 10 and followed the logarithmic scheme of Equation 4.10 for 20 iterations. Co-occurrence statistics were collected for 10 units of time. The weight change value was 1.0. Noisy clamping was not used and the simulator was operated in the asynchronous mode. Figure 4.14 depicts the percentage bit error rate smoothed with a moving average window of length 25 learning cycles.

The weights learned at 300 learning cycles are shown in Figure 4.15. That there are no connections between the input and output units and also no connections between the two hidden units is clearly shown. The hidden units encode the input units into a binary format as follows:-

Input unit on	Hidden unit states
1	1 1
2	0 0
3	1 0
4	0 1

Table 4.10 4-2-4 Hidden Unit States

A similar set of weight values are used to decode the hidden units to give the required output states. The connections between the output units indicate that they inhibit each other very strongly.

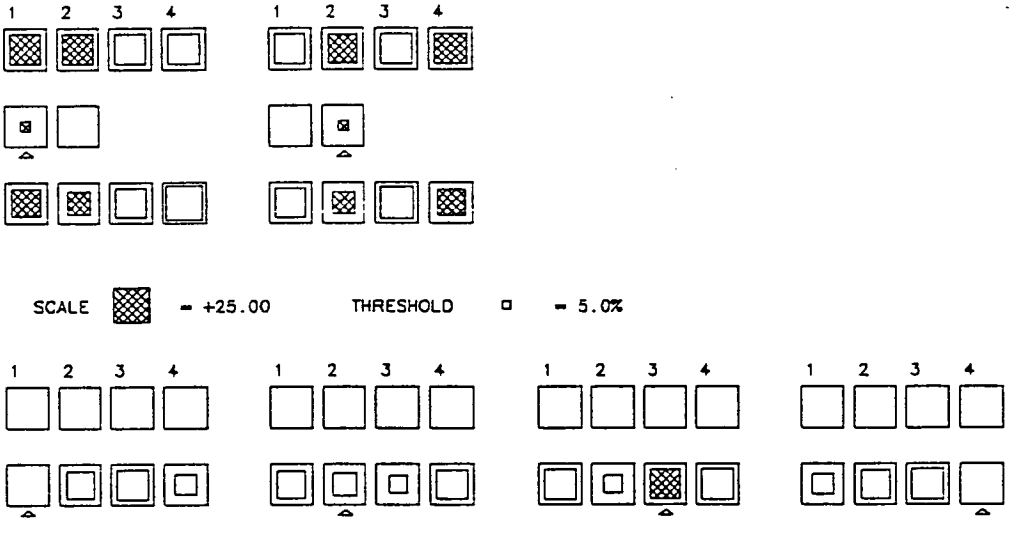

Figure 4.15 Weight map for 4-2-4 Encoder

5 LARYNGEAL PATHOLOGY AND WAVEFORM PERTURBATIONS

5 LARYNGEAL PATHOLOGY AND WAVEFORM PERTURBATIONS

5.1 INTRODUCTION

The larynx is situated at the top of the trachea and houses two lips of ligament called the vocal folds or cords. Situated between them is a slit-like orifice, the glottis. The opening and closing of the vocal cords can take place at varying speeds. This vibration of the vocal cords produces a buzzing sound known as *voicing.* It is the frequency of vibration that determines the *pitch* of the voice. The complex vibratory process by which *voicing* is produced is known as *phonation.* However as Baken comments:-

> *"The phonatory system is not a perfect machine, and every speaker's vibratory cycles are erratic to some extent", (Baken, 1987:166).*

The cycle-to-cycle duration and/or amplitude of the pitch periods of the laryngeal waveform are characterised to a certain extent by apparently random fluctuations. The term perturbation is used to refer to such random deviations from the regularity of the laryngeal waveform. Pitch period perturbation is known as *jitter,* and amplitude perturbation at waveform peaks is called *shimmer.*

Davis (1979) suggested that the measurement of pitch perturbations may be of value in screening for vocal disorders. The feasibility of using pitch perturbations to separate pathological speakers from control speakers has been demonstrated by Laver, Hiller, Mackenzie & Rooney (1986).

In this chapter the opportunity is taken to acquaint the reader with the structure of the larynx, vocal folds and the acoustic significance of various laryngeal pathologies. Well over thirty studies have been published in which pitch period F0 and A0 contours have been examined for perturbatory behaviour as a method for quantifying laryngeal function associated with voice pathology. A brief review of perturbation analysis is presented. The perturbation measurement system of

Hiller (1985), which provided the raw data for the Boltzmann Machine experiments, is also described along with details of the speakers and speech samples used.

5.2 SPEECH PRODUCTION

There are no organs of the human body which have been specifically designed for speech. The parts of the body which produce the sounds of language are from a biological point of view primarily used for breathing and eating. In fact, the muscular activity necessary for speaking must be learned. A diagram of the vocal apparatus is shown in Figure 5.1.

In speaking the lungs fill with air as in breathing, except that the oral tract as well as the nasal tract is used. The airstream from the lungs passes between the vocal cords, which are two small muscular folds located in the larynx at the top of the trachea. Situated between the vocal folds is a slit-like orifice known as the glottis. If the glottis is open (i.e. vocal cords wide apart), as it is normally when breathing out, the air from the lungs will have a relatively free passage into the pharynx and mouth. However if the vocal cords are drawn together so that there is a narrow passage between them, the air flow will increase and the local pressure will be reduced. This very local reduction is a consequence of the Bernoulli effect and causes the cords to be sucked together. With the glottis now closed there will be no flow of air, and the pressure below the cords will be built up until they are blown apart again. The flow of air between them will then cause them to be sucked together again, and the vibratory cycle continues. This process is known as *phonation.* Sounds produced when the vocal cords are vibrating are said to be voiced, whereas those produced when the vocal cords are apart, are said to be voiceless.

The air passages above the vocal cords are known collectively as the vocal tract. Each time the vocal cords open and close there is a pulse of air from the lungs. These pulses act like sharp excitations on the air in the vocal tract, which is accordingly set into vibration in a way determined by its shape and size. The air

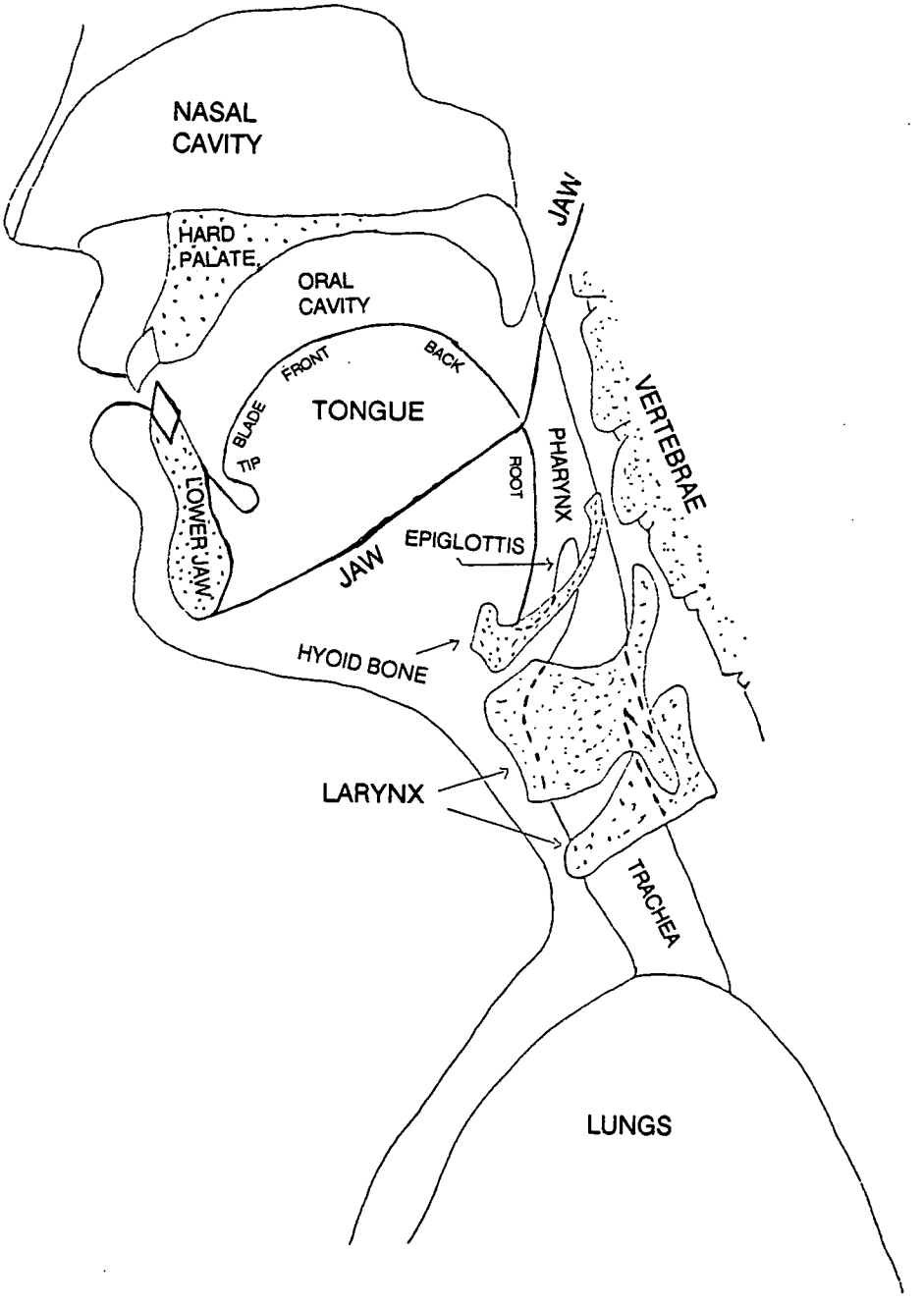

Figure 5.1 The vocal apparatus

in the vocal tract vibrates at a number of resonant frequencies and provided the position of the vocal apparatus remains the same is regardless of fundamental frequency, which is determined by the rate of vibration of the vocal cords. The resonances of the vocal tract are known as formants.

The opening and closing of the vocal cords can take place at varying speeds, the frequency at which they open and close being governed by their tension and by the force of the airstream brought into play on them. The pitch of a voice is determined by the frequency of vocal cord vibration, and is in constant fluctuation while we are talking. Furthermore, voicing is not continuous during speech, as part of the time the glottis is in vibration and part of the time it is not. In normal speech voicing is responsible for most of the noise that is made, and for the carrying power of what we say.

5.3 THE LARYNX

The larynx is a fairly rigid box made up of cartilages, situated at the top of the trachea. The biological function of the larynx is to provide protection by acting as a valve to prevent air from escaping the lungs, and by preventing foreign substances from entering the trachea. As described in the previous section an important secondary function of the larynx is sound production. The fundamental frequency of a voiced sound is a function of the mass, elasticity, compliance and length of the vocal folds. It also depends slightly on the subglottal pressure and the acoustical load of the vocal tract.

Figure 5.2 depicts a cross-section view through the larynx. A full description of the anatomy of the cartilages, muscles and other tissues which make up the larynx is beyond the scope of this thesis. The interested reader is referred to texts by Kaplan (1960), Saunders (1964), Hardcastle (1976), Romanes (1978) and Laver (1980).

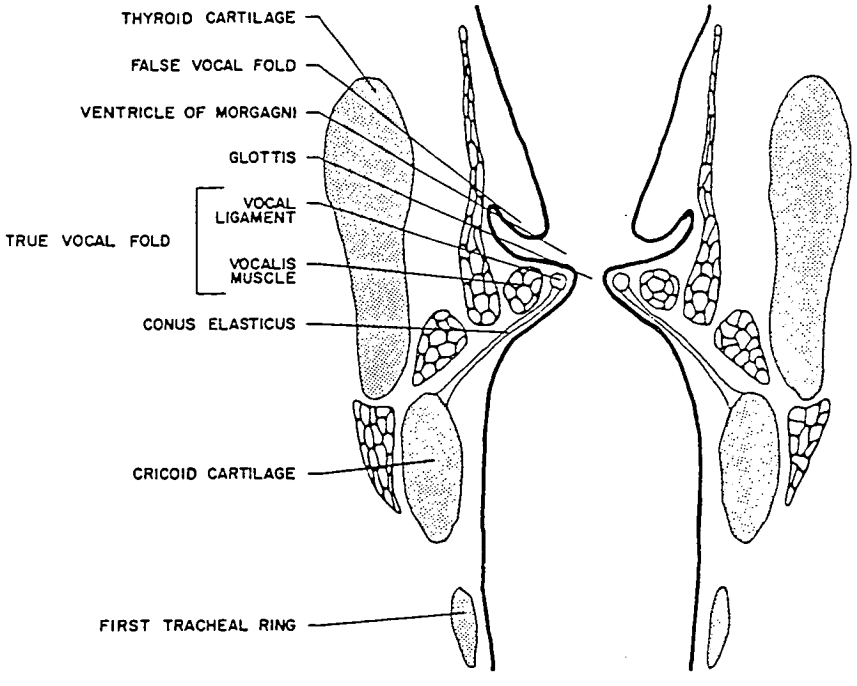

Figure 5.2 Cross-sectional view through the larynx

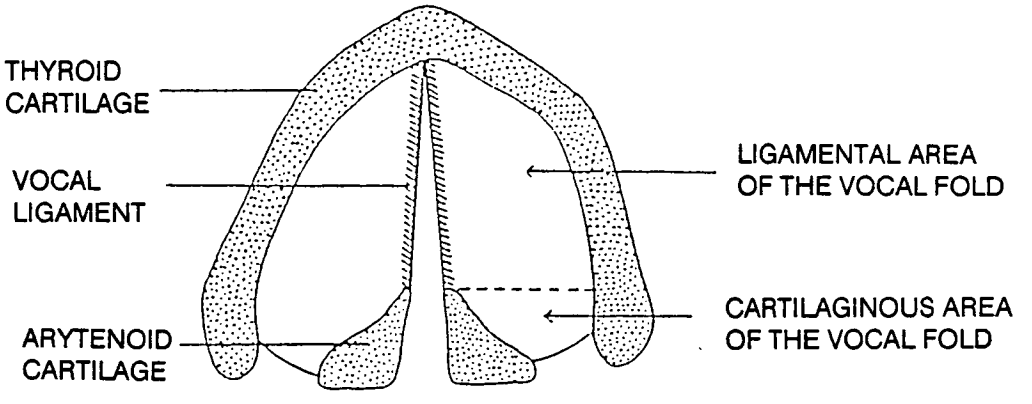

Figure 5.3 Schematic view of the vocal folds, seen from above.

A schematic view of the vocal fold structure is shown in Figure 5.3. The area of the vocal fold that is most freely involved in vibration during phonation is the ligamental area. The vocal fold itself is a layered structure, which in the ligamental area consists of the vocalis muscle and a covering of mucous membrane (Figure 5.4). The cartilaginous area of the vocal fold is also built up from a series of tissue layers with the cartilage lending rigidity to the edge of the vocal fold.

The area of laryngeal interest lies within the true vocal fold, and in particular around the ligamental part of the fold. The ligamental area is the part most prone to pathological problem and change. The cartilaginous area is much less freely involved in vibration and so disorders in this area may have only minimal acoustic consequences. A look in detail at the tissue make up of the ligamental border of the fold shows that there are five different types of tissue layers involved (Mackenzie, Laver & Hiller 1983).

Figure 5.4 shows a schematic representation of the ligamental portion of the vocal fold seen in cross-section. The vocalis muscle makes up the body of the fold and lies under a cover of mucous membrane. This flexible cover is itself made up of four thin layers, each with different mechanical properties. The three inner layers make up the lamina propria and the outer layer is the epithelium. The epithelium is thin, relatively stiff and inelastic and has the same degree of stretchability lengthwise and crosswise. It rests on a very thin base membrane which acts as a biological barrier to infection. Many disorders arise in the epithelium, but do not necessarily spread into the lamina propria or the vocalis muscle (Mackenzie, Laver & Hiller 1983).

The lamina propria is made up of three layers of connective tissue. The superficial layer has been likened by Hirano (1981) to soft gelatin. With the epithelium, this layer acts like a liquid with a very high surface tension. The intermediate layer is much more tightly packed with fibres. These are elastic fibres in an orderly arrangement parallel to the unattached edge of the vocal fold. They are elastic along the length of the fibres, but stiffer across the grain. The tissue is assumed to be incompressible (Titze, 1973). The deep layer is similar to the intermediate

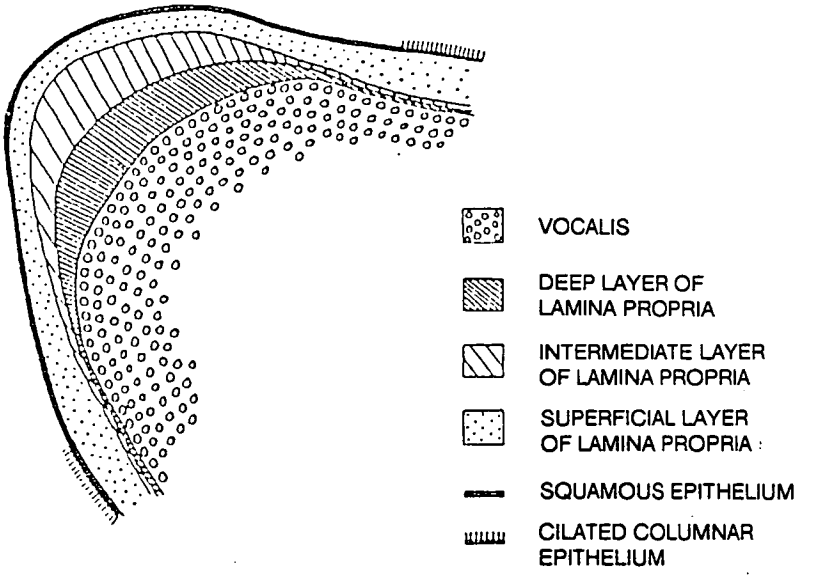

Figure 5.4 Schematic representation of the ligamental portion of the vocal fold, seen in cross section.

layer, but the fibres are mostly formed of collagen. They are flexible, but difficult to stretch. Like the intermediate layer, the deep layer is assumed to be anisotropic and incompressible.

The final layer is made up of the vocalis muscle, part of the thyroarytenoid muscle. The fibres run parallel to the unattached edge of the fold, but their elasticity depends very much on their state of contraction. Hirano et al. (1982) cites research establishing as much as a tenfold difference in elasticity between resting and contracted muscle. The vocalis muscle is anisotropic, being much more elastic longitudinally than crosswise, and is relatively incompressible.

These five different layers behave basically like three relatively independent masses, with the epithelium and the superficial layer of the lamina propria forming one semi-fluid layer, the intermediate and the deep layers of the lamina propria forming a stiffer layer within the cover, and the vocalis muscle forming the third mass as the body. If any of these layers change in mass, stiffness or geometry, then the vibratory pattern of the fold changes.

Every change causes an acoustic effect which is more directly understood in some cases than in others. For example, an increase in mass will normally bring about a drop in fundamental frequency, and an increase in stiffness will normally generate a rise in fundamental frequency and a drop in intensity. Prevention of full glottal approximation by the intrusion of a surface mass will inject inter-harmonic noise into the glottal spectrum (giving audible whisperiness), and asymmetrical structural changes will increase waveform perturbation, increasing frequency jitter and intensity shimmer factors. From an analysis of the acoustic detail, it seems possible to some extent to work backwards, and use the acoustic information as evidence of the possible existence of a given class of laryngeal disorder, which can in turn be checked by laryngological and tissue examination.

Experienced laryngologists may often use simple listening tests to judge existing pathology in a patient's voice. It follows from this that information suggesting pathology is reflected in the speech signal. High-speed motion pictures of pathological vocal folds reveal that frequently there are irregular vibratory patterns

(von Leden, Moore & Timcke, 1960). Measures of pitch period perturbation (Liebermann 1961; 1963; Smith & Liebermann 1964; Koike 1967, 1973; Hiki, Sugawara & Oizumi 1968; Crystal & Jackson 1970; Hecker & Kreul 1971) and measures of amplitude perturbation (Koike 1969) from acoustical waveforms of pathological speakers are different from those of normal speakers. Two psychoacoustical studies (Wendahl 1963, 1966) involving the synthesis of speech with pitch period and amplitude perturbations indicated that the resulting sounds are correlated closely with subjective judgements of roughness, and that these perturbations are indicative of pathological vocal quality.

A fundamental frequency which is judged to be too low when compared to persons of similar age, sex and body size may result in a hoarse, harsh, husky or rough voice (Davis 1976). Low fundamental frequency stemming from functional vocal abuse may lead to contact ulcers (Luchsinger & Arnold 1965). Organic causes may be virilisation, tumours or other enlargements which increase the mass of the folds, or nerve paralysis which decreases the elasticity and compliance of one or both folds.

A study by Laver, Mackenzie, Hiller & Rooney (1986) used discriminant analysis techniques with nine intonation and perturbation parameters to provide correct identification for pathologies of 91.7% for females and 87.5% for males, with false alarm rates of 7.5% for females and 6.0% for males. The study also explored the question of whether the acoustic technique is capable of discriminating diagnostic sub-groups of disorders. Evidence for this was provided by using a comparison of average values for the full set of nine acoustic parameters, in units of standard deviation away from the control group means, for 15 male patients with epithelial disorders versus 11 male patients with polyps or nodules. Beck (1988) also reports qualitative findings for the classification of pathological subjects into three broad classes, epithelial disorders, polyps/nodules and disorders of the cartilaginous area.

The validity of acoustic assessment procedures is dependent on the complex relationship between the vibrating source function and the resultant speech signal output by the production system. The nature of this complex relationship can be summarised from Davis (1979) as follows:-

1) In general, asymmetrical changes in the mass and elastic properties of the vocal folds are created by the presence of laryngeal pathology.
2) These symmetric changes result in the modulation of subglottal airstream by unbalanced vocal fold movement.
3) Irregular air pulses are emitted by the larynx into the supraglottal structures which are then radiated at the lips and nose.
4) The resultant acoustic signal is therefore affected by a disturbance of the vocal folds, and the acoustic speech signal can be used to quantify the disturbance.

Every act of phonation is characterised to a degree by apparently random variations of the cycle-to-cycle duration and/or amplitude of the pitch periods of the laryngeal waveform. Acoustically, perturbation can characterise the laryngeal waveform in both the time domain (jitter) and the frequency domain (shimmer). The term perturbations is used to refer to such random deviations from the underlying regularity of the laryngeal waveform. In the frequency domain, a frictional element can add spectral noise to the laryngeal waveform.

Jitter and shimmer contribute to the auditory effect usually called *harshness*. Physiologically, a voice which is audibly and habitually harsh can arise either functionally from misuse of the phonatory musculature, or pathologically from disruptive mechanical effects of neoplastic growths in the laminar tissue layers of one or both vocal folds.

Spectral noise alone is associated with the auditory effect known as *whisperiness*. When spectral noise is added to jitter and/or shimmer, this has the effect of adding whisperiness to harshness, and the composite voice quality is known as *hoarseness*. Physiologically, the frictional element in whispery and hoarse voices is caused by incomplete glottal closure. Incomplete closure can be the result of

either idiosyncratic, habitual adjustments of the phonatory musculature, or of mechanical intrusion into the glottis of obstructions such as vocal nodules, polyps and other types of growths, or of paralysis of one or both vocal folds.

Perturbation in the laryngeal waveform of a given speaker can occur across a wide range of incidence, from occasional to habitual, and in degree from slight to severe. Severe perturbations are almost always signs of either pathological or functional disorder, but slight perturbations occurring occasionally are evident in all speaking voices.

5.4 LARYNGEAL PATHOLOGY

5.4.1 Diagnosis

When patients complain of harsh or hoarse voices it is important to determine whether the symptoms present are due to pathological causes or functional causes. Historically, physicians have relied on two basic techniques in the diagnosis of pathological conditions in the larynx. These are auditory evaluation of the voice and visual examination of the larynx. Since laryngeal diseases are often accompanied by changes in the voice, simple listening tests sometimes give useful information. The principal criticisms of this auditory method are its subjectivity and its lack of absolute quantitative standards.

Visual laryngeal examination is usually completed by indirect laryngoscopy which involves the placing of a mirror in the patient's throat so that the vocal folds and surrounding tissues maybe observed. This visual examination provides a restricted supralaryngeal view of the larynx under static conditions. An improved visual examination of the larynx can be achieved with other instruments such as the fibreoptic laryngoscope. All these techniques are intrusive and may not be readily accepted by some patients. To allow a view of the vocal folds in motion both the indirect and direct methods may be combined with stroboscopic illumination. It is particularly important to obtain a dynamic view of phonatory

action, since neoplastic growths below the surface of the vocal folds may display their effects only in terms of interference with symmetrical and regular vibration of the folds.

Ultrasound techniques may also be used for inspection of the vocal folds and are comparable to laryngoscopic techniques in that they afford a view of the vocal fold geometry. These techniques can be useful when the patient is not able to endure laryngoscope examination, but their low resolution tends to make them of limited value.

Neither laryngoscopy nor ultrasonic scanning techniques give directly quantifiable information about pitch perturbations. However they can give important background information about the probable mechanical state of the vocal folds.

Auditory assessment of the phonatory quality of a patient's voice suffers from extraneous factors, with differences between individuals' perceptions being the greatest compounding factor. The development of an automatic acoustic technique for screening voices for the presence of laryngeal pathologies and the differentiation of such pathologies is seen as being complementary to the techniques already used by the laryngologist.

5.4.2 Screening

Unlike vocal injuries which have a functional origin, diseases are acute or chronic organic disorders. The most important ones are benign and malignant tumours and paralyses. In all cases a structural deviation is initially responsible for a change in the voice. As the disease advances, especially in malignant pathologies, the voice becomes progressively more hoarse, and in the majority of cases, surgery or radiation may be necessary. The best way to insure good prognosis and to avoid serious complications is early diagnosis and treatment.

The detection of laryngeal pathologies by an acoustic analysis system has several potential applications (Laver, Hiller, Mackenzie & Rooney, 1986):-

1) Screening of populations known to be at-risk alongside existing screening programmes in hospitals. An acoustic system has the advantage of being completely non-invasive, and the recording procedure is simple and causes minimal distress to subjects. It would also be highly portable so that screening could be extended to factories. At-risk populations include factory workers in potentially laryngeal-damaging environments such as flour-mills, cement factories and the asbestos industry.
2) Priority assessment of patients visiting their general practitioner with complaints of harshness or hoarseness. The use of an automatic acoustic system could speed the process of referral for laryngeal examination by specialists with hospital-based fibreoptic laryngoscope facilities in those cases where evidence of serious pathology was indicated
3) Diagnostic support where a particular laryngeal pathology is already suspected. This would depend on the acoustic system being shown to reliably discriminate one type of pathology from another, or general evidence of pathological versus functional aetiology.
4) Monitoring to assess changes with time of the phonatory efficiency of patients receiving surgery, radiotherapy, chemotherapy or speech therapy, or to track deterioration or remission in progressive disease.

5.4.3 Structural Pathologies of the Larynx

A vocal disorder must show either a structural or a functional change from the characteristics of the healthy normal larynx to have an effect on the acoustic signal. From Mackenzie, Laver, Hiller (1983) guidelines to the acoustic consequences of changes in physical parameters to the larynx may be summarised as follows:-

1) Mass. An increase in mass adds inertial force to the vocal fold , which will tend to decrease the frequency of oscillation. It may be expected to exert its effect most strongly at the onset of phonation, when the vocal fold is accelerating from a relatively stationary position.

2) Stiffness. Increasing the stiffness of a vibrating body should inhibit the vibratory movement, causing a decrease in amplitude of excursion.

3) Protrusion. Protrusion of a mass into the glottal space will only interfere with vocal fold approximation if it is relatively localised. A distinction must be made between localised and non-localised protrusions. An example of localised protrusion is a vocal polyp, which may become wedged between the vocal folds, thus preventing the folds from meeting. A non-localised protrusion may occur due to mild inflammation (a uniform swelling along the full length) of the vocal folds during upper respiratory tract infections.

4) Asymmetry. Asymmetry of the vocal fold structure may cause the two vibrating folds to move out of phase with each other, with complex consequences for the acoustic waveform. Structural asymmetry is a feature of many laryngeal pathologies including vocal polyps and papillomata.

Mackenzie, Laver & Hiller (1983) attempted to summarise pathologies in terms of the presence or absence of mass and stiffness changes, protrusion into the glottal space, symmetry, and tissue layer geometry (Table 5.1). From this work an important point emerged concerning the potential power of acoustic screening to differentiate between disorders. Some clinically separable disorders may be expected to impose rather similar mechanical constraints on vibration, and hence on acoustic output, so that they are unlikely to be separable by a solely acoustic assessment procedure. An example of this is the grouping of papilloma, squamous carcinoma, and verrucous carcinoma, all of which may show an asymmetric increase in mass and stiffness originating in the epithelium, with protrusion into the glottis and altered tissue layer geometry. It therefore appears more realistic to attempt a healthy/pathological speaker separation or the discrimination of sub-groups of disorders that have similar mechanical effects.

In conclusion, the relationship between diagnosis of pathology, structural status of the vocal folds, mechanics of their vibration and the resulting acoustic output are complex. There will seldom be one-to-one links (Mackenzie, Laver & Hiller

1983) to be traced between them, and this is particularly true of the link between diagnosis of pathology and the structural status of the vocal folds. From an examination of Table 5.1 it would appear not to be possible to discriminate individual conditions based on the structural status of the vocal folds. In addition a given pathology may show different structural attributes in different individuals, or at different stages of development. However Table 5.1 indicates that the identification of sub-groups of disorders rather than individual disorders may be achievable and results from Laver, Mackenzie, Hiller & Rooney (1985) and Beck (1988) support this to a limited extent for a small number of sub-groups.

For completeness some of the pathologies tabulated are described giving detail of their mechanical properties and consequences for laryngeal vibration as highlighted by Mackenzie, Laver & Hiller (1983).

The most common injury to the larynx is vocal abuse, which is usually a functional problem. At the onset, it may be brought on by improper use of the voice during prolonged speaking or singing, often with excessive tension, If the problem persists, changes in phonatory structures may result to produce disorders such as functional fatigue, nodule, polyp, contact ulcer or chronic laryngitis.

Functional vocal fatigue (myasthenia laryngis) is characterised by weakness of the vocal muscles. This allows the vocal folds to be lengthened, stretched and tensed so that the balanced tension of the folds is not maintained, and the voice sounds hoarse.

Hyperplasia is an increase in cell number resulting from rapid division of the basal cell layer. The increase in basal cell number may cause buckling and distortion of the basement membrane, but the stratified arrangement of cells is maintained, and the cells appear normal. Hyperplasia occurs anywhere within the laryngeal epithelium. It is common at the centre of the ligamental area of the vocal fold. Mechanical factors associated are an asymmetric increase in mass, with normal tissue layer geometry.

	PATHOLOGY	Disrupted tissue layer geometry	Mass Change	Stiffness change	Protrusion	Asymmetry
A.	LIGAMENTAL PORTION					
A.1	EPITHELIAL					
A.1.1	Hyperplasia		+			+
A.1.1	Keratosis		(+)	+	(+)	+
A.1.1	Carcinoma-in-situ		+	+	(+)	+
A.1.2	Squamous carcinoma	+	+	+	+	+
A.1.2	Verrucous carcinoma	+	+	+	+	+
A.1.2	Adult papilloma	+	+	+	+	+
A.2	SUPERFICIAL L.P.*					
A.2.1	Reinke's oedema		+		N.L	
A.3	UNSPECIFIED L.P.*					
A.3.1	Vocal nodules		+		+	(+)
A.3.1	Vocal polyps (sessile)	(+)	+	+	+	(+)
A.3.1	Acute laryngitis		+		N.L.	
A.3.1	Chronic laryngitis		+		N.L.	
A.3.1	Chronic hyperplastic laryngitis		+	+	N.L	
A.3.1	Fibrome		+	+	+	+
A.3.2	Vocal polyps (pedunculated)	+	+	+	+	(+)
A.4	VOCALIS MUSCLE					
A.4.1	Sarcoma		+	?		+
B.	CARTILAGINOUS PORTION					
B.1.	EPITHELIAL (as under A.1.)					
B.2.	UNSPECIFIED L.P.*					
B.2.1	Acute oedema		+		+	
B.2.2	Contact ulcer	+	+	+	+	(+)

L.P.* = lamina propria
+ = presence of a factor
(+) = possible or variable presence
N.L. = non-localised protrusion, not expected to prevent vocal fold approximation

Table 5.1 A summary of mechanical characteristics of vocal fold pathologies (from Mackenzie, Laver & Hiller, 1983).

Keratosis is a condition in which the squamous cells of the epithelium begin to produce keratin, which is laid down as a layer at the surface of the epithelium. It may form a large whitish mass, which protrudes into the glottal space and may interfere with vocal fold approximation. The site of occurrence is as for hyperplasia. Mechanical factors associated are an asymmetric increase in stiffness, with normal tissue layer geometry. Eventually there may be a significant increase in mass and protrusion into the glottal space.

Tumours are tissue masses which have no function and which develop from structures normally present. A benign tumour does not infiltrate neighbouring tissue, although it may displace or crowd nearby areas. A malignant tumour such as cancer, may invade and destroy adjacent structures, and it may migrate via the blood or lymph circulation systems to other sites. Because of the danger of the malignancy spreading, it is very important to recognise and treat such tumours early, since there is then a significantly higher probability of a complete cure. Tumours may develop anywhere within the larynx. When located on the vocal folds, they cause hoarseness at an early stage. Subsequent removal may build up scar tissue and hinder effective phonation.

Carcinoma-in-situ is usually regarded as the earliest recognisable stage of cancer of the larynx, although it is not an inevitable precursor of invasive cancer, and not all cases of carcinoma-in-situ necessarily progress to become fully invasive. Carcinoma-in-situ occurs anywhere within the laryngeal epithelium. Mechanical factors associated with it are an asymmetrical increase in mass, with normal tissue layer geometry and a variable increase in stiffness and protrusion into the glottal space. (Baur & McGavran, 1972; Ferlito, 1974; Friedmann & Osborn, 1978).

Squamous cell carcinoma is the commonest type of laryngeal cancer. It may occur anywhere within the larynx and is most common in the ligamental portion of the vocal fold. Mechanical factors associated with it are an asymmetric change in mass and stiffness, with disruptive tissue layer geometry and protrusion into the glottal space. (Ferlito, 1974; Michaels, 1976; Friedmann & Osborn, 1978, Shaw, 1979).

Verrucous carcinoma is a specific type of squamous cell carcinoma, which presents as a slowly growing warty mass. It may occur any where within the larynx, although is commonest in the ligamental portion of the vocal fold. Mechanical factors are an asymmetrical increase in stiffness and mass, with localised protrusion into the glottal space and disruptive tissue geometry. (Ferlito, 1974; Michaels, 1976; Friedmann & Osborn, 1978, Shaw, 1979; Maw et al., 1982).

Papilloma is a benign warty tumour, which in adults forms multiple branch-like projections of highly keratinised epithelium. There may be extrusion of thin columns of lamina propria into the tumour, so that tissue geometry is substantially disrupted. They are commonest at the edge of the ligamental portion of the vocal fold or at the anterior commissure. (Friedmann & Osborn, 1978, Shaw, 1979; Hall & Colman, 1975).

Reinke's oedema is a specific form of chronic laryngitis which is characterised by the loosening of the attachment between tissue layers in the ligamental portion of the vocal fold. Both vocal folds are affected along their full length. Mechanical factors associated with this are a symmetrical mass increase with non-localised protrusion into glottal space. Tissue layer geometry is normal but with weakened adherence between layers. (Saunders, 1964; Birrell, 1977; Friedmann & Osborn, 1978).

Vocal nodules are benign growths which vary in size, and may have a pointed or round shape and a white to red colour. They probably represent inflammatory responses to trauma or vocal abuse over a prolonged period of time. The size and shape of the nodule determines the degraded quality of the voice. Vocal nodules usually occur on the edge of the vocal fold in the centre of the ligamental portion. The mechanical factors associated with the early stages of vocal nodules are a symmetrical or asymmetrical increase in mass, with localised protrusion into the glottal space and normal tissue layer geometry. Stiffness is only increased slightly. (Arnold, 1962; Michaels, 1976; Aronson, 1980).

Polyps are very similar in appearance to nodules. Vocal polyps may be sessile or pedunculated. One distinguishing feature may be the appearance of a base or stalk between the polyp and the vocal fold. Unlike nodules, the vocal abuse causing the injury need not be of long duration and may sometimes be related to a single event of vocal strain. The effect on voice quality is similar to the effect of nodules. Vocal polyps usually occur at the edge of the ligamental portion of the vocal fold. Mechanical factors associated with vocal polyps are an asymmetrical (or rarely symmetrical) increase in mass and stiffness, with localised protrusion into the glottal space. Tissue layer geometry is significantly disrupted only if growth is pedunculated. (Arnold, 1962; Aronson, 1980; Greene, 1972; Hall & Colman, 1975; Michaels, 1976).

Chronic laryngitis has many causes which include smoking, alcoholism, repeated attacks of acute laryngitis and voice abuse. Chronic laryngitis may be rather variable in form and the whole larynx may be involved. Mechanical factors associated with this are a symmetrical increase in mass, with non-localised protrusion into the glottal space, and normal tissue layer geometry. (Saunders, 1964; Hall & Colman, 1975; Friedmann & Osborn, 1978; Aronson, 1980).

Contact ulcers are found less often than nodules or polyps. They may be caused by improper use of the voice during prolonged forceful speech. Contact ulcers are generally thought to develop from a localised area of inflammation over the vocal process of the arytenoid cartilage, which is the point of maximum impact during adduction of the cartilages for phonation. Contact ulcers occur on the mucosa overlying the vocal processes of the arytenoid cartilages. Mechanical factors associated with them are an increase in stiffness with a redistribution of mass, localised protrusion into the glottal space, and disrupted tissue layer geometry. The degree of symmetry is variable. (Perkins, 1977; Aronson 1980).

5.5 PERTURBATION STUDIES

Measures of pitch period perturbation (Liebermann 1961, 1963; Smith & Liebermann, 1964; Koike, 1967, 1973; Hecker and Kreul 1971) and measures of amplitude perturbations (Koike, 1973) from acoustical waveforms of pathological speakers are different from those of normal speakers.

Ephemeral variations in the mucal conditions of the fluid bathing the surface of the vocal folds can also cause momentary perturbations, often eliminated by the speakers clearing their throats. Other slight perturbations in healthy voices during the course of phonation probably have a neuromuscular origin (Baer, 1978, 1980, 1981). Increased perturbation is also one symptom of the aging voice (Benjamin 1981; Ramig and Ringel 1983; Wilcox and Horii 1980).

Perturbatory differences of the cycle-to-cycle period in the normal voice are very small. Liebermann (1963) found that in normal adult male speakers, only some 15% of such differences typically exceed 0.5ms in duration. Perturbations of the cycle-to-cycle period below about 1% of the local fundamental frequency or intensity seem not to be audible as such, though they can be registered by acoustic or physiological analysis. An important consideration in the ability of acoustic techniques to register perturbation data is therefore the sensitivity of the acoustic technique itself. The major factor controlling such sensitivity is the sampling frequency used. If too low a frequency is employed, the minimum difference of period that can be registered when comparing successive cycles is unsuitably large (Heilberger & Horii 1982; Hiller 1985). Sampling frequencies giving suitable resolution to detect low levels of perturbation in waveforms from adult male speakers and most adult female speakers range should be not lower than about 20kHz.

The attempt to discover objective acoustic and physiological methods for characterising laryngeal waveforms in terms of perturbation parameters now has a history of over 30 years, starting from the pioneering work of researchers such as Moore and von Leden (1958), Moore (1962), Liebermann (1961, 1963) and Michel (1964). The literature of perturbation studies is already substantial.

Interested readers are referred to Hiller (1985) who reviews the studies in detail and to Baken (1987) who gives a compact review of different approaches to the measurement of perturbation.

Most studies of waveform perturbation can be classified according to two main types of perturbation parameter. A frequency or period perturbation parameter (jitter) quantifies the degree of regularity displayed by the temporal components of the fundamental frequency of the speech signal. Measures of amplitude perturbation (shimmer) evaluate the regularity of the peak amplitude structures associated with the speech waveform's fundamental periodicity. Secondly, two basic units of waveform perturbation have been used to produce frequency and amplitude perturbation parameters. The cycle-to-cycle perturbation describes the relationships between adjacent pulses of vibration as seen in speech waveforms. In the trend line approach, the basic unit of perturbation is measured as the deviation of F0 and A0 values from equivalent smoothed values produced by a local statistical smoothing algorithm. Thirdly perturbatory behaviour has been observed for two types of speech sample including sustained vowel phonations and samples of connected speech. A majority of studies have measured cycle-to-cycle frequency and amplitude perturbation parameters from samples of sustained vowel phonations. Only a small number of investigations have applied trend line analysis techniques to samples of connected speech in order to derive measures of frequency and amplitude perturbation.

A disadvantage of taking isolated vowels is that the production of speech in such circumstances is a less demanding task for the subject, who may have learned to mask the effects of pathology by limiting his or her production to a more restricted but manageable zone of vocal performance.

The advantage of taking continuous speech as input to perturbation analysis is that it is more likely to yield representative data. In the case of pathological phonation, the production of continuous speech may tax the laryngeal mechanism in a manner which makes the pathology more evident (as reflected in increased magnitude of perturbatory parameter values). The analysis of period

and amplitude values from continuous speech is more difficult than from sustained monotone vowels since the detecting algorithm has to examine a variety of signal structures produced by interactive segmental effects of the dynamic movements of the articulators, together with multiple voicing onsets and offsets.

If continuous speech data is used for perturbation analysis then a minimum duration of speech material is required to stabilise long-term measurements of perturbation (e.g. the mean and standard deviation of each perturbatory parameter). Hiller (1985) reported that a 40-second sample of read speech provided relatively stable long-term speaker-characterising parameters of perturbation for healthy male and female speakers.

5.6 PERTURBATION MEASUREMENT SYSTEM

This section contains material relating to the data used in this thesis. This data has been made available to the Research Group in which the Boltzmann Machine studies were carried out, and acknowledgement is due to Dr. Steven Hiller. An indication of the processing required for the speech samples used, and detail of the choice of control parameters for determining the various acoustic parameters is appropriate. This section provides the necessary detail. A full description of the data preparation is to be found in Hiller (1985).

Briefly, the Hiller (1985) Perturbation Measurement System was comprised of the three following major components:-

1) Pitch Detection algorithm, using a modified parallel processor operating in the time domain to extract fundamental frequency and amplitude contours from samples of connected speech.
2) Non-linear smoothing algorithm, comprising of a running-median and Hanning window which is applied to the contours to produce trend lines of fundamental frequency and amplitude, from which excursions of the unsmoothed values may be extracted.

3) Statistical evaluation of long-term parameters of intonation and perturbation. The excursions of fundamental frequency and amplitude are statistically evaluated to produce the perturbation parameters.

5.6.1 Pitch Detection

The basic scheme of the parallel processor pitch detector has been described by Rabiner and Schafer (1978) as follows:-

> *"1. The speech signal is processed so as to create a number of impulse trains which retain the periodicity of the original signal and discard features which are irrelevant to the pitch detection process.*
> *2. This processing permits very simple pitch detectors to be used to estimate the periodicity of each impulse train.*
> *3. The estimates of several of these simple pitch detectors are logically combined to infer the period of the speech waveform." (Rabiner & Schafer, 1978).*

A block diagram of the parallel processor is shown in Figure 5.5 adapted from Gold & Rabiner (1969).

The speech data used by Hiller (1985) and Beck (1988) was recorded on high quality analogue tape recorders (Revox A77 & Uher 4000). The data was then digitised at 20kHz with 12-bit quantisation. A 20kHz sampling rate provides a resolution of 0.05ms for each pitch period detected by the parallel processor. Prior to digitisation, the input speech was passed through an analogue filter which prevents aliasing of the waveform during sampling. The analogue filter was a Butterworth type which produced a -48dB/octave roll-off beyond a cutoff frequency of 10kHz.

The first pre-processing step in the pitch period detection process was the phase compensation of low-frequency distortions of the input voice sample. These phase distortions were the result of reactive and resistive components in the tape recording and playback system which did not maintain the relative phases of the harmonics of the recorded system (Olsen, 1982). A more accurate representation of the original speech signal was produced by compensation of the low-frequency phase distortions in the recorded (and digitised) version of the signal. It was found by Hiller (1985) that low-frequency phase distortions of tape recorded voice

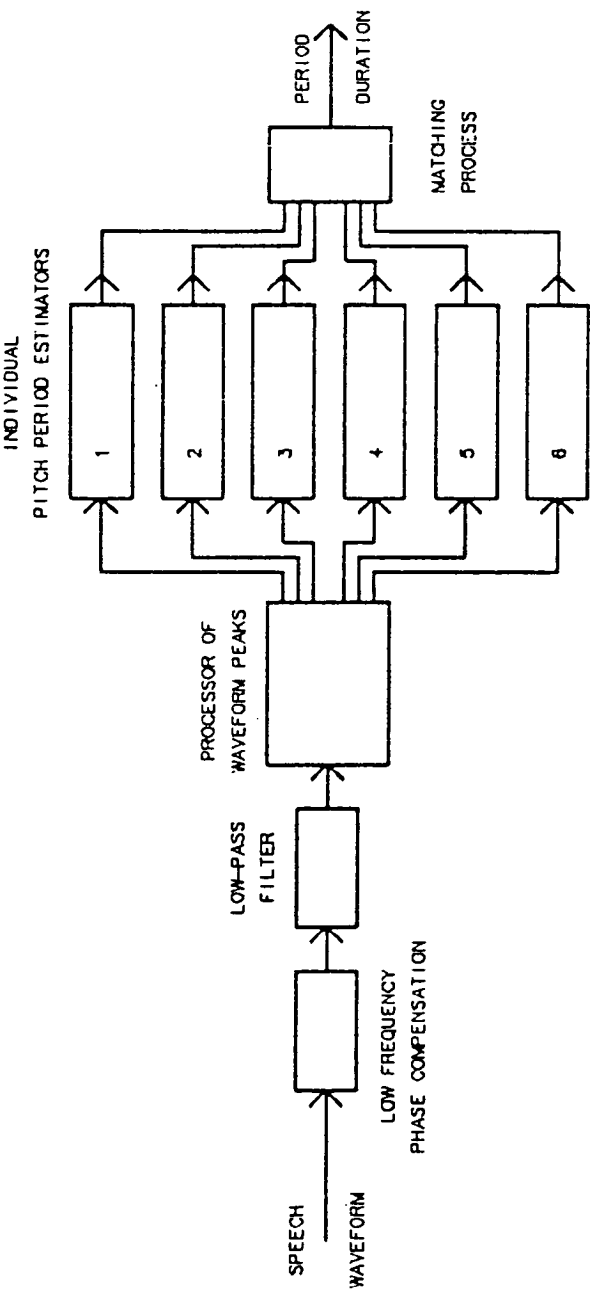

Figure 5.5 Parallel processor pitch detector

samples adversely effects the discrimination of healthy and pathological speakers by waveform perturbation parameters. The phase compensation technique employed was the frequency domain procedure of Berouti, Childers & Paige (1977).

A digital linear-phase filter (McClellan 1975) consisting of 32 coefficients which produced approximately -48dB/octave roll-off beyond the cutoff frequency was used to low-pass filter the phase compensated voice samples to reduce formant information. For male speakers the cut-off frequency was set to 600Hz.

The silence detector used was that of Gold (1964) in which a segment of speech is searched for two samples which exceed a pre-determined *silence* energy threshold. If the energy threshold is exceeded then the remainder of the estimation is completed, otherwise the pitch period result is set to zero and the next frame of data is processed. If speech is present, then the smoothed speech is examined for the presence of *peaks and valleys* which represent periodic behaviour in the waveform. Several measures of amplitude are calculated as each valley and peak is located.

The first stage of the basic extractor was to minimise the data by finding anchor points for determining periodicity and the elimination of the remaining data samples. For the parallel processor anchor points are peak minima (valleys) and maxima (peaks). Six functions of peakedness were derived for the local minima and maxima within the preprocessed speech signal. The sampling resolution of each detected peak was increased by parabolic interpolation of the three sampled data points which define the peak. The use of six different measures of waveform characteristics was designed to cover a range of different types of waveform, varying from a simple sinusoid to a signal composed of a weak fundamental component with a strong second harmonic. Each type of peak and valley measurement produced an impulse train made up of positive impulses representing the amplitudes and locations of the measurements.

The second stage of the basic extraction process of the parallel processor was the selection of the remaining samples which are likely to represent a period delimiter and the rejection of other samples. This step was completed for each of the six impulse functions based on the peak measurements. For each function, the marker detection was completed by a time-varying exponential rundown circuit.

Each impulse train was evaluated for periodicity by a peak detecting circuit based on an exponential decay function (Gold 1962). Following the detection of a possible pitch period marker, the circuit is reset and held for a blanking interval during which no detection occurs. After the blanking interval, the circuit begins to decay. The decay continues until an impulse of sufficient amplitude exceeds the decay threshold, and then is once again reset. In this manner, possible pitch period information is stored and extraneous data discarded. The decay behaviour of the exponential circuit (i.e. blanking time and decay rate) was dependent upon local pitch period trends in order that reasonable limits were set for the detection of the next period.

For each analysis interval the peak detecting circuit produced six estimates of the pitch period, one for each of the six impulse trains. These estimates of periodicity were combined with the two most recent sets of estimates from the six parallel pitch period detectors. The final estimation of the pitch period was based on a comparison of all the estimates. The estimate with the greatest level of agreement among the six immediate candidates was declared the pitch period for the speech segment.

Voiced/voiceless decisions were determined from the level of agreement between the chosen pitch period estimate and the other period measures, Gold (1964). For voiced speech the agreement level would be high since each simple detector represents redundant information concerning the periodic behaviour of the waveform. There is a lack of redundancy associated with noisy voiceless

speech and therefore a low level of agreement for any pitch period estimate. A voiced/voiceless decision threshold was determined from the distributions of the agreements calculated for voiced and voiceless speech (Gold 1964).

Each pitch period estimation was completed on a segment of filtered speech data selected by a rectangular analysis window. The interval within the window was set at 25ms for male speakers and was designed to accommodate the largest probable pitch period. Since three sampled data points were required to define a peak minimum or maximum in the pre-processed speech signal contained within an analysis interval, sampled data points were lost at the beginning and end of the interval. Thus the longest possible pitch period permitted in a given interval was slightly less than the length of the window. For a 25ms window and a sampling rate of 20kHz, the largest duration pitch period is 24.9ms (40.2 Hz).

Cycle-to-cycle data was estimated by shifting the rectangular window along the data in such a way as to try to bring just one new pitch period into the window. Variable shifting was used with range limits of 40Hz to 240 Hz set for male speakers. With variable shifting inaccuracies will arise only under certain conditions of F0 movement.

5.6.2 Perturbation Algorithm

The Hiller (1985) Perturbation Measurement System was comprised of three notable features:-

1) Input data consisting of F0 and A0 contours extracted from samples of connected speech processed using the parallel processing pitch detection system described in Section 5.6.1.
2) Perturbation measurement based on excursions (i.e. deviations) of the individual input values from a local smoothed trend line of samples - a non-linear smoothing consisting of a 5-point running-median plus 3-point Hanning filter produces the smoothed trend line.

3) Output parameters comprised of long-term distributional measures of frequency and amplitude perturbation based on the excursions from the smoothed trend line.

The measurement system also produced long-term distributional measures of the fundamental frequency based on the smoothed trend line of F0 values.

The parallel processing pitch detection algorithm produced from samples of connected speech two outputs consisting of the inverse values of the detected pitch periods in units of Hz and an A0 contour whose sample values were based on peak amplitudes derived from each detected pitch period represented in the F0 contour.

The trendline underlying the raw F0 curve was constructed by a non-linear smoother as presented by Rabiner, Sambur & Schmitdt (1975). A non-linear smoother has advantages over more conventional linear smoothers (e.g. Running average) which tend to smear sharp discontinuities present in speech signals as well as being affected by gross errors in contour. A non-linear smoother was used so as to preserve realistic discontinuities present in F0 contours (i.e. transitions from voiced to voiceless states and vice-versa), while smoothing micro-perturbatory roughness and gross pitch period estimation errors. The non-linear smoother implemented was a combination of running median filter plus a Hanning window. This was used for extracting both the F0 and A0 trendlines.

The application of the non-linear smoother to a given F0 and A0 contour produces a trendline from which perturbations can be measured for connected speech samples. The basic unit of perturbation measurement is defined as the difference between an input unsmoothed value of F0 or A0 and its equivalent smoothed value along the trend line derived by the non-linear smoother. This basic unit of perturbation is called an Excursion. This unit is measured in relation to a smoothed trendline in order that slow-moving modulations and intonational movements of F0 and A0 are excluded from contributing to the estimation of perturbation parameters. The use of excursion measures from a smoothed trendline have

been presented in Koike (1973), Kitajima, Tanabe & Isshiki (1975), Davis (1976), Kitajima & Gould (1976), Koike, Takahashi & Calcaterra (1977) and Laver, Hiller & Hanson (1982).

An excursion was derived for each output of the non-linear smoother and defined as the difference between the unsmoothed input value of F0 or A0 and its equivalent smoothed sample. Each excursion of F0 was stored in four formats:-

1) signed excursion in Hz, the difference between input and smoothed F0 in units of Hz with algebraic sign retained.
2) signed excursion in percent (SE%)-- the ratio of the signed excursion in Hz to its equivalent smoothed F0 value multiplied by 100.
3) magnitude of excursion in Hz - the absolute value of the signed excursion in Hz
4) Magnitude excursion in percent (ME%) - the absolute value of the signed excursion in percent.

The four formats were also produced for a given input A0 contour with the appropriate amplitude values.

Two sets of long-term intonation and perturbation parameters were based on the output of the simple non-linear smoother. The symbols used for these parameters are those adopted by Laver, Hiller, Mackenzie & Rooney (1986) and Beck (1988). The parameters describe the magnitude, distribution, and frequency of micro-perturbatory behaviour for each F0 contour in the speech sample. The long-term intonational parameters produced are described below.

F0-AV is the average fundamental frequency in units of Hz for all non-zero F0 samples in the smoothed contour. The use of the smoothed contour meant that values outside the pre-set limits of 40-240Hz (for males) were eliminated from the original data pool. This was to limit the effects of gross measures in F0 produced by the pitch detection algorithm.

F0-DEV is the standard deviation of the fundamental frequency in units of Hz for all non-zero values in the smoothed F0 contour. This parameter represents the typical range of F0 produced by each speaker for a given speaking task. The pre-set limits for acceptable values are also used for producing this parameter. Intonational parameters were included as it is possible that some pathological speakers may be able to maintain normal perturbation whilst deviating from normal in their range of F0.

The long-term perturbation parameters produced are described below.

J-AVEX/S-AVEX. These parameters represent the average excursion (AVEX) as found for the magnitude excursions in percent of F0 (jitter) and A0 (shimmer) contours, respectively. The magnitude of the excursion is used as the input to this parameter in order that all non-zero values of F0 and A0 will produce a non-zero mean value.

J-DEVEX/S-DEVEX. These parameters represent the standard deviation of the excursions (DEVEX) as derived for the SE% for F0 (jitter) A0 (shimmer) contours, respectively. In this case, signed excursions are input since it is expected that a normal-like distribution of excursions will occur around a mean value close to zero. If there is a tendency towards the production of larger than average excursions of F0 and A0 then these should be revealed as a large spread of excursion values as seen in the DEVEX measures for jitter and shimmer.

J-RATEX/S-RATEX. These parameters represent the rate of excursions (RATEX) found in F0 and A0 contour respectively. RATEX is adapted from the *Pitch Perturbation Quotient* (PPQ) of Koike, Takahashi & Calcaterra (1977), and uses a non-linear smoother instead of the moving average approach used to calculate PPQ. RATEX is the percentage of points in a given contour where a magnitude of excursion in percent is greater than a pre-set threshold. The pre-set threshold is used to quantify the number of significant perturbations in any given voice sample. The pre-set threshold was set to 3%, because even in the healthiest voices, uttering a sustained monotone vowel, the successive pitch periods typically show approximately 2% frequency jitter in a normal distribution (Hanson

1978). The 3% threshold enables one to discount this factor from the J-RATEX measure. The choice of threshold for S-RATEX is less certain since equivalent statistical statements are not available for shimmer. A 3% threshold was also chosen for S-RATEX.

J-DPF/S-DPF. This parameter is the Directional Perturbation Factor (DPF) which has been adapted from the work of Hecker & Kreul (1971). The DPF is based on changes of direction in values within F0 and A0 contours. DPF counts the number of times there is a change in algebraic sign when a difference is measured between adjacent input contour values. Therefore, roughness in an F0 or A0 contour may be evaluated as small directional changes in the contour. DPF totals the number of directional changes and this total is divided by the total number of possible directional changes within a given contour and multiplied by 100 to produce percentage values of J-DPF and S-DPF. The DPF parameter has been modified by including the requirement that a magnitude difference of greater than 3% must occur for any given directional change before it is included in the total DPF count. This threshold is another attempt to exclude the normal distribution of F0 directional changes from perturbation parameters. Any zero in the F0 contour is not evaluated by the DPF algorithm and this includes onsets/offsets of voicing as well as short discontinuities within the contour.

5.7 LARYNGEAL PATHOLOGY DETECTION STUDIES USING ACOUSTIC PARAMETERS

As mentioned in Section 5.6 the data used in this thesis was made available to the Research Group in which the Boltzmann Machine studies were carried out. Sets of this data have also been used in other pathological/healthy detection studies. This section briefly describes some of these studies.

Laver, Hiller, Mackenzie and Rooney (1986) studied various group separation techniques. Initially, parameter values were transformed to Z-scores and expressed as multiples of the control standard deviation from the control group mean. Given that two standard deviations on any one parameter should include

approximately 90% to 95% of control subjects (assuming normal distribution), any subject whose score on a given parameter deviates from the control group mean by more than two standard deviations was considered at risk of pathology. On this basis, no single parameter of the ten was shown to distinguish between the two groups sufficiently for the purposes of screening. The use of one F0 parameter and one perturbation parameter was shown to be more successful. The parameters F0-AV and S-DPF were analysed to give an ellipse (at the 2 SD level) indicating the covariance between the parameters. The boundary of the ellipse formed the screening threshold for the detection of pathology. Using this method, 90.1% of pathological males were outside the ellipse and classified as pathological. However, 9.5% of control males were also outside the ellipse and registered as false positives. The control group was comprised of 63 male speakers and the pathological group was comprised of 55 male speakers with diagnosed laryngeal pathologies.

Laver, Hiller, Mackenzie and Rooney (1986) also describe experiments using linear discriminant analysis. This is a statistical technique for discriminating between two (or more) groups on the basis of several parameters simultaneously. A discriminant function is derived by weighting and combining the parameters in such a way that the groups will be maximally separated by their member's score on this function. Using discriminant functions derived from all ten acoustic parameters the results in Table 5.2 were reported.

	Classified Correctly	Classified Incorrectly
Pathological	85.5%	14.5%
Control	92.1%	7.9%

Table 5.2 Group separation results using linear discriminant functions.

The results of this discriminant analysis need to be treated with caution. Linear discriminant analysis assumes that the data show a multivariate normal

distribution, but given the heterogeneous composition of the pathological group it is likely that this assumption was seriously violated. However, the technique is quite robust in the face of such violations. A more serious problem is that the groups are rather small.

Laver, Hiller, Mackenzie and Rooney (1986) also analysed the weighting coefficients derived from the linear discriminant analysis procedure and were able to conclude that all ten parameters are not equally useful. Some do not separate the groups very well, while others are redundant by virtue of their high correlation with those that do. They concluded that S-DPF was the most important contributor to both the male and female discriminant functions.

Using a larger data set, Beck (1988) reports, using linear discriminant analysis with all 10 parameters, 87.5% correct classification of pathological voices and 8.4% of control speakers classified as pathological. The control group in this study comprised 83 male speakers and the pathology group was comprised of 56 pathological male voices.

Beck (1988) also reports results using 9 acoustic parameters (excluding S-DEVEX because of its non-normal distribution) with maximum likelihood techniques. Using this approach 87.5% of pathological speakers were correctly classified, and 14.3% of control speakers were classified as belonging to the pathological group. Beck (1988) concluded that discriminant analysis is probably the best of these screening options.

All these techniques have been applied to within group testing and thus the classification rates obtained cannot safely be asserted to be necessarily predictive of future success in classifying another set of subjects with the same function.

5.8 SPEAKERS USED FOR PERTURBATION ANALYSIS

The data made available to the Research Group in which the Boltzmann Machine studies were carried out was comprised of two groups of native British speakers.

One group consisted of 53 male adult speakers who showed a variety of laryngeal pathologies. All had undergone laryngeal examination, and details of laryngeal status were obtained from their medical records. The pathologies detected are listed in Table 5.3

Type of Pathology	Males
DISORDERS OF THE LIGAMENTAL AREA	
Epithelial disorders	
Hyperplasia	1
Keratosis	2
Squamous carcinoma	9
Verrucous carcinoma	1
Papillomata	2
Lamina Propria disorders	
Polyps, nodules	10
Acute laryngitis	2
Chronic laryngitis	3
Oedema	1
Mild redness, thickening	6
DISORDERS OF THE CARTILAGINOUS AREA	
Contact ulcer	4
Cyst	1
VOCAL FOLD PALSIES	11

Table 5.3 Laryngeal disorders diagnosed in the pathological subject group.

The control group was comprised of 78 male speakers who had no reported history of speech or voice disorder. It was not feasible to subject this group to a laryngeal examination. The age range for the control speakers was 18 - 71, and the age range for the pathological speakers was 23 - 82.

The acoustic parameter data made available for the Boltzmann Machine studies was obtained in the following way. A high quality tape recording of each speaker was made as he read the first two paragraphs of the *The Rainbow Passage,* (Fairbanks 1960). Prior to the recording, the speakers had familiarised themselves with the passage, and were asked to read out loud at a comfortable level, using their habitual voice. Forty seconds of each recording was digitised and stored on computer magnetic tape for intonation and perturbation analysis as described in Section 5.6.

This chapter has described the methods by which the intonation and perturbation parameters were obtained from speech recordings of 78 healthy adult male speakers and 53 adult male speakers with known laryngeal pathologies. These parameters have been used to investigate two screening applications using Boltzmann Machine models. The first of these is for discriminating between healthy and pathological speakers, and is presented in Chapter 6. The second application is for discriminating between groups of pathologies present, and is presented in Chapter 7.

6 SEPARATION OF CONTROL AND PATHOLOGICAL GROUPS

6 SEPARATION OF CONTROL AND PATHOLOGICAL GROUPS

6.1 INTRODUCTION

The experiments presented in this chapter make use of the intonational and perturbation parameters made available to the Research Group in which the Boltzmann Machine studies were made. The details concerning the method by which these intonation and perturbation parameters were produced has been described in the previous chapter. Each experiment described in the current chapter is only concerned with the separation of pathological and healthy speakers. The aim of these experiments is to investigate a neuromorphic approach to the screening of populations for voice pathologies.

The subject groups used to provide the intonation and perturbation parameters for the experiments comprised 53 adult male speakers, with diagnosed voice pathologies, and 78 adult male control speakers, with no history of voice disorders. This data was subdivided into a training set of 39 control and 28 pathological speakers and into a test set of 39 control and 25 pathological speakers. A smaller training set was also produced which had only 10 control and 10 pathological speakers. The testing data set was larger, with 68 control and 43 pathological speakers. The function of this smaller training set was to investigate the effect on network generalisation using a reduced set of training data. As mentioned above however the overall objectives of the experiments discussed in this chapter are to differentiate between healthy and pathological speakers only.

Before the intonation and perturbation parameter data could be applied to the Boltzmann Machine networks used in the various experiments it was necessary to transform the data from floating point format into a binary format. Three types of binary format were investigated.

The networks were trained using examples from the training set, and then tested to see whether they could correctly identify the control and pathological subjects in new examples from the test set. The network's generalisation to the test set

was measured by counting the number of cases that the network classified correctly. A severe problem with this approach is that it is liable to require a large amount of accurately labelled training data to give good generalisation. A network may require a large number of weights to allow it to capture the relevant structure, and so the training data must contain a large number of samples in order to constrain these weights sufficiently to give good generalisation. If the number of samples in the training data does not significantly exceed the number of degrees of freedom in the model, the network can use a form of *rote-learning* or *table-look up* to learn the training cases. In other words, it can find a setting of the weights that give the correct output for all the training examples, without capturing the underlying regularities of the task. Unfortunately, each training example only contains a few bits of information, because for a supervised learning procedure, as used in Boltzmann Machine learning, the information in an example is the number of bits it takes to specify the correct *output.* So there is a severe practical problem to provision of sufficient training data. Results of Prager et al. (1986) confirm that the generalisation is poor when a general learning procedure is used to fit a model that has more degrees of freedom than there are samples in the training set.

For the Boltzmann Machine with binary inputs, the number of bits in a Boltzmann Machine's input pattern directly affects the number of weights required in the network, which in turn affects the network's information capacity and hence its ability to learn and generalise from a given number of training cases. Because there were only 67 training patterns in the larger training set, each of which requires an output choice that can be specified by 1-bit, a network would need to learn 67-bits of information to perform the task by table look-up. Each weight appears (in the single-layer model) to be able to comfortably store approximately 1.5 bits of information, so a network with more than about 50 weights may have a tendency to memorise the training cases and thus fail to generalise to the test set. Such limitations on training data would also apply to other supervised network learning procedures, such as the Multi-layer Perceptron. However, this general information capacity argument does not take into consideration specific

properties of the problem domain. In addition it is not clear how much an oversized network would suffer in generalisation. Elman & Zipser (1987) have reported that generalisation performance can be improved by introducing certain kinds of noise into the training patterns and hence expanding the data set. This tends to result in greater error during the learning phase but better generalisation. A noisy pattern clamping technique, described later, was developed and its effect on generalisation investigated.

One reason why so much data may be required is that there is no prior knowledge built into the network. In other words, the network has no prior expectations that adjacent parameters or quantisation levels are likely to be more relevant to each other than non-adjacent ones. This lack of prior expectation means that the learning is searching a huge space of possible feature detectors, most of which are almost certainly useless. It is possible that the training data could be reduced by using an architecture that omits irrelevant connections, or by starting with reasonable hand-coded feature detectors.

A network can be forced to develop localised feature detectors by restricting its connectivity in such a way that each hidden unit receives input from a receptive-field that only covers a small region of the input. (Details of how Boltzmann Machine connectivity can be restricted are to be found in Section 4.4.2). Because the total number of its weights is a relatively small multiple of the number of its hidden units, a network with small-receptive-fields can possess a rich inventory of hidden unit codes to represent subtleties of the input, without being burdened by an excessive number of free parameters that would allow the network to learn its training set by rote. This assumes that the capacity limitation that forces good generalisation is the number of weights.

Initially, experiments with single-layer networks (in which the input units are directly connected to the output units) were conducted to provided a performance baseline. Networks having hidden units, but no connections between input and output units were then investigated. In addition, experiments were conducted with these networks using *noisy* data to clamp the input units during training.

Experiments were also conducted using network architectures with small-receptive-fields. A natural partition for the input data was to divide it into three receptive fields with each field corresponding to the intonation, jitter or shimmer parameters. Each hidden unit was connected to either the intonation, jitter or shimmer parameter data present at the input units. To permit the detection of multiple features in each set of parameters, the network had up to 2 hidden units connected to each of the three receptive fields. No connections were allowed between hidden units, and the two output units were connected to each other and the hidden units. There were no connections between the input units and the output units.

In another variation of the restricted receptive-field network architecture, each hidden unit was only allowed to make connections to the input units corresponding to one of the intonation or perturbation parameters. Again, to permit the detection of multiple features up to two hidden units were allowed for each of the ten receptive fields. Limitations on the compute time available meant that it was not possible to investigate additional numbers of hidden units for each receptive fields, or indeed further different sizes of receptive fields. A restricted receptive field network with two hidden units per field would thus have approximately the same number of weights as the networks with two hidden units (but no connections between input and output units) mentioned above.

Finally, experiments were performed using as input data to networks a number of different subsets of the ten intonation and perturbation parameters. The network architectures investigated included those with, and without hidden units.

The power of any pattern-recognition system lies in its ability to deal with noise or distortion. That is, after being trained on representative patterns of a class, the system is able to recognise all other patterns of that same equivalence class. In the supervised learning for the Boltzmann Machine, the output is an estimated value of the class index. Consider a Boltzmann Machine that has been trained to yield an output of 1 for all of the pathological patterns and of 0 for all non-pathological (in this case the control) patterns. An unknown pattern is

clamped over the input units and maintained while the network is annealed. Due to the stochastic nature of the operation of the Boltzmann Machine, the network is then run for a given length of time at thermal equilibrium to determine how often the output unit is either *on* or *off*. Thus, the probability of that particular pattern occurring is obtained. An actual test-set pattern may thus occur with a probability of 0.85 and would be considered to be of the pathological class. The basic act is that of estimation, although the final result is clothed in the guise of classification.

For many of the networks, a *map* of the weights learned is presented. Each unit in the network is identified by a square outline. The input units are presented as an array (see Figure 6.1 for an example of 10 by 12 units). Each of the columns, representing one of the ten parameters, is labelled with an identification number (see Table 6.1). The quantisation levels are stacked in the column with lowest values at the base of the column, and highest values at the top of the column.

A positive weight is represented by a shaded or black square within the unit square. This represents an excitatory link. A negative weight is represented by a square outline within the unit square. This represents an inhibitory link. A zero weight or a weight that is below a given weight value threshold is represented by the unit square being blank.

Some of the weight maps presented have a weight threshold value of 5% of the maximum absolute weight value. Any weights below this threshold are not displayed because primarily the reprographic procedure used makes it difficult to distinguish whether very small weights have positive or negative values. It is also expected that very small weights will have an inconsequential effect on attempts made to interpret the patterns of the weights learned.

For some of the weight maps presented, the output patterns used for training set the left-hand output unit *on*, and the right-hand output unit *off* for a pathological subject. A control subject was indicated by training with the left-hand unit set to *off,* and the right hand unit set *on.*

6.2 DATA PREPARATION

The corpus of data has been described in Section 5.8 and comprises one pathological group of 53 adult male speakers and one control group of 78 adult male speakers. The speakers in the pathological group all showed a variety of laryngeal disorders and all had undergone laryngeal examination. The control group speakers had no reported history of speech or voice disorder, but had not undergone laryngeal examination. For each of the speakers ten intonation and perturbation parameters were obtained using the perturbation measurement system described in Section 5.6, see Table 6.1.

#	PARAMETER	DESCRIPTION
1	F0-AV	Mean Fundamental frequency
2	F0-DEV	Standard deviation of FO
3	J-DEVEX	Jitter- standard deviation of excursions
4	J-AVEX	Jitter- average excursions in % of FO
5	J-RATEX	Jitter- rate of excursions
6	J-DPF	Jitter- directional perturbation factor
7	S-DEVEX	Shimmer- standard deviation of excursions
8	S-AVEX	Shimmer- average excursions in % of A0
9	S-RATEX	Shimmer- rate of excursions
10	S-DPF	Shimmer- directional perturbation factor

Table 6.1 Intonation and perturbation parameters.

All the parameter values were in floating point format and had to be transformed into a binary format for presentation to the Boltzmann Machines in the simulator. To obtain the binary representations required, the maximum global range for

each of the intonation and perturbation parameters was linearly quantised into an integer number of *buckets* or quantisation levels. The smallest number of quantisation levels required using all ten parameters, from the total of 131 speakers, to ensure that the binary representation of each speaker's data remained unique was six. Thus quantisation levels of six and twelve, which provided twice the minimum resolution, were used. Three binary formats of this data were employed. The first being the *dot* method. In this case the bucket corresponding to the value of the parameter was set to the value 1 and all the remaining buckets were set to the value 0. The second approach is referred to as the *slide* method. It is similar to the *dot* method but in addition the two buckets immediately adjacent to the bucket corresponding to the parameter value are also set to 1, with the remaining buckets set to zero as before. The final method is referred to as the *bar* method and provides data in a format that is similar to a bar-graph display. That is, all the buckets below and including the bucket corresponding to the parameter value are set to the value 1, and the remaining buckets are set to the value 0. Table 6.2 indicates these three binary representations of the parameter data.

The slide format has been described as a coarse coding method by Hinton (1984) and Kanerva (1984). The object behind this approach is that to encode features accurately using as few units as possible, it is best to use units that are very coarsely aligned, so that each feature activates many different units and each unit is activated by many different features. A specific feature is then encoded by a pattern of activity in many units rather than by a single active unit, so coarse coding is a form of distributed representation.

These formats essentially require that a separate code be used for each of the quantisation levels. Alternatively, each quantisation level could be represented by a binary number, which would have reduced the number of bits in the input patterns. It would however have made interpretation of the weight maps produced particularly difficult. Furthermore, informal experiments showed that learning times were much longer for this method and so this representation was not used.

Coarse coding is only effective when the features that must be represented are relatively sparse. If many feature-points are crowded together, each receptive field will contain many features and the activity pattern in coarse coded units will not discriminate between many alternative combinations of feature points. There is thus a resolution/accuracy trade-off. It may thus be possible to use fewer quantisation levels, to achieve a given level of accuracy, provided the features to be represented are relatively sparse.

FORMAT	REPRESENTATION
DOT	00010000
SLIDE	00111000
BAR	11110000

Table 6.2 Examples of the three binary formats for a given parameter data value.

The range of values for parameter 7 (S-DEVEX) was distorted by one speaker having a particularly large value for the rate of excursions found in his F0 contour. This particular value was disregarded and the next largest value was used to determine the overall range for S-DEVEX. The parameter value in excess of the maximum range value was catered for by saturating to the highest quantisation level available.

A computer program was written, which would process the raw parameter data and provide an output file containing the parameters selected, the desired binary format, and the number of quantisation levels required. It was also established that the smallest number of quantisation levels required using all ten parameters, from the total of 131 speakers, to ensure that the binary representation of each speaker's data remained unique was six. This meant that patterns in each of the control and pathology groups were not duplicated, so that the number of different patterns applied was known. Uniqueness in each of the healthy and pathological groups is not particularly relevant in the healthy/pathology discrimination task

as the speakers are only being classified into these two groups. Quantisation levels below six also meant that patterns overlapped between the control and pathology groups, which is clearly undesirable. Uniqueness in the pathology group is more important when considering the classification of pathological speakers into groups of laryngeal disorders.

6.3 TRAINING AND TEST GROUPS

For purposes of training and testing the various networks the data was divided into two training and test sets. The smallest training set comprised 10 pathological speakers and 10 randomly selected control speakers. The pathologies were chosen to be representative of all the pathologies reported in the data set and are indicated in Table 6.3. The test sets were comprised of the remaining control and pathological subjects.

The second training set comprised 39 randomly selected control speakers and 28 pathological speakers. The pathologies present in the training and test sets are indicated in Table 6.4.

Type of Pathology	TRAIN	TEST
DISORDERS OF THE LIGAMENTAL AREA		
Epithelial disorders		
Hyperplasia	0	1
Keratosis	1	1
Squamous carcinoma	2	7
Verrucous carcinoma	0	1
Papillomata	0	2
Lamina Propria disorders		
Polyps, nodules	2	8
Acute laryngitis	0	2
Chronic laryngitis	0	3
Oedema	0	1
Mild redness, thickening	1	5
DISORDERS OF THE CARTILAGINOUS AREA		
Contact ulcer	2	2
Cyst	0	1
VOCAL FOLD PALSIES	2	9

Table 6.3 Laryngeal disorders diagnosed in the small pathological training and test data sets.

Type of Pathology	TRAIN	TEST
DISORDERS OF THE LIGAMENTAL AREA		
Epithelial disorders		
Hyperplasia	1	0
Keratosis	1	1
Squamous carcinoma	4	5
Verrucous carcinoma	1	0
Papillomata	1	1
Lamina Propria disorders		
Polyps, nodules	4	6
Acute laryngitis	1	1
Chronic laryngitis	2	1
Oedema	1	0
Mild redness, thickening	3	3
DISORDERS OF THE CARTILAGINOUS AREA		
Contact ulcer	2	2
Cyst	1	0
VOCAL FOLD PALSIES	6	5

Table 6.4 Laryngeal disorders diagnosed in the large pathological training and test data sets.

6.4 SIMULATIONS

6.4.1 No Hidden Units

Much of the literature detailing the application of Boltzmann Machines and other networks with hidden units describes results obtained using large numbers of hidden units. In many instances, the use of hidden units is justified in terms of providing specific feature detectors or in aiding generalisation. The performance of these networks with no hidden units is not dealt with. To provide a baseline measure with which to compare the performance for various topologies of network using the Boltzmann Machine learning algorithm a number of experiments were carried out using no hidden units. These were performed with the simulator by making the update rule deterministic rather than stochastic. This was achieved by setting the temperature parameter to zero. As no annealing is required, due to this type of network having a concave energy space, the length of the annealing schedule was set to unity, to allow a single update per learning cycle. *Statistics* were collected for one unit of time and the simulator was operated in the synchronous mode. The simulator is thus allowing Hebbian learning to be performed on the network. The weights were updated according to the scheme detailed in Equation 4.7. Thus, the weights were changed by a fixed step which decreased in magnitude as a function of the mean error as the network converged. The weight polarity is determined by the sign of the difference between the probability that a pair of units is on together in the clamping and free phases.

Two baseline networks with no hidden units were initially investigated. The first network had 60 input units each connected to 2 output units for use with 6-bit quantised data, and the other comprised 120 input units and 2 output units. In each case the two output units were not connected together. The two required output classes were coded so that one unit *on* represented control speakers, and the other unit on would represent pathological speakers. A single output could have been used instead, but using the two output units in this way meant that the learning problem presented to the simulator was in fact equivalent to

two networks, with 60/120 input units and one output unit each. Thus two networks were being trained with the weights learned in each being virtually identical except that in one, connections to features indicative of pathological voices have positive value weights, whereas in the other, connections indicative of pathological voices have negative value weights.

For each of the two sizes of network, experiments were initially conducted using both the training and test sets of data to train the networks with all 10 parameters. Experiments were then conducted using the small and large training sets for these networks, again using all 10 parameters. Noisy clamping of the training data was also investigated, whereby the quantisation level for each parameter was perturbed up or down by one level with a 5%, 10%, 15% or 20% probability.

6.4.2 Hidden Units

To investigate the effect of hidden units experiments were conducted using networks comprising 2, 4 and 8 hidden units, 60/120 input units and 2 output units. The architecture of these networks was constrained so that there were no direct connections from the input units to the output units, and also no connections between the hidden units. The objectives of these experiments was to examine the feature detectors formed using hidden units and to compare generalisation results with those of the baseline networks. The topological variations possible with the simulator for networks containing hidden units and one receptive field are detailed in Section 4.4.2

For each of the two networks, experiments were conducted using all the parameters from the large training set. Experiments were also conducted, again using all 10 parameters, whereby the data for each parameter was perturbed up or down the scale by 1-bit with a 5%, 10%, 15% or 20% probability.

During the training procedure, all the networks were annealed using a logarithmic schedule based on a starting *temperature* of three times the mean hidden unit energy. The end temperature was approximately one third of the starting

temperature. Co-occurrence statistics were collected for between 6 and 40 units of time depending on the mean output error of the networks, (see section 4.4.7). The weights were adapted according to the scheme of Equation 4.7.

6.4.3 Intonation, Jitter and Shimmer Receptive Field Networks

To determine the effect of restricting network connectivity so that each hidden unit covers only a small region of the input units, the input array was divided into three regions. These three regions were made to correspond to parameters of intonation, jitter and shimmer. Each of the hidden units was connected to either the frequency, jitter or shimmer parameter data present at the input units. To permit the detection of multiple features for each set of parameters, the network had up to 2 hidden units allowed per receptive field. No connections were allowed between the hidden units, and the two output units were connected to each other and to the hidden units. There were no connections between the input units and the output units. An example of this type of structure is given in Figure 4.5.

For each of the networks the large training set was used, with bit perturbation also applied. A similar training schedule was used to that described for the previous section.

6.4.4 Intonation and Perturbation Parameter Receptive Field Networks

To determine the effect of using a receptive field for each of the 10 intonation and perturbation parameters, a small receptive field network was simulated which had each hidden unit connected to one of these parameters at the input units. To permit the detection of multiple features in each set of parameters up to two hidden units connected to each of the ten receptive fields. No connections were allowed between hidden units, and the two output units were connected to each other and the hidden units. There were also no connections between the input units and the output units. An example of this type of structure is given in Figure

4.6. An annealing schedule similar to that described for the previous section was implemented for networks trained on the large training set of 39 control and 28 pathological speakers.

6.4.5 Selected Parameter Networks

Results from experiments undertaken by Laver, Hiller, Mackenzie & Rooney (1986) and Beck (1988) have suggested that there is a degree of redundant information in the 10 parameters. Despite these indications, no optimal parameter set has been presented due essentially to the small data set available. Although it was not feasible to attempt the selection of an optimal parameter set, it was decided to investigate the performance of networks with subgroups of parameters selected from observations of weight maps produced by networks without hidden units. Experiments were conducted for networks with and without hidden units, for a number of parameter subsets. Each of these subsets was comprised of four parameters. The parameters within each of the four subsets in the networks are shown in Table 6.5.

SET 1	SET 2	SET 3	SET 4
F0-AV (1)	F0-AV (1)	F0-AV (1)	F0-AV (1)
J-DPF (6)	J-RATEX (5)	J-AVEX (4)	J-AVEX (4)
S-AVEX (8)	S-AVEX (8)	S-AVEX (8)	S-RATEX (9)
S-DPF (10)	S-DPF (10)	S-DPF (10)	S-DPF (10)

Table 6.5 Parameter subsets

The parameters were selected following observations of a number of weight maps for networks without hidden units. From these it was not possible to determine an optimum parameter set, but those parameters that had strong connections over the range of the parameter were interpreted as being more useful than those which only utilised a small number of the parameter units. The validity of this approach rests on the assumption that the weights have not become very large

as a way of improving the performance of patterns already learned, but have been kept suitably small to allow equilibrium to be reached. Although no tests of this assumption were made it appears to a certain extent to be verified in the results from Sections 6.5.1 and 6.5.2 which indicate the presence of higher than normal levels of perturbation for pathological speakers.

For all the networks simulated, training was performed with the 39 control and 28 pathology speaker training set. With quantisation to six levels it was found for each of the four sub-sets that each pathological pattern was different from the control patterns. However, for sets 2 and 4 eighteen of the 39 control training speaker patterns were identical and for sets 1 and 3 fifteen of the 39 control speaker patterns were identical. As discussed in Section 6.2 uniqueness is not important in discriminating between the control and pathology group. However, the fact that the number of different control patterns presented to the selected parameter networks was different from the number obtained with all ten parameters, means that direct comparisons cannot be made between these experiments.

Initially, networks with no hidden units, two output units and 24/48 input units were trained using all of the three data formats. Experiments were also conducted using networks with 24 input units and 2 hidden units. These networks had connections between their input and output units, and each hidden unit was also connected to each input unit and to each output unit. No connections between hidden units and no connections between output units were allowed.

6.5 RESULTS

6.5.1 No Hidden Units

Using all the patterns from the training and testing sets to train the 60 and 120 input unit networks it was found that with 6 quantisation levels it was not possible for the network to correctly learn all the patterns. The dot format data allowed

the network to correctly learn 88.5% of the patterns, wheres the slide format allowed 54.9% and the bar format 60.3% of patterns to be correctly learned. With 12 quantisation levels the networks were able to correctly learn all of the patterns.

Various sizes of network using different numbers of coefficients for each perturbation parameter were then trained with both the training and test tokens. The objective of this experiment was to determine approximately what the storage capacity of this net was using all of the control and pathological subjects data available, and thus approximately how much information could be encoded by each weight using this data.

Using the dot data format it was possible to store 131 patterns using 80 weights (i.e. 8 coefficients per parameter). This corresponds to approximately to 1.6 bits of information per weight. For the slide representation 9 coefficients per parameter (90 weights) were required giving approximately 1.5-bits per weight. For the bar format 10 coefficients per parameter (100 weights) were required giving approximately 1.3-bits per weight.

Both the slide and bar representations of the training data were too coarse to allow the network to correctly distinguish between the two groups when the resolution was down to 8 levels or less per parameter. At 9 levels the slide representation was able provide 100% classification, but the bar representation required 10 levels before the same performance could be achieved. The degree of overlap for patterns is greater for the slide format than the dot format, and greater still for the bar format than the dot and slide formats. This overlap can be resolved by increasing the resolution of the parameters and thus increasing the number of input units.

It is expected that there is a degree of correlation between patterns within each of the control or pathological classes, and thus the number of bits stored per weight is probably higher than one would expect for an orthogonal data set. Lang, Waibel & Hinton (1990) mentions that as a rule of thumb for backpropagation

networks (according to Carnegie-Mellon University folklore) each weight is comfortably able to store approximately 1.5-bits of information. This appears to be broadly similar to the results obtained in these experiments.

The pattern of weights learned provides some indication as to which intonation and perturbation parameters are being used in differentiating between the control and pathological speakers.

Figures 6.1 to 6.3 depict maps of the weights learned for networks with 12 quantisation levels per parameter and one output unit. The training data comprised the total 131 patterns in the *dot, slide* and *bar* formats. A positive weight is represented by a black square within the grids and causes the output unit attached to it to become activated when a 1 is present in the component of the input pattern to which the weight is connected. A negative weight is represented by a white square within the network grids and is inhibitory. This causes the output unit to become deactivated when a 1 is present in the corresponding component of the input pattern. A zero weight is represented by a blank unit square within the grid causing the output unit to ignore the contents of the corresponding component of the input pattern.

The left hand map displays the weights learned when the state of 1 at the output determines a member of the pathological class. Thus the positive weights (in black) represent connections made to those features which allow the network to identify pathological speakers. The right hand map indicates the weights learned when a state of 0 at the output unit determines a member of the pathological class. Thus the negative weights (in black outline) represent connections made to those features which allow the network to identify pathological speakers. As would be expected the left-hand map values are an almost exact inverse of the right-hand map values.

Each column in the input array represents one of the parameters labelled 1 to 10 (see Table 6.1). The quantisation levels are stacked in the column with lowest values at the base of the column, and highest values at the top of the column. In Figures 6.1 to 6.3 a 5% threshold of the maximum absolute weight value has

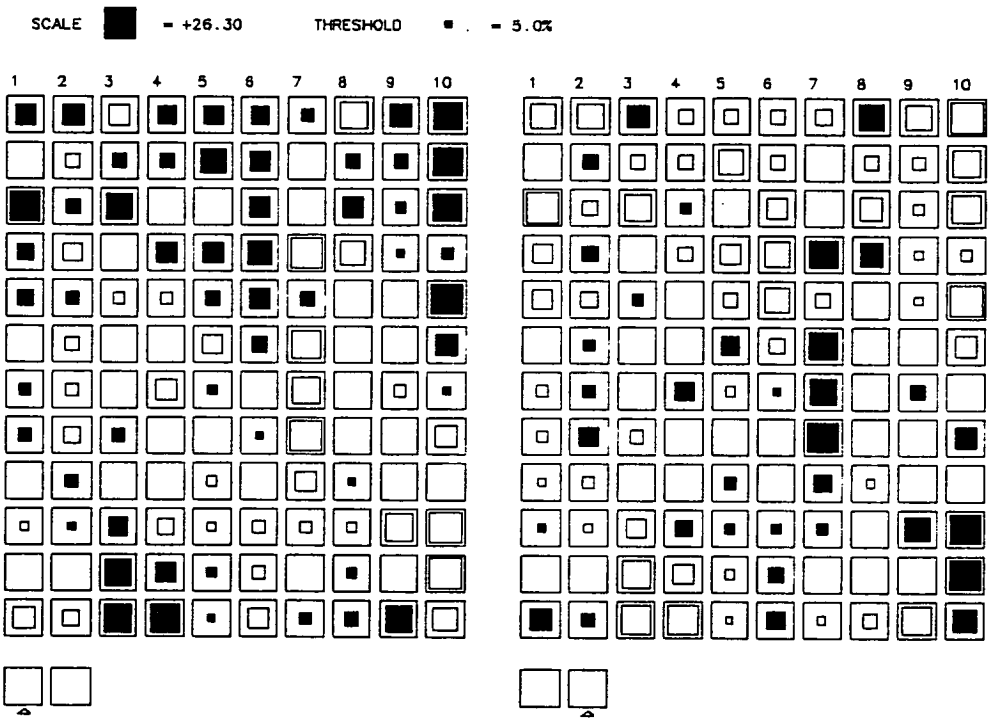

Figure 6.1 Weights learned by single-layer network. Learning data comprised all 131 control and pathological subject patterns in the dot format, with 12 quantisation levels per parameter.

been applied to remove very small weights from the maps. These weights will only have a nominal effect on the networks ability to distinguish between the two groups by only contributing very low levels of activation to the output units.

Figures 6.1 and 6.2 provide a clear indication that some pathological individuals were distinguished from the control group by virtue of having lower than *normal* levels of perturbation. This is particularly reflected in parameters 3 (J-DEVEX), 4 (J-AVEX) and 9 (S-RATEX). To a lesser extent parameters 5 (J-RATEX), 7 (S-DEVEX) and 8 (S-AVEX) also indicate use of lower than *normal* levels of perturbation. The number of large positive weights in the top four quantisation level positions of the left-hand side map indicates (for the perturbation parameters) that higher levels of perturbation play a significant role in identifying pathological speakers. For parameters 3 (J-DEVEX) and 8 (S-AVEX) there is an indication that relatively large levels of perturbation are present for speakers within the control group. This implies that either the control group may in fact contain speakers with laryngeal disorders, or that healthy speakers have high values of J-DEVEX and S-AVEX. Indeed, it is known that some of the control speakers were later diagnosed as pathological. Unfortunately, this happened well after the end of the data collection exercise and a detailed follow-up was never made possible.

The positive weights in the right-hand maps indicate positive connections made when an output of 1 represents the control group. The weights learned identifying the control speakers indicate that for control speakers, parameter 10 (S-DPF) and 6 (J-DPF) are lower than for the pathological speakers. Strong connections are also made to perturbation parameters 7 (S-DEVEX), 8 (S-AVEX) and 10 (S-DPF).

Most of the lesions in the pathological group involve some degree of mass increase, which would be expected to lower the F0-AV. However, the weight values for the intonational parameters indicate that F0-AV is higher than normal

in the pathological group. This suggests that many of the disorders involve an increase in stiffness, which might balance the mechanical consequences of mass increase.

Figure 6.2 depicts the weights learned using the training data in the *slide* format. At the 5% threshold level fewer weights appear to be used than with the network trained using data in the *dot* format. The use of the *slide* code means that the data presented to the network has a higher degree of overlap than for the data in the *dot* format. The number of iterations to convergence for the *slide* method was between about 5 and 10 times greater (depending on the number of quantisation levels for the parameters) than that for data in the *dot* representation.

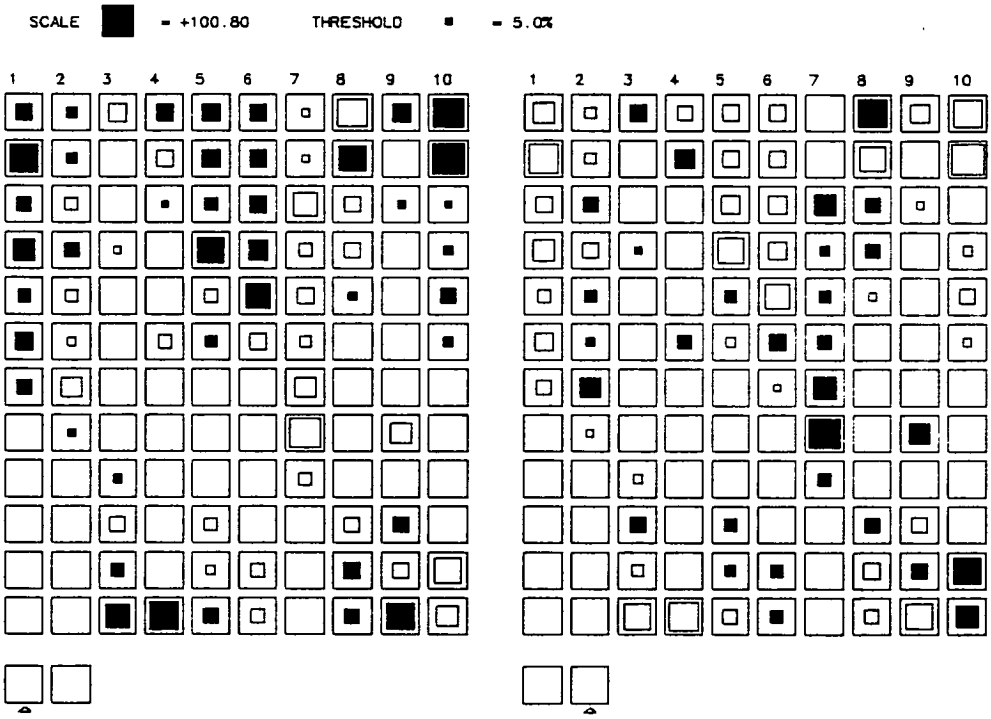

Figure 6.2 Weights learned by single-layer network. Learning data comprised all 131 control and pathological subject patterns in the slide format, with 12 quantisation levels per parameter.

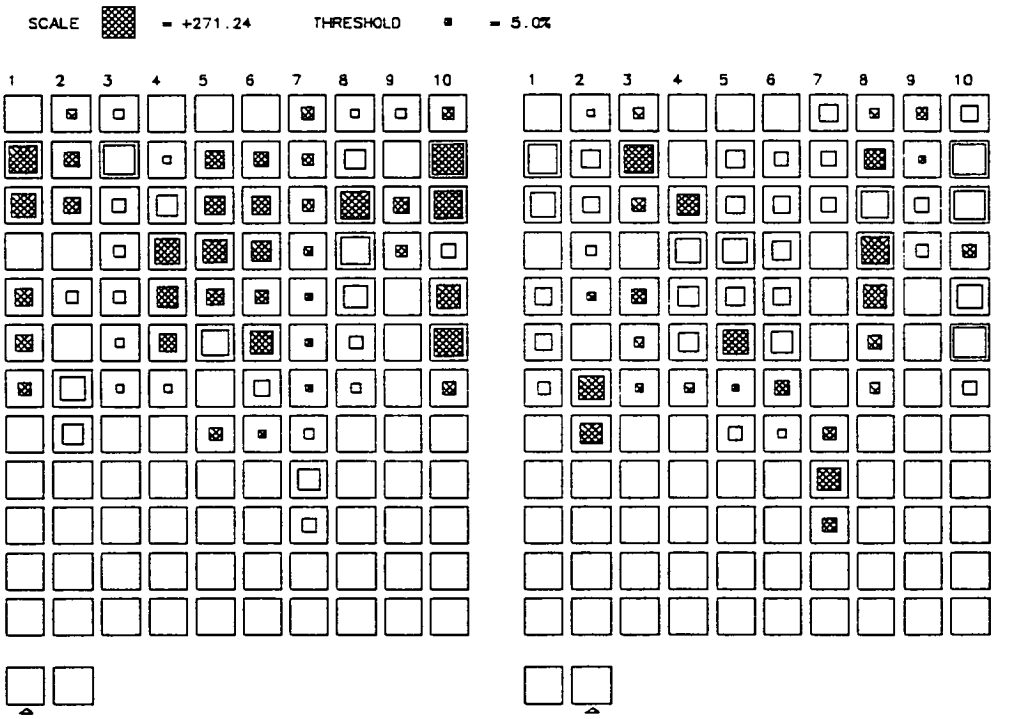

Figure 6.3 Weights learned by single-layer network. Learning data comprised all 131 control and pathological subject patterns in the bar format, with 12 quantisation levels per parameter.

Figure 6.3 depicts the weights learned with the data in the *bar* format. With this representation it appears that the presence of perturbation parameter values lower than normal are not being used in determining which group a speaker belongs to. An explanation for this is that different features in the training set are being used to discriminate between the two groups, and that with the *bar* representation, because the presence of a 1 in the training pattern at the lower quantisation values is highly likely due to the large degree of overlap, it is not possible for the network to utilise the occasional absence of 1's from the corresponding training pattern. Learning time for the *bar* method was between about 15 and 25 times longer (depending on resolution) than that for data in the dot format.

Ignoring the lower four quantisation levels, parameters with weight patterns differing substantially from the dot and slide representation models are 3 (J-DEVEX), 4 (J-AVEX) and 7 (S-DEVEX). The *bar* format does however confirms that pathological voices have relatively high levels of perturbation and intonational parameters.

Using the training data comprising of 39 control speakers and 28 pathological speakers it was possible for a simple network using a resolution of 6 quantisation levels for each of the ten parameters to correctly learn all the training patterns. In addition, networks were trained using noisy data. This was obtained by perturbing each parameter value up or down one quantisation level with a probability of 5%, 10%, 15% or 20%. There was a 50% probability that the move would be *up* one level. Overall, the best generalisation with the test data set of 39 controls and 25 pathological voices for the *dot* format was 89.7% of control speakers correctly classified as control speakers and 68.0% of pathological speakers correctly classified as pathological speakers. The network trained using data in the *slide* format gave no improvement with results at 87.2% and 68.0% respectively. The *bar* format however gave much better results for the pathological group with 84.0% correctly classified, and 84.6% for control

speakers correctly classified. The results at different noise levels varied substantially, and it was not always possible for the networks to correctly identify the base training patterns.

For parameters coded to just 6 levels, a perturbation of one level in either direction presents a relatively crude change in pattern. The number of controls correctly classified as controls decreased slightly as noise levels increase for the dot format. The number of controls classified as pathologicals increases slightly for the dot, slide and bar format from the results with no noise. Indeed the changes in classification results for the dot format were much smoother than those for the slide and bar formats. This is because there is greater pattern overlap for slide and bar formats.

The results for the networks trained with parameter data for the 39 control and 28 pathological quantised to 12-bits are shown in graphical form in Figures 6.4 to 6.6. The x-axis represents the percentage probability that a parameter has its value perturbed up or down by one quantisation level. The y-axis indicates the percentage of pathology subjects classified as (1) pathological; (2) controls, and the percentage of control patterns classified as (3) controls and (4) pathological. In an ideal classification system curves (1) and (3) would be horizontal lines at 100% and curves (2) and (4) would be horizontal lines at 0%. Thus, the closer curves (1) and (3) are to the 100% level and curves (2) and (4) are to the 0% level the better the system is performing. Figure 6.4 presents the results for the dot format with 12 quantisation levels per parameter.

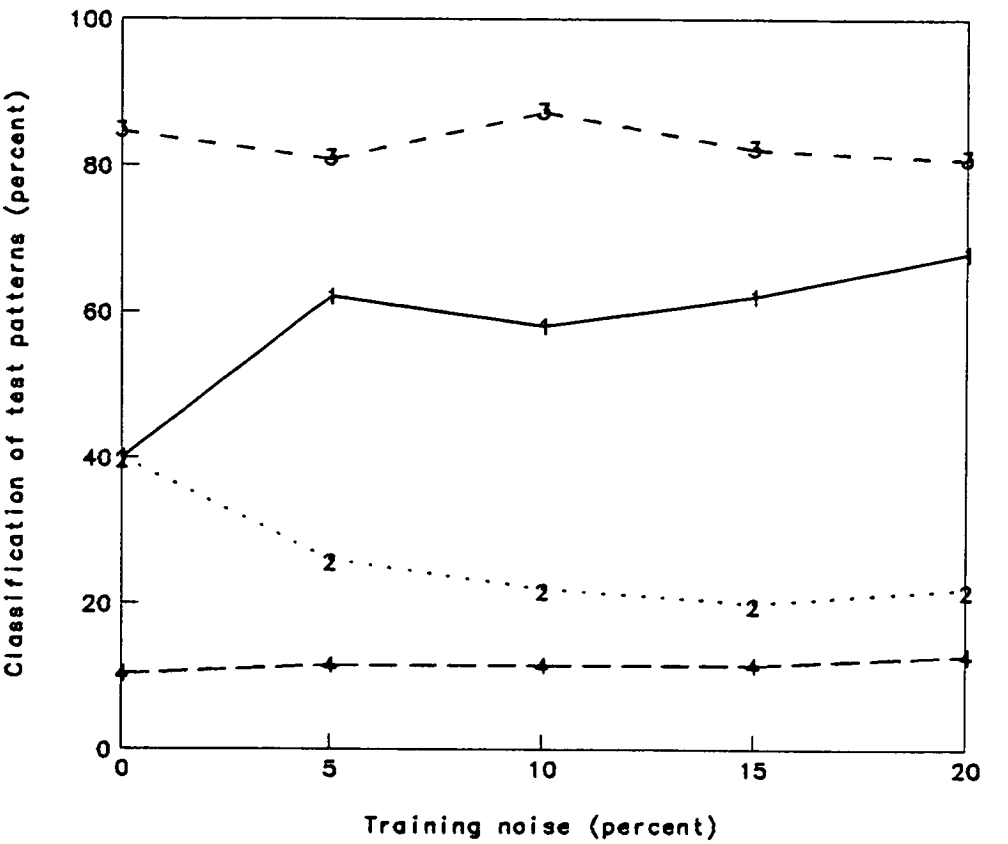

Figure 6.4 Results at various noise levels for 64 test patterns (dot format). Network trained with 67 patterns in dot format and 12 quantisation levels.

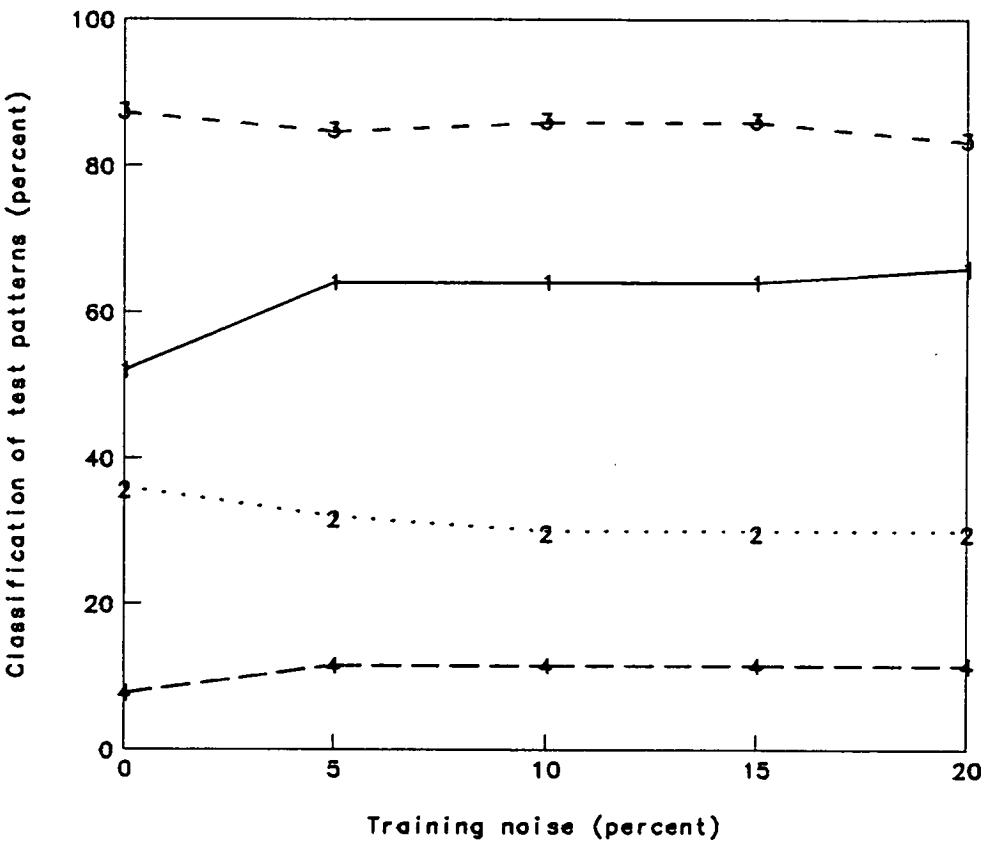

Figure 6.5 Results at various noise levels for 64 test patterns (slide format). Network trained with 67 patterns in slide format and 12 quantisation levels.

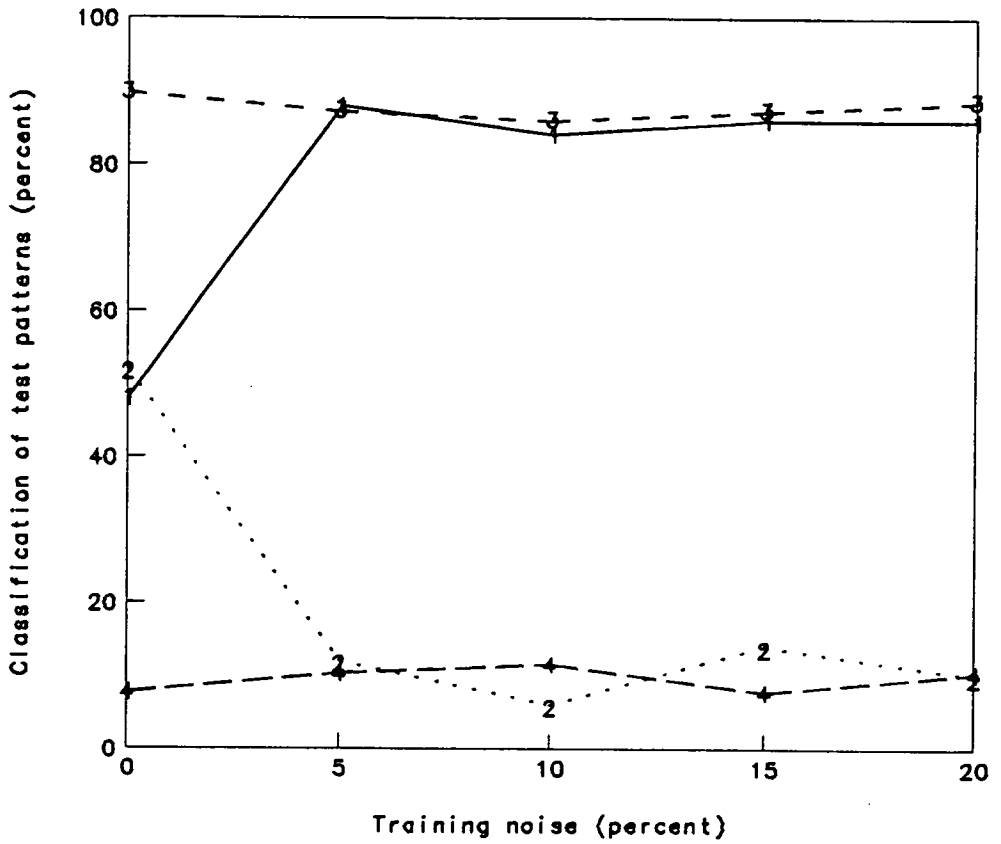

Figure 6.6 Results at various noise levels for 64 test patterns (bar format). Network trained with 67 patterns in bar format and 12 quantisation levels.

The level perturbed during noisy clamping with 12 quantisation levels is half that for each network in the experiments with 6 quantisation levels for each parameter. There are also twice as many weights. For the networks trained using the dot and slide formats the correct classification of controls and pathologicals remains very similar as the noise levels are increased. However, the miss-classifications are reduced as the noise is increased. It should be noted that the noise referred to here is the probability of perturbing the quantisation level for each parameter by one level, and does not reflect the addition of noise to the patterns by changing the state of randomly selected inputs to the network. The noise approach used here, appears to expand the training pattern set, by using the original patterns as the basis for perturbation of the parameter quantisation levels and thus increases the models ability to generalise (Elman & Zipser 1987).

For both the slide and dot format, the level of correct classification for pathologicals never reaches the levels achieved for the controls. However, with the bar format similar levels of correct classification for both controls and pathological subjects is achieved from a 5% probability of perturbing each parameter by one level.

In addition, for the dot and slide format the number of pathological subjects incorrectly classified as controls never achieves the same level as for the number of control subjects incorrectly classified as pathologicals. However, using the bar format, it was possible to achieve similar levels after training with 5% to 20% probability of perturbing a parameter by one level. Between the dot, slide and bar formats, the levels for control subjects correctly identified as controls and for control subjects incorrectly identified as pathologicals remain very similar indeed.

The best generalisation with the test data set of 39 controls and 25 pathological voices for the *dot* format was 79.5% of control speakers correctly classified as control speakers and 68.0% of pathological speakers correctly classified as pathological speakers. However, random selection would give 50% correct classification. Thus a result of 68.0% classification is very close to random, and hence would not be of any practical use in a screening system. The network

trained using data in the *slide* format gave results of 84.6% and 64.0% respectively. The *bar* format however gave much better results for the pathological group with 88.0% correctly classified, and 84.6% for control speakers correctly classified.

Figures 6.7 to 6.9 depicts the weights learned by these networks for the different data formats at 12 quantisation levels. No noisy clamping was used in training for the networks shown.

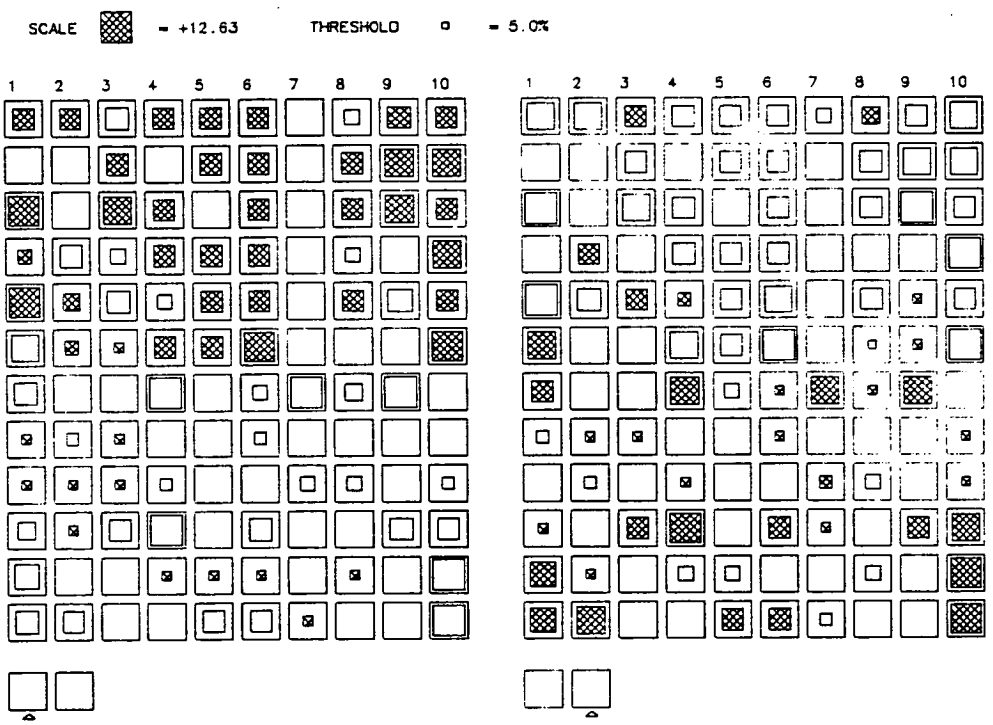

Figure 6.7 Weights learned by network with no hidden units. Training data comprised 67 pattern set in the *dot* format, with 12 quantisation levels per parameter.

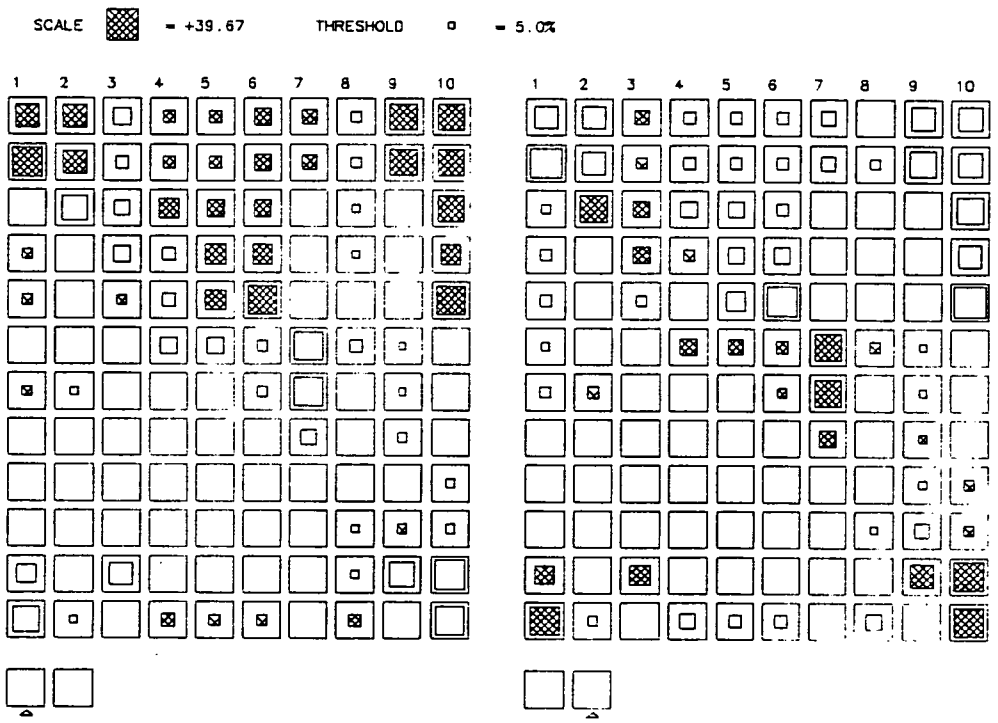

Figure 6.8 Weights learned by network with no hidden units. Training data comprised 67 pattern set in the *slide* format, with 12 quantisation levels per parameter.

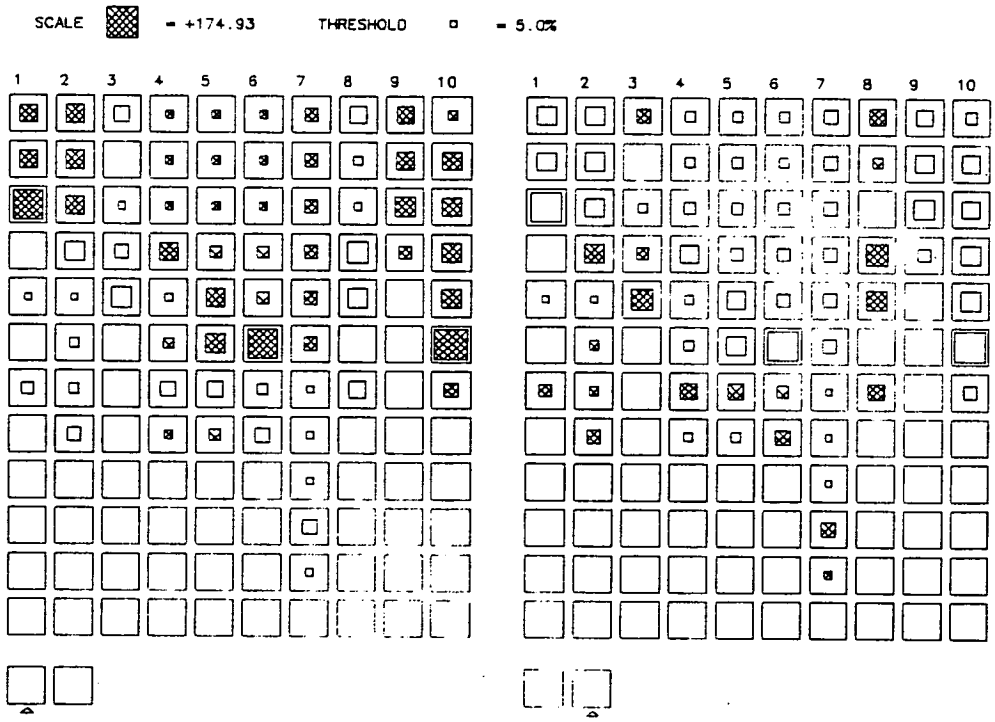

Figure 6.9 Weights learned by network with no hidden units. Training data comprised 67 pattern set in the *bar* format, with 12 quantisation levels per parameter.

The weights learned are obviously similar to those learned using all the 131 patterns discussed earlier. Of interest here is the reduced use of lower than *normal* values of perturbation parameters. This suggests that the test data set may contain speakers where lower than normal perturbation values are present. Again, the weight maps for the networks trained using data in the bar format are characterised by not using the lower four quantisation levels (except for S-DEVEX).

The networks were also trained using the training set of 10 control speakers and 10 pathological speakers with requisite resolutions of 6 and 12 quantisation levels for each of the ten parameters. The testing set data comprised 68 control speakers and 43 pathological speakers. For the networks with 60 input units trained with data in the *dot* format the best generalisation was 86.8% of controls correctly classified, and 74.4% of pathologicals correctly classified. Corresponding figures for the *slide* format are 91.2% and 68.6%, and for the *bar* format 91.9% and 74.4%.

For networks with 120 input units the result are as follows. Training with the data in *dot* format gives 80.9% controls classified correctly and 70.9% pathologicals classified correctly. Corresponding figures for the *slide* format are 90.4% and 74.4%, and for the *bar* format 90.4% and 66.3%. The weights learned by the networks for the three data formats are depicted in Figures 6.10 to 6.12. The weights are shown after training with no noisy clamping.

The weight map for the network trained with data in the *dot* format is characterised by the vast majority of weights having very similar values. This indicates that there has been very little overlap between the training patterns. The same is evident, although to a lesser extent for the *slide* format and also the *bar* format. Despite very few training patterns being used the generalisation results are similar to those obtained for the larger training set. This would seem to imply that there is a high degree of correlation between patterns of the same class.

The networks trained on data with 12 quantisation levels per parameter generally did not perform as well in generalising to the test data as networks trained using data with 6 quantisation levels per parameter. This would imply that for this data set single-layer networks were not able to develop suitable feature detectors to take advantage of the finer data resolution. Additionally, in these experiments the 120 input networks have more degrees of freedom than the 60 input networks and thus a larger search space.

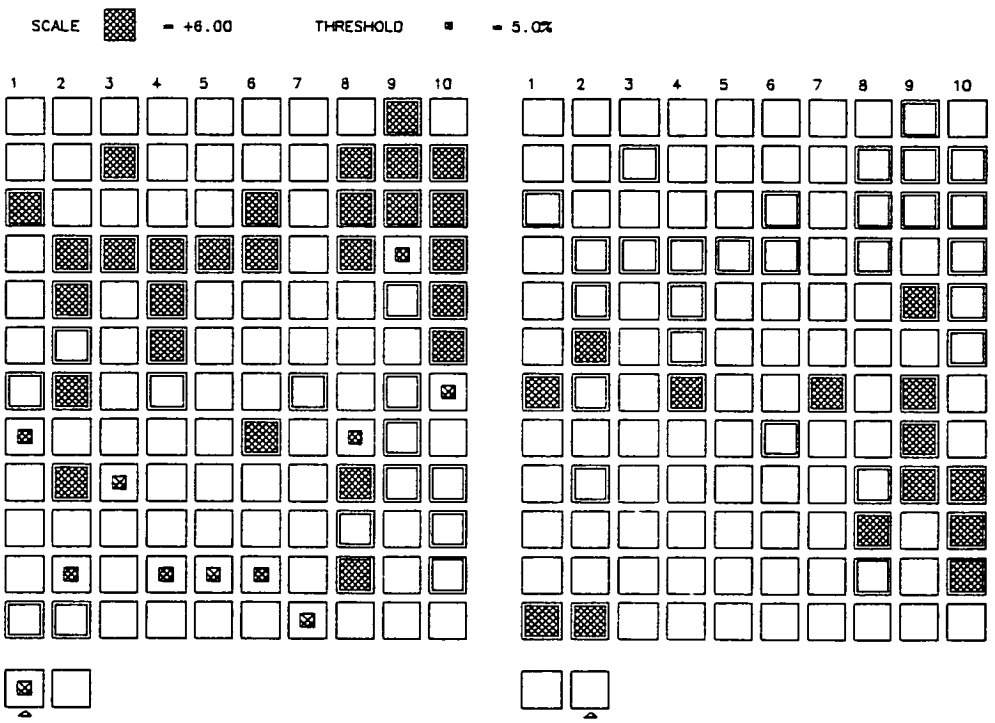

Figure 6.10 Weights learned by network with no hidden units. Training data comprised 20 pattern set in the *dot* format, with 12 quantisation levels per parameter.

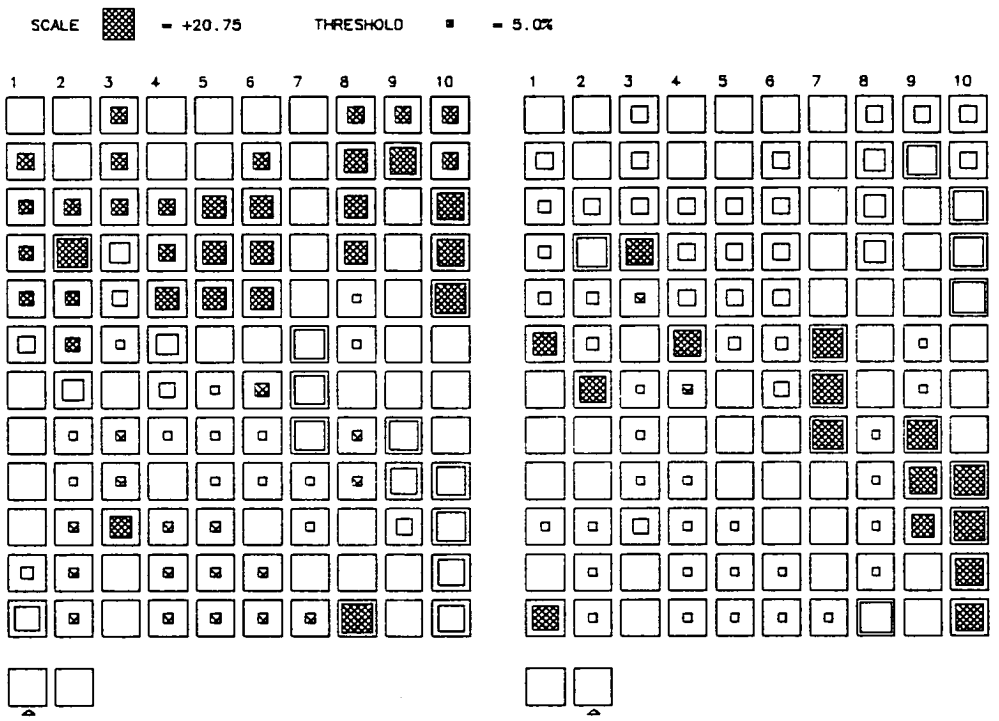

Figure 6.11 Weights learned by network with no hidden units. Training data comprised 20 pattern set in the *slide* format, with 12 quantisation levels per parameter.

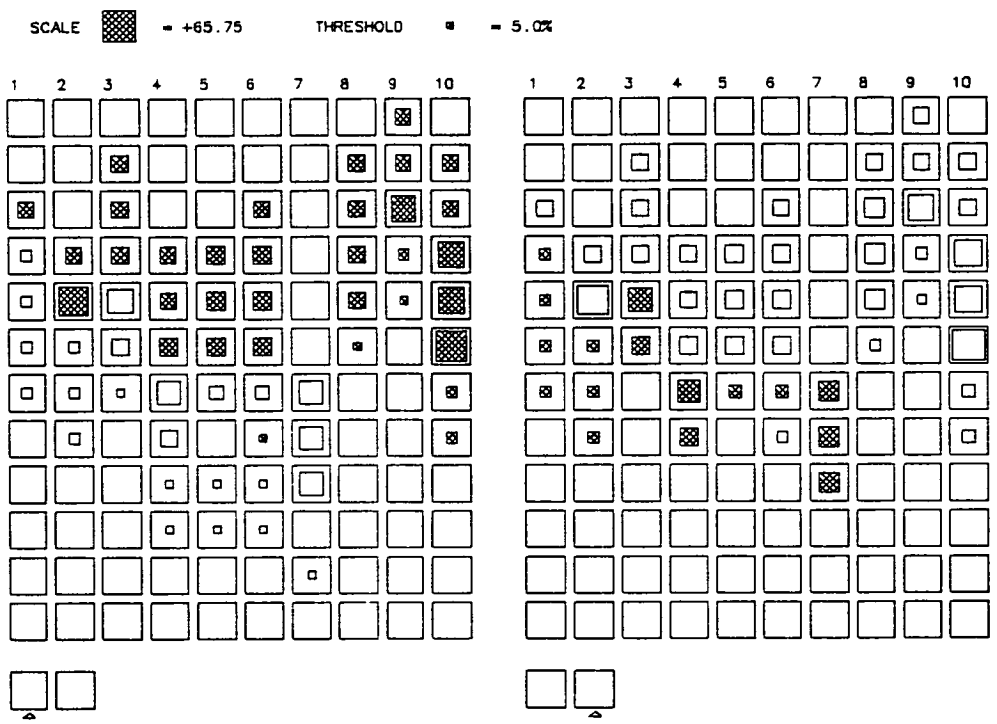

Figure 6.12 Weights learned by network with no hidden units. Training data comprised 20 pattern set in the *bar* format, with 12 quantisation levels per parameter.

6.5.2 Hidden Units

The results for experiments using networks with no connections between the input and output units and between 2, 4, and 8 hidden units are as follows for various levels of noise.

For networks trained with data in the *dot* format, with 6 quantisation levels per parameter, the generalisation results obtained are shown in Table 6.6.

Hidden Units	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
2	92.3%	7.7%	36.0%	64.0%
4	89.7%	10.3%	30.0%	70.0%
8	87.2%	12.8%	40.0%	60.0%

Table 6.6 Generalisation results for networks with hidden units trained with data in the *dot* format, with 6 quantisation levels per parameter.

For networks trained with data in the *slide* format, with 6 quantisation levels per parameter, the generalisation results obtained are shown in Table 6.7.

Hidden Units	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
2	85.9%	14.1%	28.0%	72.0%
4	87.2%	12.8%	24.0%	76.0%
8	84.6%	15.4%	28.0%	72.0%

Table 6.7 Generalisation results for networks with hidden units trained with data in the *slide* format, with 6 quantisation levels per parameter.

For networks trained with data in the *bar* format, with 6 quantisation levels per parameter, the generalisation results obtained are shown in Table 6.8.

Hidden Units	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
2	89.7%	10.3%	32.0%	68.0%
4	85.9%	14.1%	22.0%	78.0%
8	92.3%	7.7%	40.0%	60.0%

Table 6.8 Generalisation results for networks with hidden units trained with data in the *bar* format, with 6 quantisation levels per parameter.

The results indicate that the networks trained with the various formats with data transformed to 6 quantisation levels can perform slightly better in correctly identifying control speakers than those without hidden units. However, this is usually at the expense of degraded performance in correctly identifying the pathological subjects.

For networks trained with data in the *dot* format, with 12 quantisation levels per parameter, the generalisation results obtained are shown in Table 6.9.

# Hidden Units	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
2	84.6%	15.5%	32.0%	68.0%
4	83.3%	16.6%	18.0%	82.0%
8	89.7%	10.3%	44.0%	56.0%

Table 6.9 Generalisation results for networks with hidden units trained with data in the *dot* format, with 12 quantisation levels per parameter.

For networks trained with data in the *slide* format, with 12 quantisation levels per parameter, the generalisation results obtained are shown in Table 6.10.

# Hidden Units	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
2	89.7%	10.3%	20.0%	80.0%
4	84.6%	15.4%	24.0%	76.0%
8	84.6%	15.4%	24.0%	76.0%

Table 6.10 Generalisation results for networks with hidden units trained with data in the *slide* format, with 12 quantisation levels per parameter.

For networks trained with data in the *bar* format, with 12 quantisation levels per parameter, the generalisation results obtained are shown in Table 6.11.

# Hidden Units	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
2	89.7%	10.3%	8.0%	88.0%
4	89.7%	10.3%	18.0%	82.0%
8	87.2%	12.8%	20.0%	80.0%

Table 6.11 Generalisation results for networks with hidden units trained with data in the *bar* format, with 12 quantisation levels per parameter.

The results indicate that the networks trained with the various formats with data transformed to 12 quantisation levels are capable of achieving better performance in correct identification of pathological voices than networks trained with data transformed to 6 quantisation levels, with generally only a slight degradation in performance for the control speakers. However, for networks with no hidden

units, there is little difference in performance for pathological classification using either 6 or 12 quantisation levels. But at the same time, the control speaker classification performance was generally better by using only 6 quantisation levels rather than 12. This suggests that the hidden units may be developing feature detectors for pathology classification, and to a limited extent for control classification, where higher parameter resolution is required to enable the features to be adequately distinguished. For the single-layer networks, the higher degree of overlap between patterns at the lower resolution level rather than the higher resolution level may have allowed the network to learn a coarse representation of the control data, which then gave better results on the control test data set due to its greater homogeneity.

The effect of training using the noisy clamping scheme may be seen for networks trained with data in the *bar* format in Figures 6.13 to 6.15. The effect for the other two data formats is similar in that all of the various classification values (1) to (4) tended to be relatively smooth. This would appear to indicate that networks with hidden units had better ability in capturing the underlying features in the training data used.

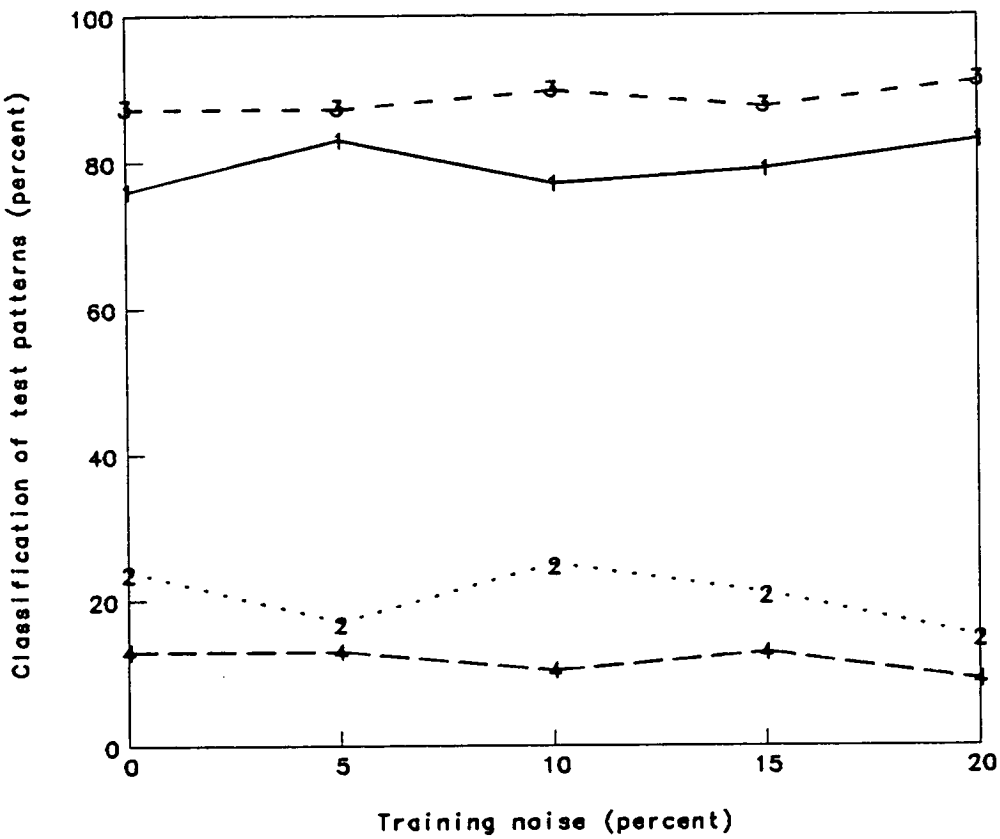

Figure 6.13 Results at various noise levels for 120 input unit network and 2 hidden units trained with 67 patterns in *bar* format.

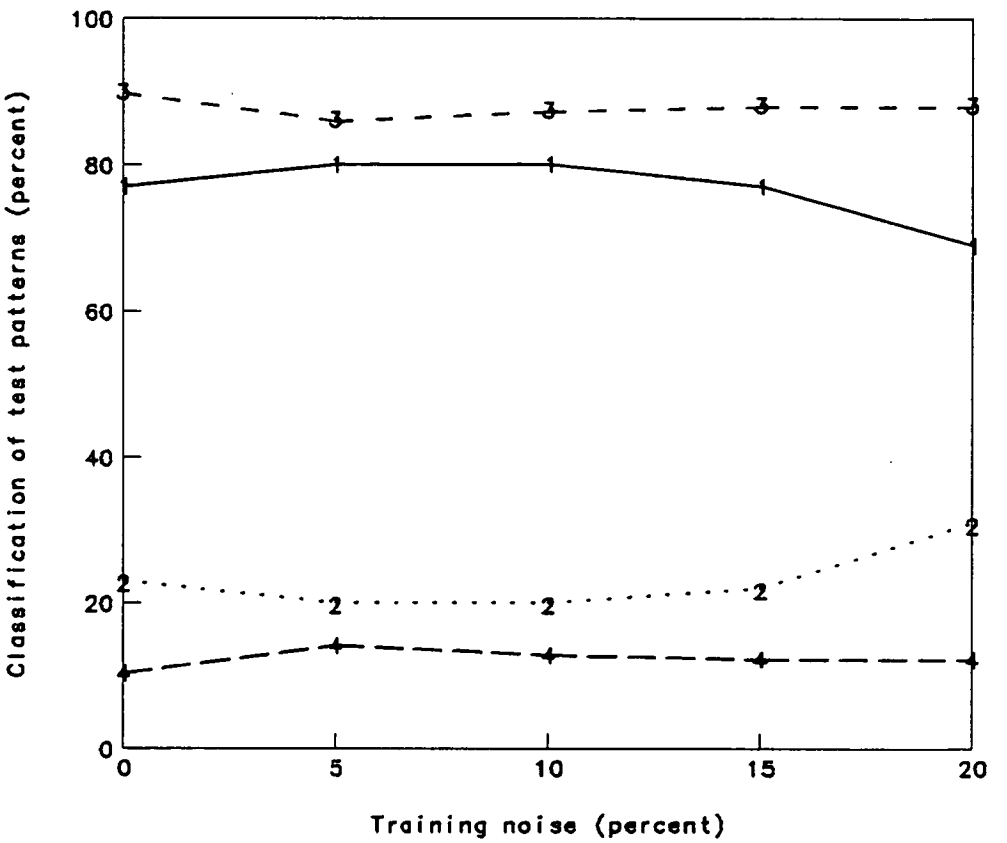

Figure 6.14 Results at various noise levels for 120 input unit network and 4 hidden units trained with 67 patterns in *bar* format.

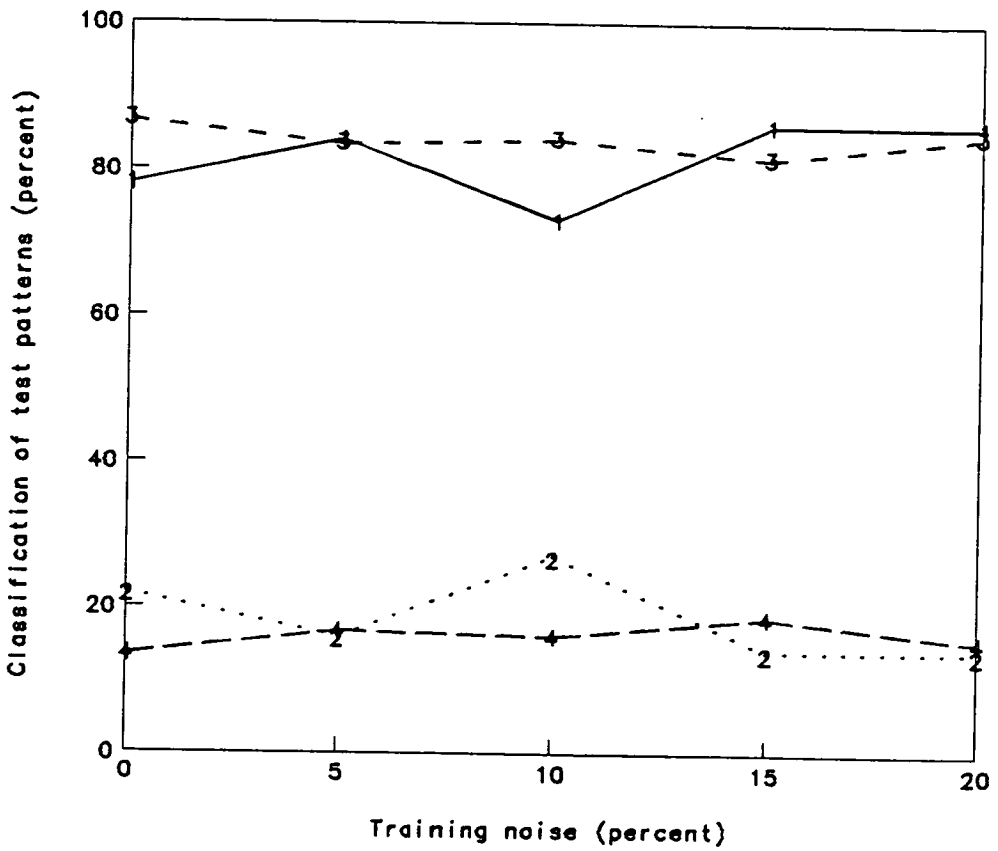

Figure 6.15 Results at various noise levels for 120 input unit network and 8 hidden units trained with 67 patterns in *bar* format.

The networks trained with the *bar* format tended to perform better with 12 quantisation levels than the other formats. Examples of weights learned by some of these networks using the data in *bar* format and noisy clamping turned off, are shown in Figures 6.16 to 6.18. Figure 6.16 indicates the weights learned using a network with two hidden units. One would expect that only one hidden unit would be used, as experiments in the previous section have shown that a 120 input unit and one output unit network can learn to differentiate between the two training classes. However, a number of small connections have been formed that use the second hidden unit.

The weights learned between the output units are shown at the bottom of the map. This indicates that there are weights with negative values connecting the two units. Thus each output unit acts to inhibit the other. When the leftmost hidden unit is *on* it indicates a control speaker. Positive weights from the input array to ensure that this hidden unit comes on when a control speaker is presented to the network may be seen for parameters 2 (F0-DEV), 3 (J-DEVEX), 4 (J-AVEX) and 8 (S-AVEX). The presence of a pathological speaker causes the leftmost hidden unit to be inhibited, so preventing the control output unit from coming on. To prevent the hidden unit from turning on, there are strong negative weights for areas of the followings parameters 1 (F0-AV), 2 (F0-DEV), 5 (J-RATEX), 6 (J-DPF), 7 (S-DEVEX), 9 (S-RATEX) and 10 (S-DPF). In addition, a number of small positive weights have been formed between the second hidden unit and the input units. These connections are for parameters 1 (F0-AV), 2 (F0-DEV), 3 (J-DEVEX) and 7 (S-DEVEX). Some of the weights developed between the hidden units and input units bear a resemblance to those learned by the network with no hidden units (see Figure 6.9) for some of the parameters 7 (S-DEVEX) and 10 (S-DPF). The network with hidden units makes more use of the lower quantisation levels than the network without hidden units. However, the hidden unit network has more than twice the weights that were used in the network with no hidden units.

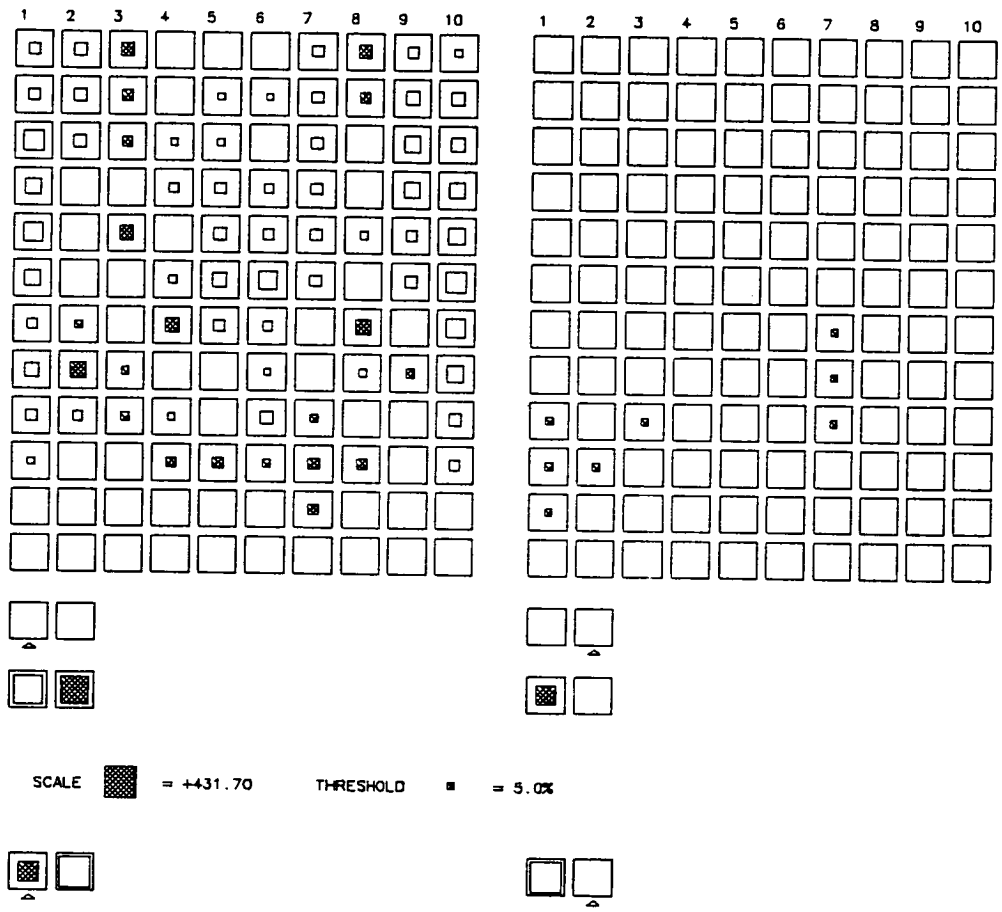

Figure 6.16 Weights learned using training data in bar format and network with 2 hidden units.

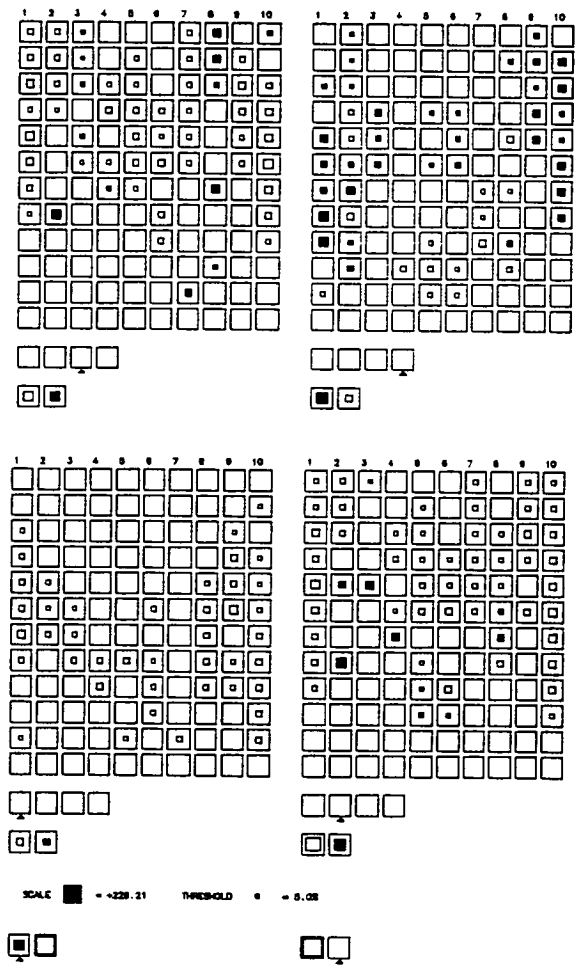

Figure 6.17 Weights learned using training data in bar format and network with 4 hidden units.

Figure 6.17 depicts the weights learned for a network with 4 hidden units. In this case each hidden unit makes use of similar numbers of connections to the input units as judged at the 5% weight threshold level indicated. The leftmost hidden unit is referred to as hidden unit 1 and the rightmost as hidden unit 4. It may be seen that positive value weights connected to hidden unit 4 are used in detecting pathological subjects, whereas positive value weights to the remaining hidden units are used in detecting control subjects.

Conversely, it may also be seen that negative value weights connected to hidden unit 4 are used in detecting control subjects, whereas negative value weights to the remaining hidden units are used in detecting pathological subjects. It may be seen that the positive weights connecting the input units to the third hidden unit are similar to those learned in the two hidden unit network in Figure 6.16 connecting the input units to the first hidden unit. Again, with the four hidden unit network, more use of the lower levels of intonation and perturbation parameters is made compared to those weights developed by the network with no hidden units (Figure 6.9).

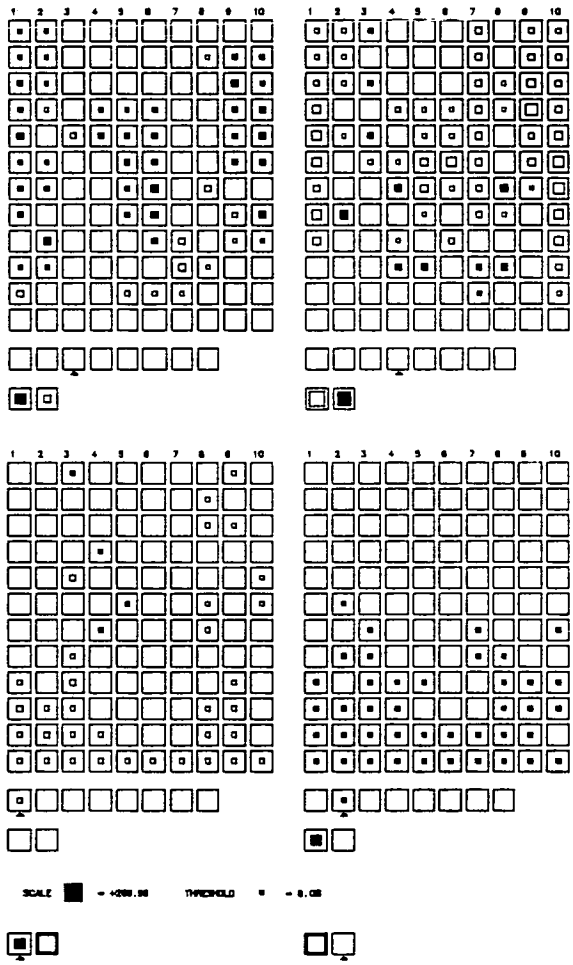

Figure 6.18a Weights learned using training data in bar format and 8 hidden units (weights to first four hidden units shown).

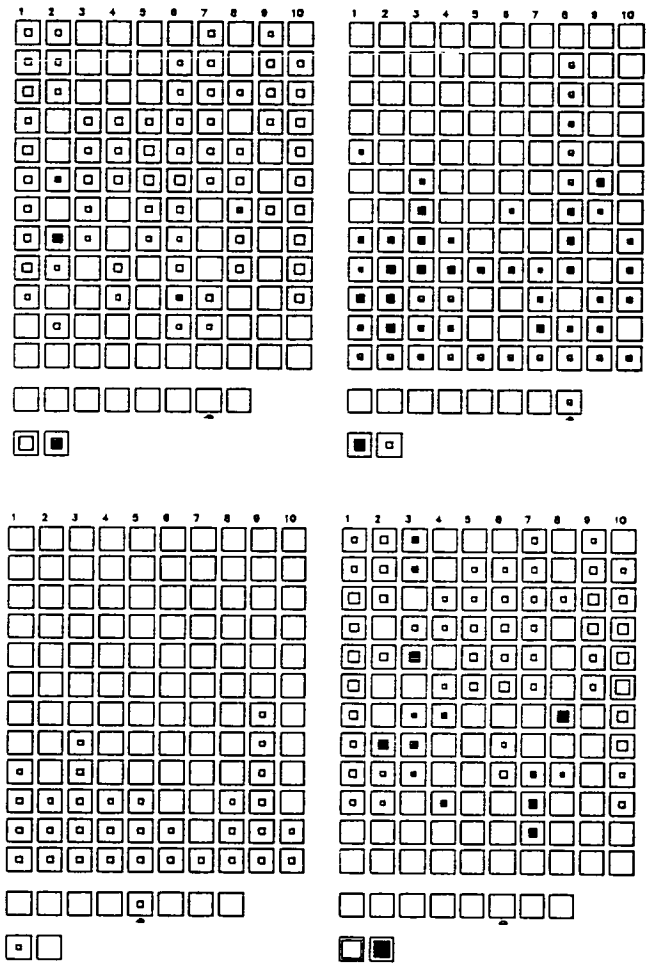

Figure 6.18b Weights learned using training data in bar format and 8 hidden units (weights to last four hidden units shown).

Figure 6.18a and 6.18b depict the weights learned for the network with 8 hidden units. Each hidden unit maybe seen to have made connections to the input units. In this case, connections utilising the lowest quantisation levels have been used by a number of the hidden units.

6.5.3 Intonation, Shimmer and Jitter Receptive Fields

Using the dot format and three receptive fields, one for the two intonational parameters, one for the four jitter parameters and one for the remaining four shimmer parameters, it was found that using 12 quantisation levels the generalisation results shown in Table 6.12 were obtained.

Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	79.5%	20.5%	44.0%	56.0%
slide	79.5%	19.2%	24.0%	76.0%
bar	85.9%	14.1%	28.0%	72.0%

Table 6.12 Generalisation results for networks with one hidden unit per intonation, shimmer and jitter receptive fields, trained with data in the three formats and 12 quantisation levels per parameter.

It was not possible for the training set to be correctly learned within the 500 learning cycles allowed for this type of network. On average 93.8% of the control patterns were correctly learned for data in the *dot, slide* and *bar* formats. For the pathological patterns the average was 89.6%. Incorporating an additional hidden unit per restricted field yielded no overall improvements in learning or generalisation.

Examples of the weights learned for the different formats are shown in Figures 6.19 to 6.21.

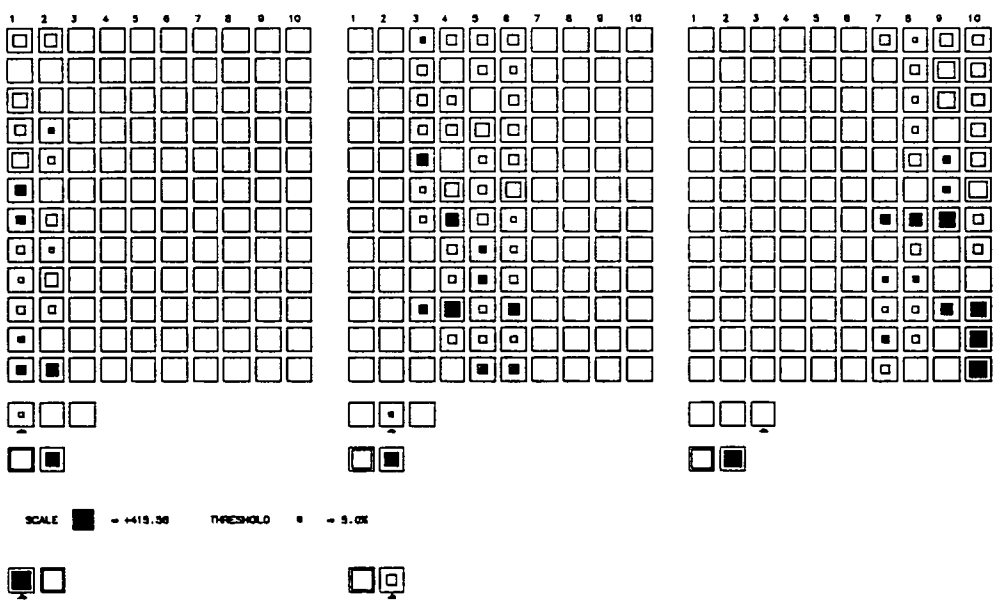

Figure 6.19 Weights learned using training data in dot format with 12 quantisation levels per parameter. Network has one hidden unit for each of the intonation, jitter and shimmer receptive fields.

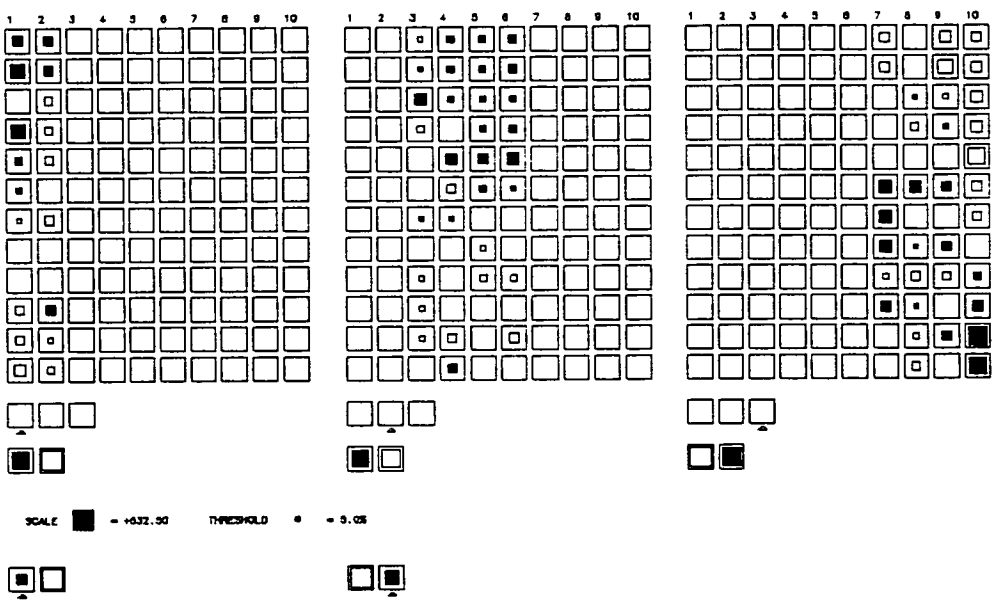

Figure 6.20 Weights learned using training data in slide format and 12 quantisation levels per parameter. Network has one hidden unit for each of the intonation, jitter and shimmer receptive fields.

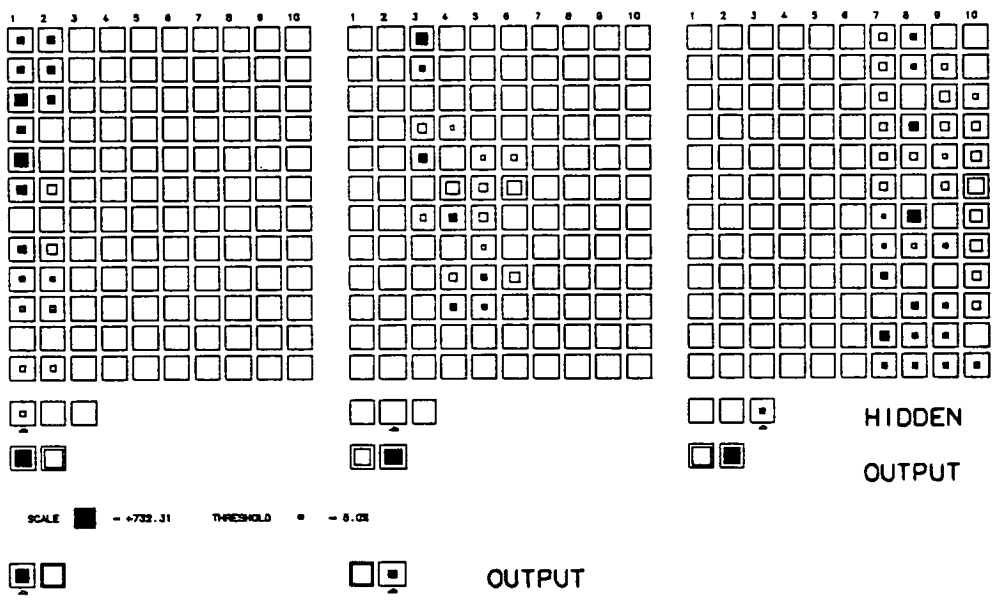

Figure 6.21 Weights learned using training data in bar format and 12 quantisation levels per parameter. Network has one hidden unit for each of the intonation, jitter and shimmer receptive fields.

Figures 6.19 depicts an example of weights learned using the *dot* data format. The positive weights in each receptive field indicate connections made for determining control subjects. The pathological group output unit has a large positive bias value. The negative value weight throughout the input fields indicate parameters used in determining the pathological group. Presentation of a subject from the pathology group (if identified correctly) results in all the hidden units being *off*. The bias value for the control output unit is negative and ensures that it is *off* when there is no input to it from the hidden units. The large positive bias value for the pathological group output unit results in this unit coming *on* when there is no input to it from the hidden units.

Figure 6.20 depicts an example of weights learned using the *slide* data format. The positive value weights in the intonation and jitter receptive fields (parameters 1 to 6) are used in identifying pathological speakers. The negative value weights are used to identify the control speakers. In the shimmer receptive field negative weights are used to identify pathological speakers, and positive value weights the control speakers.

Figure 6.21 depicts an example of weights learned using the *bar* data format. The positive value weights in the intonation receptive field (parameters 1 and 2) are used in identifying pathological speakers. The negative value weights are used to identify the control speakers. In the jitter and shimmer receptive fields negative weights are used to identify pathological speakers, and positive value weights the control speakers.

The number of weights in the networks with one hidden unit per receptive field is very similar to the number of weights used in a 120 input unit single layer network. Results for the receptive field network are not as good as those for the single-layer nets. It should be noted that none of the receptive field networks simulated were able to correctly learn all of the training data. This would imply that the receptive field restrictions did not allow the networks to develop a good set of feature detectors.

6.5.4 Intonation and Perturbation Parameter Receptive Fields

Using 10 receptive fields, the results obtained for one hidden unit per field are displayed in Table 6.13. A maximum of 500 learning cycles was allowed for each of the network. For the dot format 82.1% of controls and 73.2% of pathological voices were correctly learned. Using the slide format, 88.5% controls and 75.0% of pathological voices were correctly learned. With the bar format, 84.6% control and 78.6% of pathological voices were correctly learned.

Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	75.6%	24.4%	48.0%	52.0%
slide	79.5%	20.5%	40.0%	60.0%
bar	71.8%	28.2%	40.0%	60.0%

Table 6.13 Results for 10 receptive field networks with one hidden unit per receptive field. Training data was quantised to 12 levels.

For networks with two hidden units per receptive field the results shown in Table 6.14 were obtained. For the dot format 73.1% of controls and 55.4% of pathological voices were correctly learned. Using the slide format, 88.5% controls and 83.9% of pathological voices were correctly learned. With the bar format, 96.2% control and 69.6% of pathological voices were correctly learned.

Figures 6.22a and 6.22b depict an example of the weights learned for this type of network with training data in the *slide* format.

Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	61.5%	38.5%	58.0%	42.0%
slide	65.4%	34.6%	24.0%	76.0%
bar	73.1%	26.9%	36.0%	64.0%

Table 6.14 Results for 10 receptive field networks with two hidden unit per receptive field. Training data was quantised to 12 levels.

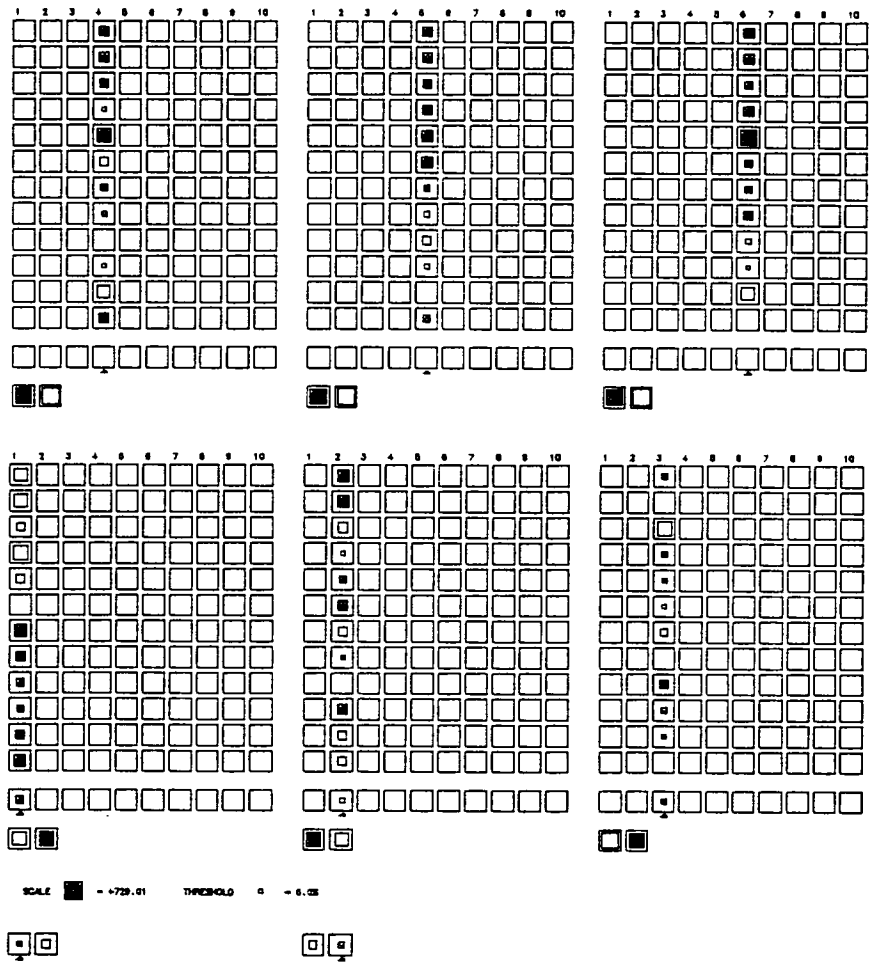

Figure 6.22a Weights learned using training data in *slide* format and a restricted receptive field for each parameter (connections to hidden units 1 to 6 shown)

The results are poor for this type of network connectivity, being comparable with being random. For all the networks considered in this topology and for all the data formats, it was not possible to learn all the training patterns. This would obviously have a detrimental affect on the classification of patterns from the testing data set. The use of two hidden units rather than one for each parameter did not generally result in any real improvements. The use of the restricted receptive fields appears to make the learning problem harder here. It is also possible that some of the weights may have become too large, thus preventing the network from reaching equilibrium and allow the network to develop good feature detectors with this training data.

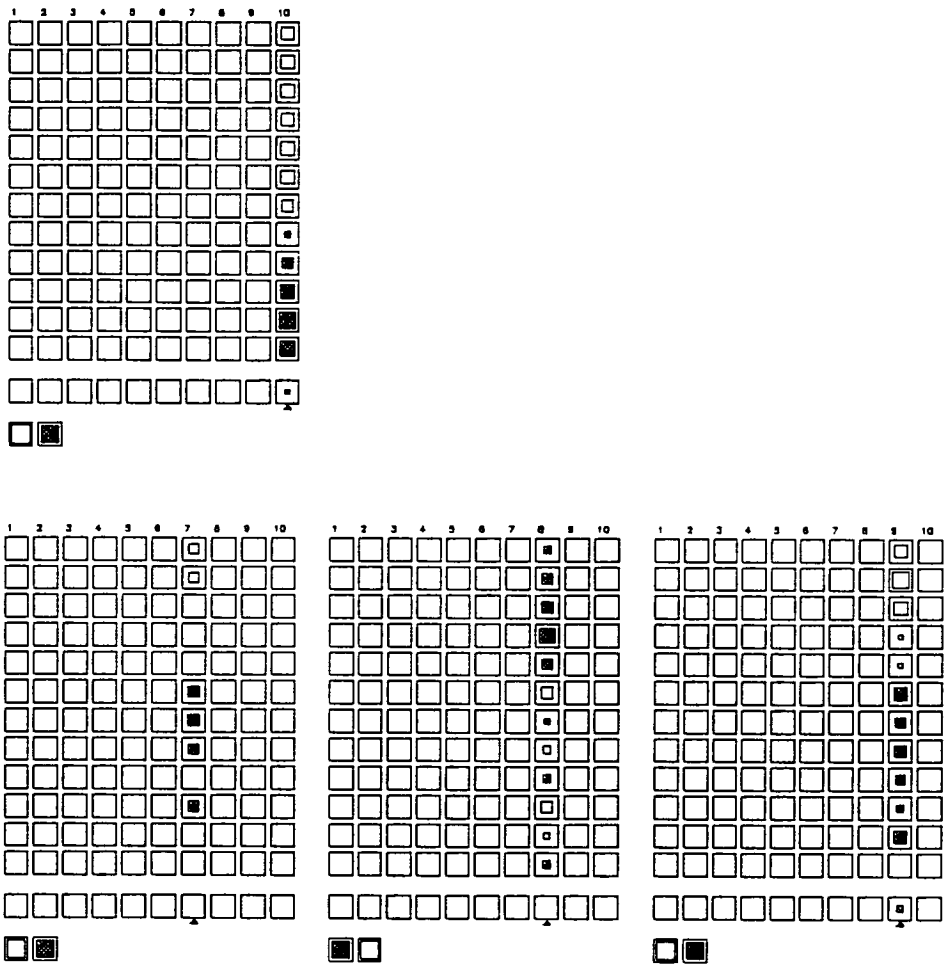

Figure 6.22b Weights learned using training data in *slide* format and a restricted receptive field for each parameter (connections to hidden units 7 to 10 shown)

6.5.5 Selected Parameter Networks

All networks with 48 input units and one output unit trained with data in the *dot* and *slide* format were able to correctly learn all the training data. However, using the *bar* format none of the networks were able to learn all of the training set. The average percentage of patterns learned was 59.7%.

Generalisation results are shown in Tables 6.15 to 6.18 for data presented to the networks in the *dot* and *slide* formats.

Set 1 Data	Control subjects classified as:		Pathological subjects classified as:	
Format	Control	Pathological	Control	Pathological
dot	87.2%	10.3%	50.0%	50.0%
slide	89.7%	10.3%	40.0%	60.0%

Table 6.15 Results for single-layer network trained using parameters from subset 1 with 12 quantisation levels per parameter.

Set 2 Data	Control subjects classified as:		Pathological subjects classified as:	
Format	Control	Pathological	Control	Pathological
dot	74.3%	25.6%	32.0%	56.0%
slide	84.6%	10.3%	28.0%	56.0%

Table 6.16 Results for single-layer network trained using parameters from subset 2 with 12 quantisation levels per parameter.

Set 3 Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	79.5%	17.9%	48.0%	44.0%
slide	84.6%	10.3%	24.0%	56.0%

Table 6.17 Results for single-layer network trained using parameters from subset 3 with 12 quantisation levels per parameter.

Set 4 Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	76.9%	17.9%	36.0%	44.0%
slide	87.2%	5.1%	32.0%	60.0%

Table 6.18 Results for single-layer network trained using parameters from subset 4 with 12 quantisation levels per parameter.

The results are characterised by poor recognition of pathological speakers for all the subset groups.

Figure 6.23 to 6.26 depict examples of the weights learned for 48 input unit and single output unit networks with the different parameter sets for the training data.

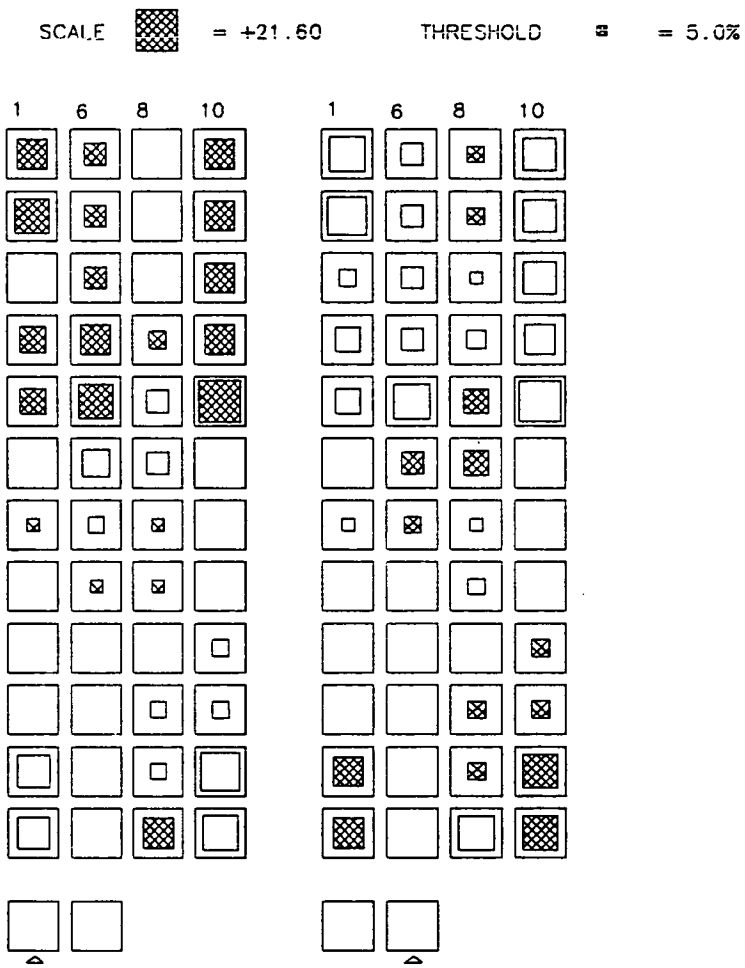

Figure 6.23 Weights learned using set 1 training data in *slide* format for network with 48 input and one output unit.

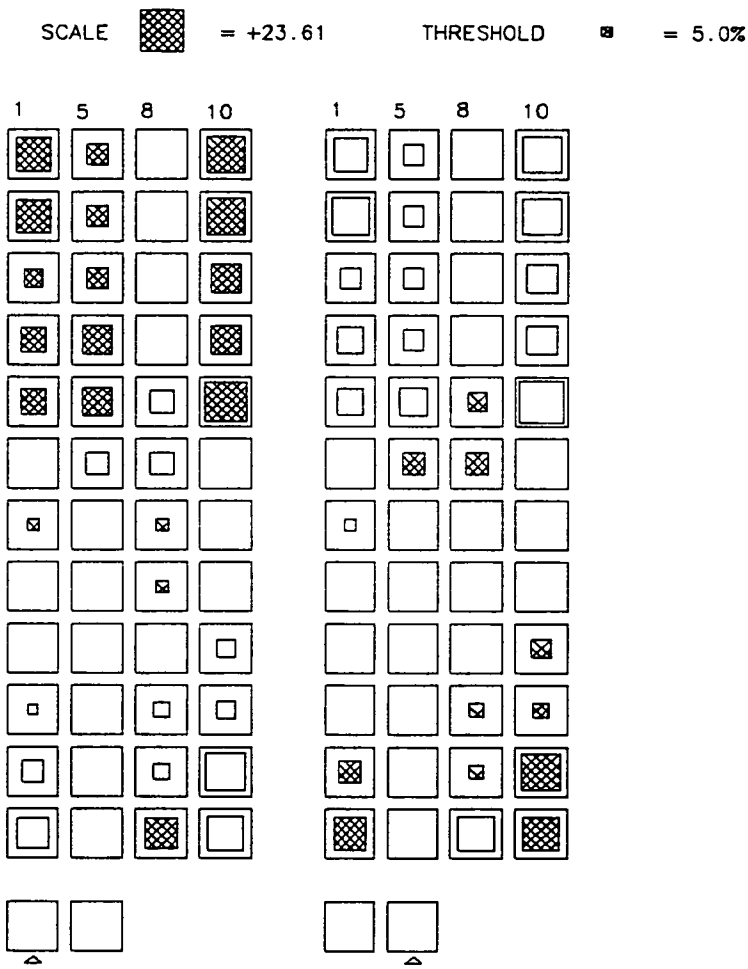

Figure 6.24 Weights learned using set 2 training data in *slide* format for network with 48 input and one output unit.

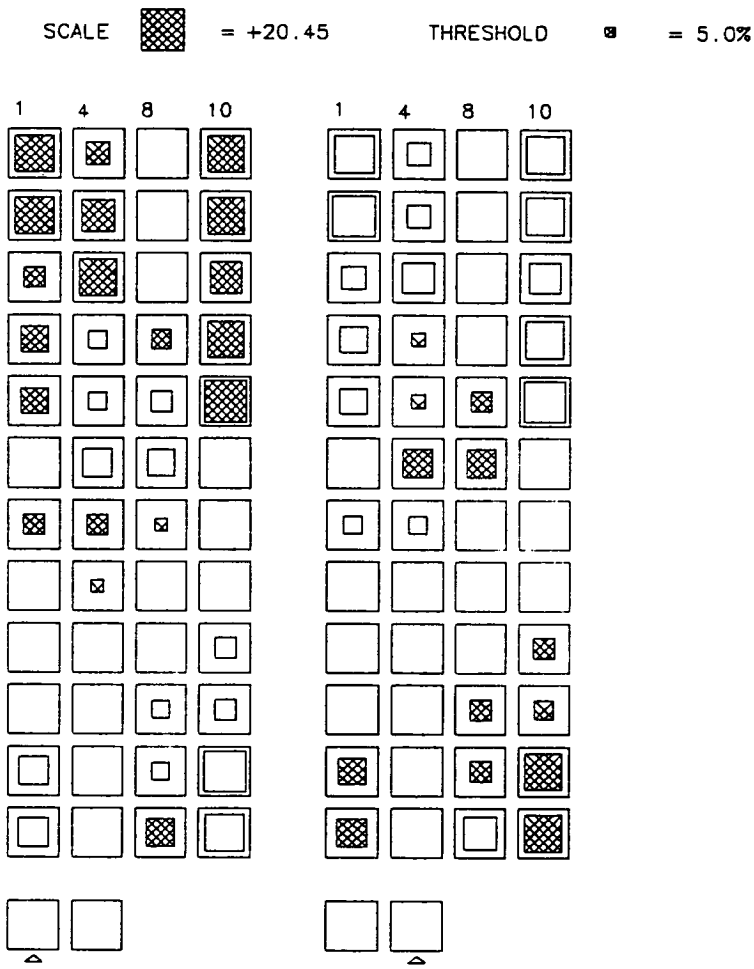

Figure 6.25 Weights learned using set 3 training data in *slide* format for network with 48 input and one output unit.

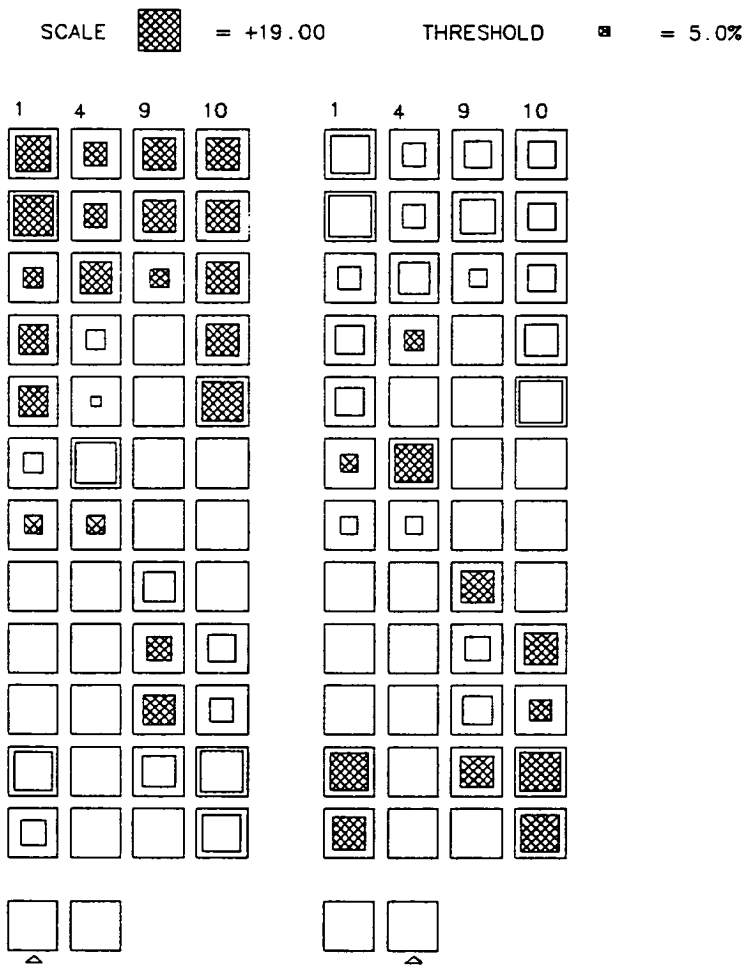

Figure 6.26 Weights learned using set 4 training data in *slide* format for network with 48 input and one output unit.

The pattern of weights learned for parameters 1 (F0-AV) and 10 (S-DPF) are similar for each of the networks trained with one of the parameter subsets. The weight patterns produced using data subsets 1 and 3 are also very similar. Lower than normal levels of perturbation are shown for parameter 8 (S-AVEX) in characterising pathological speakers.

Networks with two hidden units trained with data in the *dot* and *slide formats,* and 6 quantisation levels, were able to learn all of the training set patterns. The generalisation results obtained for each of the parameters are shown in Tables 6.19 to 6.22.

Set 1 Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	76.9%	20.5%	32.0%	64.0%
slide	71.8%	28.2%	32.0%	68.0%

Table 6.19 Results for network with two hidden units trained using parameters from subset 1 with 6 quantisation levels per parameter.

Set 2 Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	74.4%	17.9%	24.0%	60.0%
slide	64.1%	23.1%	24.0%	64.0%

Table 6.20 Results for network with two hidden units trained using parameters from subset 2 with 6 quantisation levels per parameter.

Set 3 Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	64.1%	15.4%	20.0%	60.0%
slide	59.0%	15.4%	20.0%	60.0%

Table 6.21 Results for network with two hidden units trained using parameters from subset 3 with 6 quantisation levels per parameter.

Set 4 Data Format	Control subjects classified as:		Pathological subjects classified as:	
	Control	Pathological	Control	Pathological
dot	84.6%	10.3%	28.0%	56.0%
slide	82.1%	10.3%	20.0%	68.0%

Table 6.22 Results for network with two hidden units trained using parameters from subset 4 with 6 quantisation levels per parameter.

The number of weights in the networks with no hidden units and 48 input units was 48. The total number of weights in the networks with 24 input units, 2 hidden units and 2 output units was 76. The results for networks with hidden units are better for pathological speakers than those nets with no hidden units. It appears that using the hidden units enables the networks to develop better feature detectors in this case for the pathological subjects.

6.6 COMPARISON WITH RESULTS FROM OTHER WORK

Results using data from the pool used for experiments described in this chapter have been reported by Laver, Hiller, Mackenzie & Rooney, (1986) and Beck (1988). The methods used include bivariate plots, linear discriminant analysis and maximum likelihood classification. These techniques assume that the data

shows a normal distribution, but given the heterogeneous composition of the pathological group it is likely that this assumption is seriously violated. Despite this, the techniques are reported to be quite robust in the face of such violations. The Boltzmann Machine techniques discussed do not require the data to conform to the above requirements.

Beck (1988) reports 87.5% correct classifications of pathological subjects, and 8.4% incorrect classification of pathologicals as controls using linear discriminant analysis. These results were obtained from within group testing. The results using various connectivity arrangements for Boltzmann Machines used a much smaller training set than the data set used by Beck (1988) in applying linear discriminant analysis and maximum likelihood techniques. The best results for classification of pathological speakers was 88.0% with 8.0% of pathological subjects classified as controls. This result was obtained for a network using 12 quantisation levels for all parameters, 120 input units and 2 hidden unit. It is likely that these figures would be improved using a larger training set.

6.7 SUMMARY

The best results obtained for networks with no hidden units were obtained using 120 input units and one output unit. Two output units were used in the experiments but the resulting network was equivalent to two networks of 120 input units and 1 output unit each. The data was presented in the *bar* format, with twelve quantisation levels for each of the ten intonation and perturbation parameters. The network was trained using 39 control speakers and 28 pathological speakers. Testing was performed using 39 control speakers and 25 pathological speakers. In this case 88.0% of the pathological speakers were classified as pathological and 12.0% classified as healthy. 84.6% of the healthy speakers were classified as healthy and 15.4% were classified as pathological.

The worst results for networks with no hidden units and trained using 39 control speakers and 28 pathological speakers were obtained for a network with 120 input units and 1 output unit. The data was presented in the *slide* format, with

twelve quantisation levels for each of the ten intonation and perturbation parameters. Testing was performed using 39 control speakers and 25 pathological speakers. In this case 64.0% of pathological speakers were classified as pathological and 44.0% were classified as healthy. 84.6% of healthy speakers were classified as healthy and 15.4% classified as pathological.

The single-layer networks trained on data with 12 quantisation levels per parameter generally did not perform as well in generalising to the test data as networks trained using data with 6 quantisation levels per parameter. This would imply that the larger networks were not able to make use of the finer resolution, and were not able to form suitable feature detectors. In these experiments the 120 input networks have more degrees of freedom than the 60 input networks and thus a larger search space.

The network topology which gave the best classification rates had 120 input units, 2 hidden units and 2 output units. The connections were constrained so that there were no connections from the input units to the output units and also no connections between the hidden units. This required the data to be quantised to 12 levels and be presented in the *bar* format. The network was trained using 39 control speakers and 28 pathological speakers. Testing was performed using 39 control speakers and 25 pathological speakers. This network correctly classified 88.0% of the pathological speakers and classified only 8.0% as being healthy. A total of 89.7% of the healthy speakers were correctly classified as healthy, and 10.3% classified as pathological.

The worst results obtained for networks with hidden units excluding those with restricted receptive fields were obtained using a network with 120 input units, 8 hidden units and 2 output units. The data was presented in the *dot* format, with twelve quantisation levels for each of the ten intonation and perturbation parameters. The network was trained using 39 control speakers and 28 pathological speakers. Testing was performed using 39 control speakers and 25 pathological speakers. In this case 56% of the pathological speakers were

classified as pathological and 44.0% classified as healthy. 89.7% of the healthy speakers were classified as healthy and 10.3% classified as having pathological voices.

The various weight maps produced clearly indicate that higher levels of perturbation are associated with pathological speakers. However, some pathological individuals appeared to be distinguished from the control group by virtue of having lower than *normal* levels of perturbation. For parameters (J-DEVEX) and (S-AVEX) there is an indication that relatively large levels of perturbation are present for speakers within the control group. This implies that either the control group may in fact contain speakers with laryngeal disorders, or that healthy speakers have high values of J-DEVEX and S-AVEX. Because no laryngeal examinations were made of the speakers in the control group it is quite possible that some of these speakers may have had undetected laryngeal pathologies or functional disorders.

Most of the lesions in the pathological group involve some degree of mass increase, which would be expected to lower the F0-AV parameter. However, the weight values for the intonational parameters indicate that F0-AV is higher than normal in the pathological group. This suggests that many of the disorders involve an increase in stiffness, which might balance the mechanical consequences of mass increase.

Generally, miss-classifications are reduced by perturbing the quantisation level for each parameter by one level with a probability of between 5% and 20%. The noise approach used here, appears to expand the training pattern set, by using the original patterns as the basis for perturbation of the parameter quantisation levels by one step. It would appear that networks with hidden units had better ability in capturing the underlying features in the training data used when this noisy clamping approach was used.

The results indicate that networks with hidden units, and one receptive field trained with data transformed to 12 quantisation levels for the three formats, are capable of achieving better performance in correct identification of pathological

voices than networks trained with data transformed to 6 quantisation levels. This does nevertheless give rise to a slight degradation in performance for the control speakers. However, for networks with no hidden units, there is little difference in performance for pathological classification using either 6 or 12 quantisation levels. But at the same time, the control speaker classification performance was generally better by using only 6 quantisation levels rather than 12. This suggests that the hidden units may be developing feature detectors for pathology classification, and to a limited extent for control classification, where higher parameter resolution is required to enable the features to be adequately distinguished. For the single-layer networks, the higher degree of overlap between patterns at the lower resolution level rather than the higher resolution level may have allowed the network to learn a coarse representation of the control data, which then gave better results on the control test data set due to its greater homogeneity.

Networks with hidden units tended to make more use of the lower quantisation levels than networks without hidden units. However, the hidden unit networks generally have many more weights than used in the networks with no hidden units.

The number of weights in the networks with one hidden unit per receptive field is very similar to the number of weights used in a 120 input unit single layer network. Results for the receptive field network are not as good as those for the single-layer nets. It should be noted that none of the receptive field networks simulated were able to correctly learn all of the training data. This would imply that the receptive field restrictions did not allow the networks to develop a good set of feature detectors.

The results are poor for the restricted receptive field type of network connectivity, being comparable with being random selections. For all the networks considered in this topology and for all the data formats, it was not possible to learn all the training patterns. This would obviously have a detrimental affect on the classification of patterns from the testing data set. The use of two hidden units

rather than one for each parameter did not generally result in any real improvements. The use of the restricted receptive fields appears to make the learning problem harder here. It is also possible that some of the weights may have become too large, thus preventing the network from reaching equilibrium and allow the network to develop good feature detectors with this training data.

Networks trained using four selected parameters performed only marginally better (some were worse) than random for the detection of laryngeal pathologies. Of the four groups of parameters considered the group containing the parameters F0-AV, J-AVEX, S-RATEX and S-DPF performed the best for a network of two hidden units and two output units and 6 quantisation levels for each parameter. 68.0% correct classification of pathological speakers was achieved with 20.0% classified as healthy. 82.1% of healthy speakers were correctly classified with 10.3% classified as pathological. The results for the selected parameter networks fall a long way short of the results obtained using all the parameters. The selection technique for the parameters was based on an inspection of the weight maps developed by networks using all the parameters. Unless the weights are controlled suitably during learning, there is a tendency for the weights to become large for patterns already learned. If this occurred, then the large weights formed would not necessarily have reflected the most significant parameters.

Generally, the results for healthy speaker classification did not vary greatly for different data formats. However, much greater differences in results for the pathology classification were observed. The *bar* format gave the best results for healthy/pathological discrimination. This format only allowed very small weights to be developed to the lower levels of intonation and perturbation parameters due to the high degree of overlap for patterns in this area. This may well have forced the networks to develop better feature detectors using the higher perturbation levels.

7 PATHOLOGY IDENTIFICATION

7 PATHOLOGY IDENTIFICATION

7.1 INTRODUCTION

The imprecise nature of some of the medical diagnosis for the pathological subjects meant that only rather broad classifications of pathology types appeared possible. The four classes of pathology considered are shown in Table 7.1.

Class	Disorder	Train	Test
1	Epithelial disorders	8	7
2	Lamina propria disorders	11	11
3	Disorders of the cartilaginous area	3	2
4	Palsies	6	5

Table 7.1 Pathology classes diagnosed in the training and test data set.

In medical terms it is not important if some benign laryngeal disorders are missed by a screening system, however it is important that all cases of cancer or potentially precancerous states should be detected. These mostly arise in the epithelium, so an ideal screening device would pick up all changes in the epithelium (Beck, 1988). Every network investigated in the previous chapter for healthy/pathological speaker discrimination misclassified some of the pathological speakers as healthy. Disorders that were misclassified included those of the epithelium. This indicates that if it is not possible to correctly distinguish these speakers for just two class classification, it will still not be possible to distinguish them correctly if further classes of pathology types are added. It was however decided to investigate how successfully the correctly classified pathological speakers could in fact be discriminated into various groups of pathological disorder.

To investigate the ability of the Boltzmann Machine in classifying the pathologies into the four classes shown in Table 7.1 experiments were conducted with

networks, both with and without hidden units. The training data comprised 28 pathological speakers and the testing data comprised 25 pathological speakers. The number of subjects for each of the four pathology classes is indicated in Table 7.1.

The data was transformed to various binary formats as discussed in section 6.2. As there are four pathology classes, the minimum number of output units required is two. However, since it may not always be easy to interpret the weight maps obtained for coding the four classes with two output units, networks were trained with both two and four output units.

7.2 SIMULATIONS

7.2.1 No hidden units

The simulator was operated as discussed in section 6.4.2 for single-layer networks. Initially, all the pathology data was used to train single-layer networks with two or four output units. Further experiments used just the training set for the training, and the networks were evaluated using the testing data set. In each case, all the intonation and perturbation parameters were used with quantisation to 6 or 12 levels.

7.2.2 Hidden units

Experiments were also conducted using networks with sixty input units (all the intonation and perturbation parameters quantised to 6 levels) and two hidden units. Connections between input and output units were allowed in addition to the connection of the hidden units to both the input and the output units. No connections were allowed between the output units. The annealing schedule and weight adaptation regime used followed the schemes described in section 6.4.2.

7.3 RESULTS

7.3.1 No hidden units

With all the pathology data transformed to 6 quantisation levels the only network structure that was able to learn the complete pathology data set was the four output unit net, and this required that the data be represented in the *dot* format. In this case there was far too much pattern overlap to enable the *slide* and *bar* formatted data to be learned correctly.

With all the training data transformed to 12 quantisation levels it became possible for networks with either two or four output units to learn the training data for all three formats. The number of learning cycles required for the *dot* format was approximately 25 while for the slide format the number of learning cycles was approximately 170 for a network with either two or four output units. For data in the *bar* format, nets with four output units took approximately 400 learning cycles while nets with two output units took approximately 500 learning cycles.

The weights developed for networks with four output units trained with data transformed to 12 quantisation levels are shown in Figures 7.1 to 7.3. The leftmost output unit represents class 1 whereas the rightmost output unit represents class 4. Examining the weights learned for networks with data present in the *dot* and *slide* formats indicates that epithelial disorders appear to be characterised predominantly by higher than normal levels of parameters 1 (F0-AV), 2 (F0-DEV), 4 (J-AVEX), 5 (J-RATEX) and midrange values of parameters 7 (S-DEVEX) and 9 (S-RATEX).

Lamina propria disorders appear to be characterised predominantly by midrange values of parameters 2 (F0-DEV), 7 (S-DEVEX) and 9 (S-RATEX). Cartilaginous disorders appear to be characterised predominantly by midrange values for parameters 2 (F0-DEV), 4 (J-AVEX), 5 (J-RATEX) and 6 (J-DPF) and also by high and low values for parameter 8 (S-AVEX).

Palsies appear to be predominantly characterised by high values for parameters 2 (F0-DEV) and 10 (S-DPF); midrange values for parameters 4 (J-AVEX) and 5 (J-RATEX); low values for parameters 3 (J-DEVEX) and 6 (J-DPF).

The networks trained with data in the *bar* format present differing solutions to weights learned. As for experiments undertaken distinguishing between control and pathological speakers, networks trained with data in the *bar* format do not make very effective use of the lower quantisation levels.

All the pathology groups appear to be associated with increased levels of perturbation. This fits with the theory that normal vocal fold vibration is very sensitive to changes in the mechanical state of the ligamental portion of the vocal folds, but that alterations in the cartilaginous portion have much less effect on vocal fold vibration. This may also be seen to a certain extent in the weight maps by the development of smaller weights overall for cartilaginous disorders compared to the other three classes of laryngeal disorder investigated.

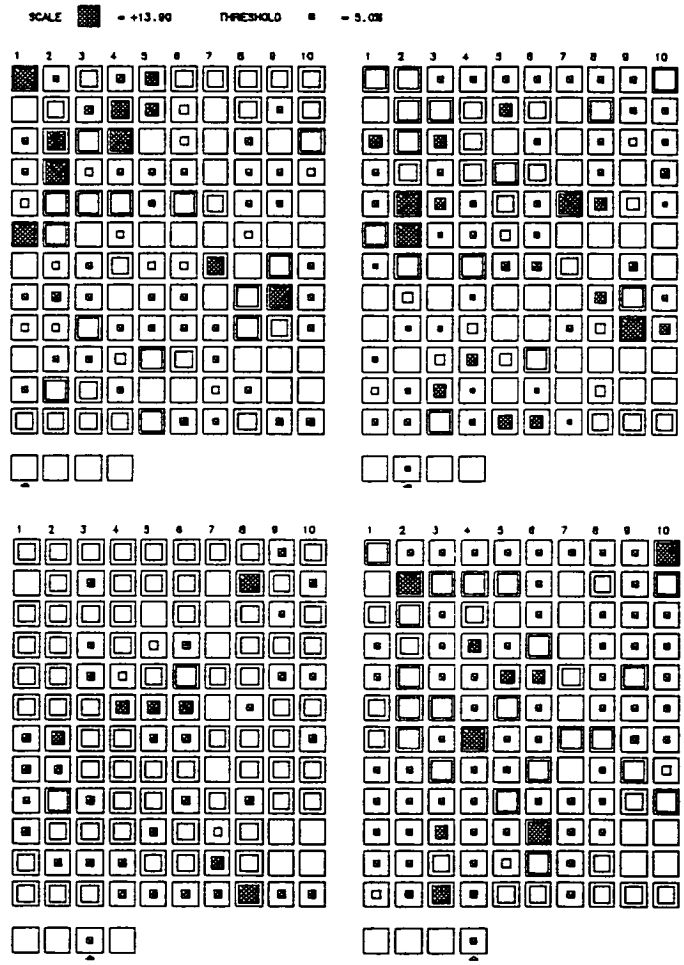

Figure 7.1 Weights learned by single layer network with four output units. Learning data comprised 53 pathological voices in the *dot* format, with 12 quantisation levels per parameter.

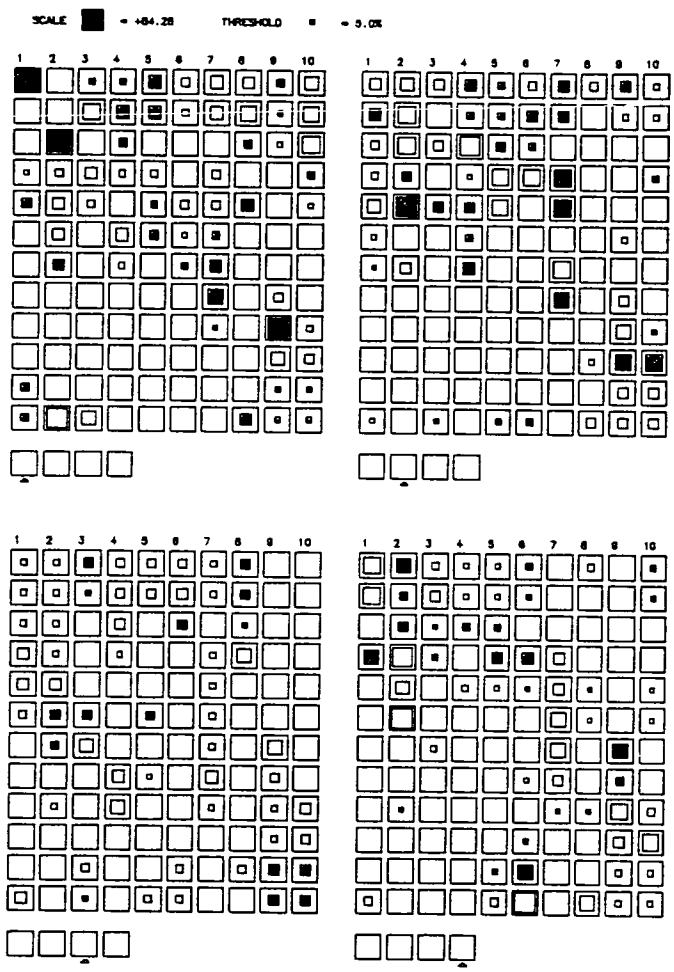

Figure 7.2 Weights learned by single layer network with four output units. Learning data comprised 53 pathological voices in the *slide* format, with 12 quantisation levels per parameter.

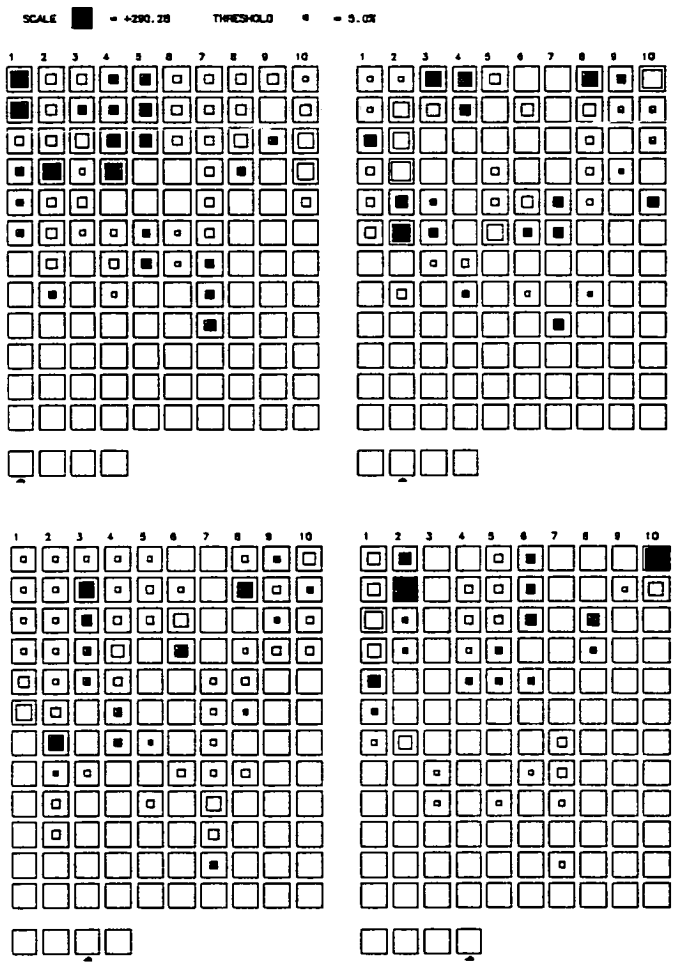

Figure 7.3 Weights learned by single layer network with four output units. Learning data comprised 53 pathological voices in the *bar* format, with 12 quantisation levels per parameter.

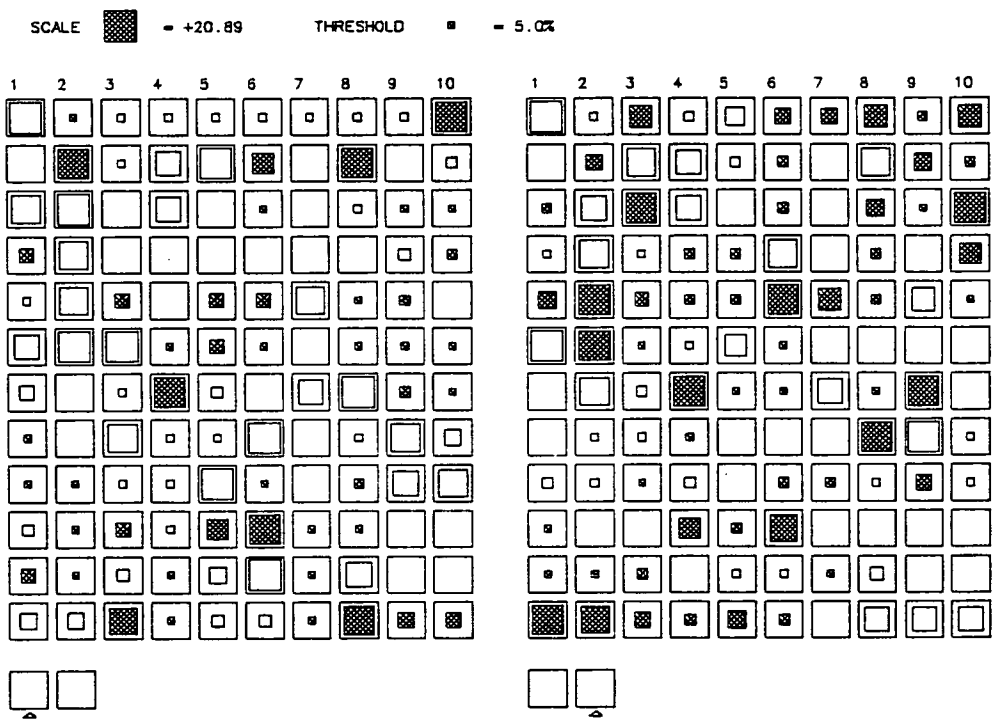

Figure 7.4 Weights learned by single layer network with two output units. Learning data comprised 53 pathological voices in the *dot* format, with 12 quantisation levels per parameter.

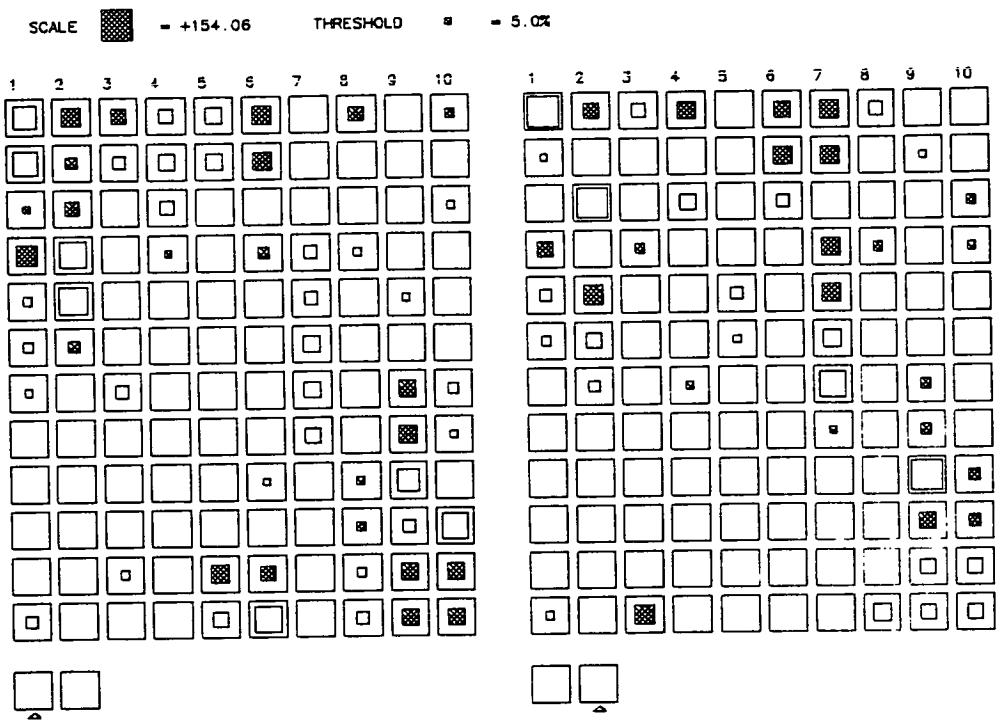

Figure 7.5 Weights learned by single layer network with two output units. Learning data comprised 53 pathological voices in the *slide* format, with 12 quantisation levels per parameter.

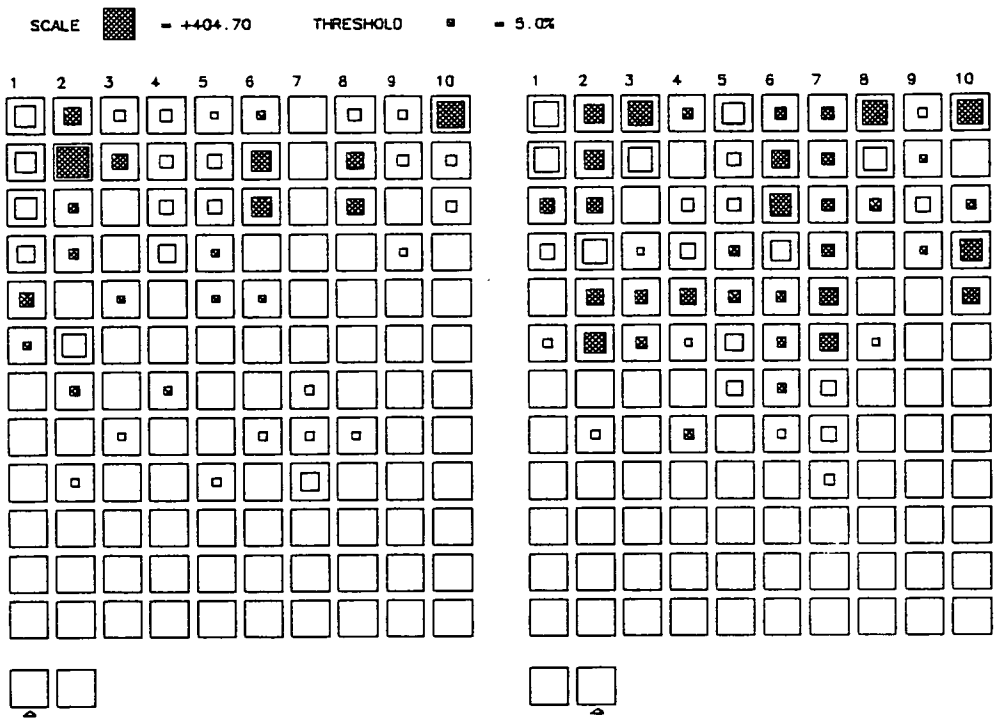

Figure 7.6 Weights learned by single layer network with two output units. Learning data comprised 53 pathological voices in the *bar* format, with 12 quantisation levels per parameter.

Figures 7.4 to 7.6 depict the weights learned by the networks with just two output units. The patterns of weights developed are harder to interpret than those developed with four output units. The output unit states for each of the classes is zero for epithelial disorders and three (binary) for palsies. As for the nets with four output units, as the data representation becomes coarser, fewer weights, as indicated above the 5% threshold value are required by the networks.

Tables 7.2 to 7.4 show the results obtained for single-layer networks with two output units trained with the 28 pathological subject training set and tested with the 25 pathological subject test set. The overall results for networks with four output units were not as good as those with two output units.

Test Class	% Subjects classified as:			
	1	2	3	4
1	71.4%	14.3%	0.0%	14.3%
2	36.4%	63.6%	0.0%	0.0%
3	50.0%	50.0%	0.0%	0.0%
4	20.0%	80.0%	0.0%	0.0%

Table 7.2 Results for single-layer network with two output units and data in *dot* format quantised to 12 levels.

Test Class	% Subjects classified as:			
	1	2	3	4
1	57.1%	42.9%	0.0%	14.3%
2	27.3%	54.5%	9.1%	9.1%
3	0.0%	100.0%	0.0%	0.0%
4	20.0%	80.0%	0.0%	0.0%

Table 7.3 Results for single-layer network with two output units and data in *slide* format quantised to 12 levels.

Test Class	% Subjects classified as:			
	1	2	3	4
1	57.1%	0.0%	0.0%	42.9%
2	27.3%	27.3%	9.1%	18.2%
3	50.0%	50.0%	0.0%	0.0%
4	20.0%	40.0%	0.0%	40.0%

Table 7.4 Results for single-layer network with two output units and data in *bar* format quantised to 12 levels.

The number of members in class three of the training data set was only three and resulted in it not being possible for the class three subjects to be correctly classified from the testing set. Performance for class 4 was also very poor. The correct identification of classes 1 and 2 was better for data in the *dot* format than either the *slide* or *bar* formats. The quantity of training data used for each of the groups appears to be insufficient in enabling the networks to develop the requisite feature detectors. The three different coding schemes used produced widely differing results, again suggesting that the networks were completely unable to develop suitable feature detectors.

Overall the number of subjects correctly classified from the test group for data in the *dot* format was 48.0%, for data in *slide* format 40.0% and in the *bar* format 36%.

7.3.2 Hidden units

The addition of two hidden units to the single-layer networks with sixty input units and either two or four output units enabled all the training data to be learned for all data formats. Overall results for the test data set for networks with two output units were with the *dot* format 32% of the test set subjects correctly classified, with the *slide* format 28% of the test subjects correctly classified, whereas with the *bar* format 40% were correctly classified.

Figure 7.7 depicts the weights learned for a network with two output units trained with data in the *dot* format. Figure 7.8 depicts the weights learned for a network with four output units with the training data also in the *dot* format.

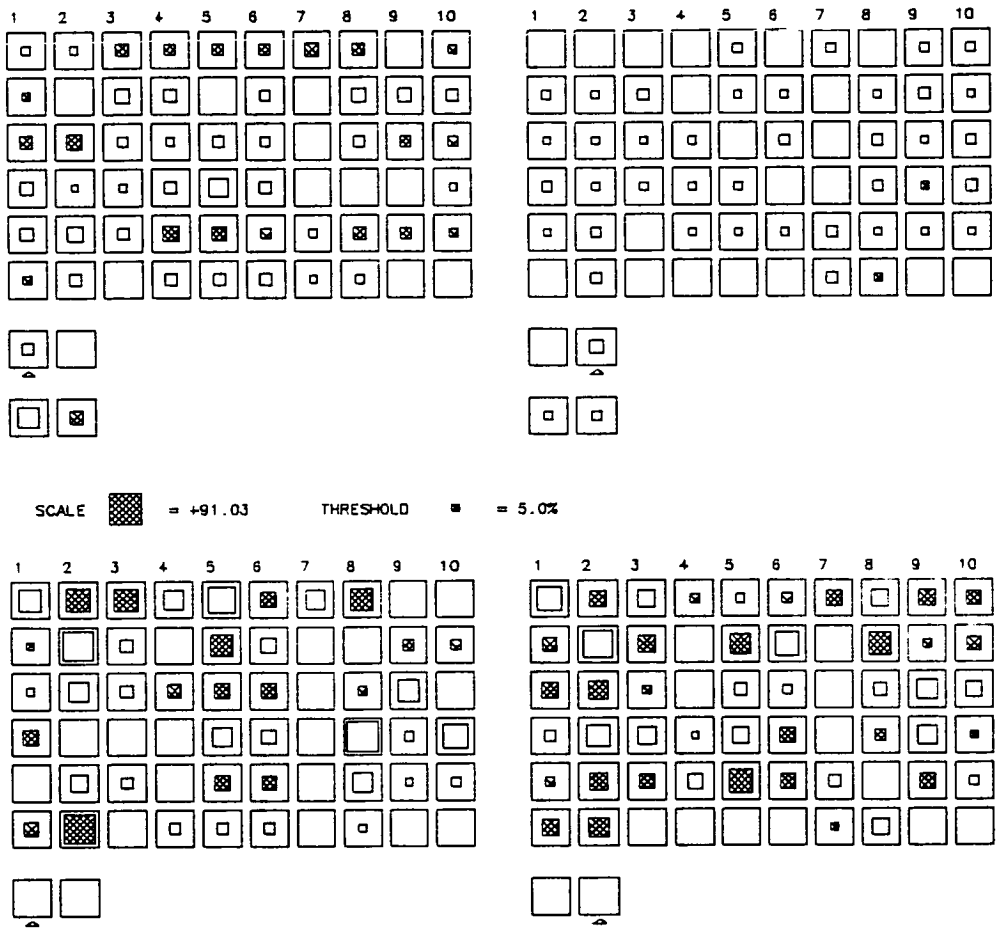

Figure 7.7 Weights learned by network with two hidden and two output units. Learning data comprised 28 pathological voices in the *dot* format, with 6 quantisation levels per parameter.

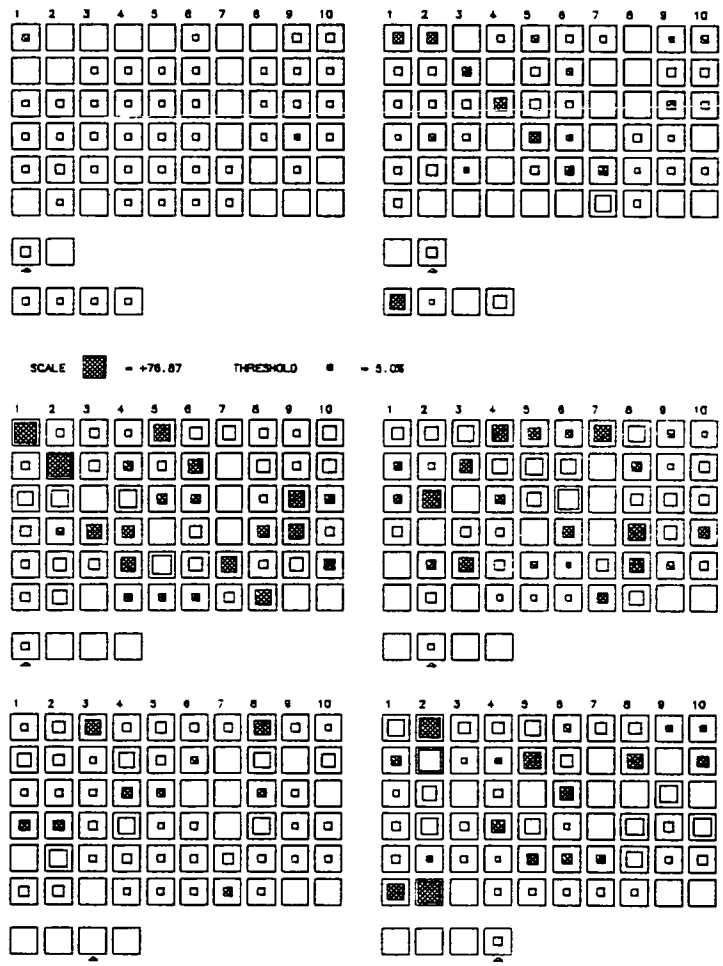

Figure 7.8 Weights learned by network with two hidden and four output units. Learning data comprised 28 pathological voices in the *dot* format, with 6 quantisation levels per parameter.

An investigation of the weights learned by networks with two output units trained with data in the *bar* format indicated that the hidden units made a large number of very small connections to the input units. The size of these weights was less than 5% of the largest weight developed in the network, and consequently they are too small to reproduce in the form of the weight maps used. Many of these weights made connections to the lower quantisation levels, and thus appears that they are able to use more of the parameter range than the networks without hidden units trained on the same data.

7.4 COMPARISON WITH RESULTS FROM PREVIOUS WORK

Beck (1988) reports qualitative findings for the classification of pathological subjects into three broad classes, epithelial disorders, polyps/nodules and disorders of the cartilaginous area. The findings are based on averaged raw scores and Z-scores for the intonation and perturbation parameters. Epithelial disorders were reported to be characterised by relatively high jitter levels, whereas polyps and nodules (disorders of the lamina propria) and also disorders of the cartilaginous region tended to be characterised by higher shimmer levels. As far as the author is aware there is no quantitative evidence for the discrimination of groups of pathology using intonation and perturbation parameters.

From examinations of the various weight maps produced it may be clearly seen that epithelial disorders are characterised by high levels of jitter. The various disorders of the cartilaginous region appear to be characterised by either high or low levels of shimmer. Some lamina propria disorders do appear to be characterised by high shimmer levels, although strong connections are also made to lower values of parameters 7 (S-DEVEX) and 9 (S-RATEX).

The Boltzmann Machine approach is at present severely limited by the amount of data available. The discrimination of groups of pathologies does not look very promising from the results obtained so far, particularly for identifying changes in the epithelium which may indicate cancer or precancerous states. An ideal classifier for use as a screening tool need only discriminate between healthy and

pathological voices. Speakers classified as pathological would then be referred to a laryngologist who would then use a further classifier capable of discriminating between the various groups of disorders in attempting to make a diagnosis of the suspected laryngeal disorder.

8 CONCLUSION

8 CONCLUSION

8.1 Review of Results Presented in Thesis

In this thesis the results obtained by applying Boltzmann Machines to the problem of determining whether speakers have healthy or pathological voices as evidenced by intonation and perturbation parameters have been presented. In addition, results obtained by applying them to the estimation of the type of pathology have also been presented. For the healthy/pathological speaker discrimination problem the Boltzmann Machine structures using all of the ten intonation and perturbation parameters with and without hidden units, but with no restricted receptive fields, are considered first.

For healthy/pathological speaker classification the best results were obtained using a network with 120 input units, 2 hidden units and 2 output units. The topology of this particular network was constrained so that there were no connections from the input units to the output units and also no connections between the hidden units. The data was presented in the *bar* format, with twelve quantisation levels for each of the ten intonation and perturbation parameters. The network was trained using 39 control speakers and 28 pathological speakers. Testing was performed using 39 control speakers and 25 pathological speakers. Results were 88.0% of the pathological speakers correctly classified as pathological and 8.0% classified as healthy. One pathological speaker could not be discriminated as belonging to either of the two classes. A total of 89.7% of the healthy speakers were correctly classified as healthy and 10.3% classified as pathological.

Using a network with no hidden units the best results were obtained using 120 input units and one output unit. Two output units were used in the experiments but the resulting network was equivalent to two networks of 120 input units and 1 output unit each. The data was presented in the *bar* format, with twelve quantisation levels for each of the ten intonation and perturbation parameters. The network was trained using 39 control speakers and 28 pathological

speakers. Testing was performed using 39 control speakers and 25 pathological speakers. In this case 88.0% of the pathological speakers were classified as pathological and 12.0% classified as healthy. 84.6% of the healthy speakers were classified as healthy and 15.4% were classified as pathological.

Other network configurations discussed in Chapter 6 provided poorer results for healthy/pathological speaker discrimination which are very nearly random. However, even for the best results, the false alarm rate is too high to enable satisfactory use of these Boltzmann Machines as a screening tool. It must of course be noted that since control speakers were not given laryngeal examinations a proportion of the false positives may actually have had undetected laryngeal pathologies or functional disorders. Indeed, a number of the control speakers were later diagnosed as having laryngeal pathologies. This happened well after the end of the data collection exercise and a detailed follow-up was never made possible.

The effect of training with noise has generally been to improve the classification results. It would therefore appear that the addition of noise to the training data is successful in expanding the training data set. However it should be noted that the noise technique was not that of adding noise randomly to the data set, but that of perturbing the quantisation level for each parameter by one level with a given probability. Thus over a number of noisy presentations of the training data the mean of the distribution will coincide with the original data. With the addition of random noise to the training data one would expect performance to degrade. However by perturbing the quantisation levels by a small amount appears to effectively increase the training data set, and hence the ability to generalise.

Generally, better results were obtained for the correct classification of healthy speakers rather than for the correct classification of pathological speakers. This may have been affected by the fact that there was in most instances more training data used for the healthy speakers than for the pathological speakers. However, with the small training set of ten control and ten pathological speakers the results for healthy speaker classification were still better than those for pathological

classification. This suggests that the composition of the control group was fairly homogeneous, with a high degree of correlation between speakers. The pathological speaker group is likely to be rather heterogeneous due to the number of different pathologies present.

From the weight maps it was possible to observe that higher levels of perturbation play a significant role in identifying pathological speakers, as strong connections are made to inputs corresponding to these higher levels. It was also observed that high levels of values for parameters J-DEVEX and S-AVEX were present for speakers within the control group. As the control group had not been subjected to a laryngological examination it is quite possible that the healthy group did in fact contain speakers with the early stages of disorders present. It was also observed that for some of the pathological speakers lower than normal levels of perturbation were present.

Networks trained using four selected parameters performed only marginally better (some were worse) than random for the detection of laryngeal pathologies. Of the four groups of parameters considered the group containing the parameters F0-AV, J-AVEX, S-RATEX and S-DPF performed the best for a network of two hidden units and two output units and 6 quantisation levels for each parameter. A total of 68.0% of the pathological speakers were correctly classified with 20.0% being classified as healthy. For healthy speakers 82.1% were correctly classified with 10.3% classified as pathological. The results for the selected parameter networks fall a long way short of the results obtained using all ten parameters. The selection technique for the parameters was based on an inspection of the weight maps developed by networks using all the parameters. Unless the weights are controlled suitably during learning, it is possible for the weights to become large for patterns already learned. If this occurred, then the large weights formed would not necessarily reflect the most significant parameters.

Results obtained for networks with intonation, shimmer and jitter receptive fields were disappointing. It was not possible for the networks considered to correctly learn all of the training data within the 500 learning cycles allowed. Incorporating

an additional hidden unit per restricted receptive field yielded no overall improvements in learning generalisation. This suggests that the use of restricted receptive fields makes the learning problem harder. It is also possible in this case that weights have become too large so that equilibrium cannot be achieved. This would result in the network being unable to learn all of the training patterns or make use of additional hidden units.

The results for the use of ten intonation and perturbation parameter receptive fields were even worse than those obtained for only three receptive fields. Again it was not possible for these types of networks to correctly learn all the data and the observations made for the intonation, shimmer and jitter receptive field networks hold.

The investigation into the use of the Boltzmann Machine for pathology group identification showed that from the data used it was not possible to discriminate between disorders of the cartilaginous area, palsies, lamina propria disorders and epithelial disorders. This result may be due to the limited data set used, the data format, and the choice of sub-groups. Or indeed it may be that it is not possible to differentiate between these groups of disorder. Further work is required here. From a medical point of view it is not important if some benign laryngeal disorders are missed, but it is important that all cases of cancer or potentially precancerous states should be detected. These mostly arise in the epithelium so an ideal screening tool would pick up all changes in the epithelium.

The data formats used affected the results for pathological classifications. For example for networks with 120 input units, 2 hidden units and two output units the results for correct pathological classification are *dot* 68.0%, *slide* 80.0% and *bar* 88.0%. Results for correct healthy speaker classification are *dot* 84.6%, *slide* 89.7% and *bar* 89.7%. Generally the results for healthy classification did not vary a great deal for different data formats. However, much greater differences in results for the pathological classification were observed. This suggests that the transformations were not affecting the control data to a great extent, but did affect the pathological data. One would expect the data for the control speakers to be

centred approximately mid-range for each parameter and the pathological data to be at the extremities of the range. This may have introduced boundary problems. Furthermore, it may be that due to the large number of laryngeal disorders present higher resolution of data representation would be beneficial. Interestingly, the *bar* format gave the best results for discriminating between healthy and pathological voices. It can be observed from the weight maps that no weights or at least very small weights were developed for lower levels of the intonation and perturbation parameters. This would suggest that the high degree of overlap for the *bar* format for the low levels of intonation and perturbation may have caused the network to ignore features in this area, and forced the network to develop feature detectors using the higher perturbation levels.

In all cases, the data representation format affected the number of learning cycles required to obtain a given error in learning of the training data. The higher the degree of overlap between patterns the more learning cycles were required. The identification of control speakers was much better than the desired identification of pathological speakers. A possible explanation for this is that it is due to the heterogeneous composition of the pathological speaker group and the fact that approximately 40% more subjects were used in the control speaker training group than in the pathological speaker training group.

Results obtained with a representation of 6 quantisation levels per parameter did not perform as well as when a representation of 12 quantisation levels per parameter was used for pathological classification for networks with hidden units (excluding restricted receptive field models). However performance for networks with no hidden units was very similar between the two resolutions for pathological classification. Results for 6 levels with the dot and slide formats were better than those for 12 levels, whereas the bar format was better with 12 levels.

The scope of the experiments was severely limited by the amount of data available. There are enormous problems in acquiring good voice recordings and adequate laryngeal observations simultaneously. In addition since control

subjects were not given laryngeal examinations a proportion of the false positives reported may actually have had minor laryngeal pathologies or functional disorders.

The use of training data in binary format and the small number of classification classes for this data suggest that in many cases it was possible for the networks studied to *memorise* the training data, rather than to generalise to the test set data. However, generalisation with these networks was in many cases better than those with restricted connectivities, where it was hoped to force the network to generalise. This was particularly so for the control speakers and would suggest that there is a high degree of correlation between the intonation and perturbation parameters for the control speakers.

The Boltzmann Machine has been shown to offer only a marginal improvement in classification accuracy over the use of linear discriminant techniques for distinguishing between control and pathological speakers, but overall at much greater computational expense. As a technique for learning with networks of hidden units it is very slow, due to the time required for annealing and collecting co-occurrence statistics. However, once a network has been trained, applying a test pattern to the network and searching for an estimate of class index is very fast compared to the training time.

It was concluded that the intonation and perturbation parameters used were useful for differentiating between groups of healthy speakers and speakers with known pathological conditions of the larynx using a Boltzmann Machine. However, the performance of such a system using a Boltzmann Machine is not yet good enough to be used as a screening procedure. From interpreting the various weight maps of the weights developed by the networks studied, pathological voices could generally be seen to be characterised by higher than normal levels of perturbation. However, some pathological speakers did indicate lower than normal levels of perturbation.

8.2 Comparison to Other Work

Results for healthy/pathological voice discrimination of male speakers using data from the pool used for experiments described in this thesis have been reported by Laver, Hiller, Mackenzie & Rooney (1986) and Beck (1988). Laver et al. used a control group of 63 speakers and a pathological group of 55 speakers. Two techniques were presented, the first of these being the use of bivariate plots. Principal components analysis was applied to the F0-AV and S-DPF data for the control group to give an ellipse (at the 2 SD level) indicating the covariance between the parameters. The boundary of the ellipse formed the screening threshold boundary for the detection of pathology. Using this technique 90.1% of the pathological speakers were correctly classified as pathological and 9.5% were classified as healthy.

The second technique used linear discriminant analysis, which is a statistical technique for discriminating between two (or more) nominal groups on the basis of several parameters simultaneously. Using this technique 85.5% of pathological speakers were classified as pathological, and 14.5% were classified as healthy.

Beck (1988) however suggested that discriminant analysis was probably the best screening option from a number of techniques examined. These included using bivariate plots to compare pathological speakers with control group distributions and a pattern discrimination technique based on the maximum likelihood principle. Beck used 83 control male speakers and 56 pathological male speakers. For the bivariate technique using F0-AV versus S-DPF, results of 80.4% of pathological speakers being correctly classified as pathological and 10.8% being classified as healthy were obtained. Using linear discriminant analysis, with ten intonation and perturbation parameters 87.5% of pathological speakers were correctly classified as pathological and 8.4% were classified as healthy.

As Beck points out, the results of the discriminant analysis technique need to be treated with caution. Linear discriminant analysis assumes that the data show a multivariate normal distribution, but given the heterogeneous composition of the pathological group it is likely that this assumption is seriously violated. However,

the technique appeared to be quite robust in the face of such violations. The classification rates obtained, however, cannot safely be asserted to be necessarily predictive of future success in classifying another set of subjects with the same function.

The above techniques have all used within group testing whereas the Boltzmann Machine experiments have attempted classification on a test set of subjects. By training a single-layer network on the data in both the training and test set it is possible to achieve 100% correct classification of the healthy and pathological voices in the total training set. It was not possible to determine whether the patterns had just been memorised or whether generalisation had taken place. In any instance, one could not safely assert that the classifier be capable of such success in classifying another set of subjects with the same function.

Beck (1988) reports qualitative findings using averaged raw scores and Z scores for three broad classifications of pathology. These were epithelial disorders, polyps/nodules and disorders of the cartilaginous area. For males epithelial disorders of ligamental region appeared to be characterised by relatively higher jitter scores, whilst polyps, nodules and disorders of the cartilaginous regions appeared to have higher shimmer scores.

As far as the author is aware there is no quantitative evidence for the discrimination of groups of pathology using intonation and perturbation parameters. Results to-date using Boltzmann Machines suggest that it is unlikely to be successful.

8.3 Areas for Further Work

As mentioned earlier in this chapter the scope of the experiments was severely limited by the amount of data available. Furthermore, the healthy speakers were not subjected to a laryngological examination. The experiments were also only undertaken using intonation and perturbation data from male speakers. However, any screening system produced should be able to cope with both male

and female speakers. Despite there being enormous problems in acquiring good voice recordings and adequate laryngeal observations simultaneously they are both essential if further studies are to be made.

The pathological data group should include samples from as wide a range of disorders as possible. It should also include graded samples of a wide range of disorders at different degrees of severity from both adult males, adult females and also children. The control group database speakers should be matched in sex, age, socioeconomic status, general physique and general state of health to the speakers in the pathological group.

The application of the Boltzmann Machine to the discrimination of healthy or pathological speakers as evidenced by intonation and perturbation parameters has been demonstrated. Further work applying neuromorphic systems requires an expanded set of training examples, including the addition of data from female speakers. The investigations conducted made use of a binary representation of the data. The use of networks that can use continuous valued data would eliminate limitations in the binary transformation carried out and would seem worthwhile. Almeida (1987a) has suggested that back-propagation can be used to train Boltzmann Machines and would thus allow the speed of learning to be considerably improved. There is a fairly high level of processing of the raw speech data to obtain the ten acoustic parameters for the various healthy and pathological speakers. It would seem worthwhile to attempt to discriminate between healthy and pathological speakers by applying their digitised raw speech data to a network. This would have possible problems in that the network would need to handle a stream of input data. However, this approach would benefit from not being hampered by any restrictions that the choice of the ten intonation and perturbation parameters maybe imposing on the discrimination task.

Pattern recognition devices implemented using the Boltzmann Machine or Multi-layer Perceptron require supervised learning. In other words the device must be presented with labelled patterns so that it can learn the mapping between the feature space and the classification space. However, it would be useful to

know how the ensemble of patterns observed in healthy/pathological speaker discrimination is distributed in pattern space. If the mechanism giving rise to the patterns also segregates them into clusters in a meaningful manner, then any procedure that identifies the location and distribution of these clusters is also meaningful. Some neural net algorithms have been suggested for cluster formation, these are the Adaptive Resonance Theory (ART) structure (Carpenter & Grossberg 1987) and the self organising neural array, Kohonen (1982, 1987) and would appear worthy of further examination.

REFERENCES

REFERENCES

Akiyama Y., Yamashita A., Kajiura M. & Aiso H. (1989). Combinatorial Optimisation with Gaussian Machines. Proceedings of 1989 IEEE Int. Conf. on Neural Networks.

Aleksander I. (1989). Neural Computing Architectures. North Oxford Academic.

Almeida L. (1987b). A learning rule for asynchronous perceptrons with feedback in a combinatorial environment. Proc. of 1987 IEEE First Ann. Int. Conf. on Neural Networks, S. Diego.

Almeida L. (1987a). Backpropagation in perceptrons with feedback in neural computers. Proc. of the NATO ARW on Neural Computers, Dusseldorf (Heidelberg: Springer Verlag, 1987).

Alspector J. & Allen R. B. (1987). A Neuromorphic VLSI Learning System. Advanced Research in VLSI: Proc. of the 1987 Stanford Conf., P. Losleben, ed., pp313-349, Cambridge, MA. MIT Press.

Alspector J. (1989). Neural-Style Microsystems that Learn. IEEE Communications Magazine, November 1989.

Amari S. A. (1977a). A mathematical approach to neural systems. In Metzler J., (Ed.), Systems Neuroscience (pp. 67-117). New York: Academic Press.

Amari S. A. (1977b). Neural theory of association and concept formation. Biological Cybernetics, 26, 175-185.

Amari S. I. & Akikazu T. (1978). Biol. Cybern. 29, 127-136.

Amit D. J., Gutfreund H., & Sompolinsky H. (1985a). Spin-glass models of neural networks. Physical Reviews, A2, 1007-1018.

Amit D. J., Gutfreund H., & Sompolinsky H. (1985b). Storing infinite numbers of patterns in a spin-glass model of neural networks. Physical Review Letters, 55, 1530-1533.

Amit D. J., Gutfreund H., & Sompolinsky H. (1987). Ann. Phys, 173, 30.

Arnold G. E. (1962). Vocal nodules and polyps: laryngeal tissue reaction to habitual hyperkinetic dysfunction. J. Speech and Hearing Res., 27, 205-216.

Aronson A. E. (1980). Clinical Voice Disorders. An Interdisciplinary Approach. New York: Thieme-Stratton Inc..

Baer T. (1973). Measurement of vibration patterns of excised larynxes. JASA, 54, 318.

Baer T. (1978). Effect of single-motor-unit firings on fundamental frequency of phonation. JASA 64:S90(A).

Baer T. (1980). Vocal jitter: a neuromuscular explanation. In Lawrence V. & Weinberg B. (Eds.), Transcripts of the Eighth Symposium on: Care of the Professional Voice, Part 1: Physical Factors in Voice, Vibrato, registers, 19-24, The Voice Foundation, New York.

Baer T. (1981). Investigation of the phonatory mechanism. In Ludlow C. & Hart M. O. (Eds.) Proceedings of the Conference on the Assessment of Vocal Pathology, AHSA Reports 11:38-46., American Speech-Language-Hearing association, Rockville, MD.

Baken R. J. (1987). Clinical Measurement of Speech and Voice. College-Hill Press, Little, Brown and Company, Boston, MA.

Baldi P. & Venkatash S. (1987). Phys. Rev. Lett. 58, 913.

Baur W. C. & McGavran M. H. (1972). Carcinoma in situ and evaluation of epithelial changes in laryngopharyngeal biopsies. J. of the Amer. Med. Assoc.,221, 72-75.

Beck, J. M. M. (1988). Organic variations and voice quality. Phd thesis, University of Edinburgh.

Benjamin B. J. (1981). Frequency variability in the aged voice. J. Gerontology, 36, 722-726.

Berouti M., Childers D. G. & Paige A. (1977). Correction of tape recorder distortion. ICASSP-77, 397-400.

Birrell J. F. (1977). Logan Turner's Diseases of the Nose, Throat and Ear (8th edn). Bristol: John Wright & Sons Ltd.

Braitenburg V. (1978). Cell Assemblies in the cerebral cortex. In: Theoretical approaches to complex systems. Heim R. & Palm G. (eds.), p 171, Berlin, Heidelberg, New York: Springer

Broca P-P. (1863). Localisation des fonctions cerebrales: Siege du langage articule. Bulletin de la Societe d'anthropologie. Paris 4:200-202.

Bruce A. D.; Canning A.; Forrest B.; Gardner E. & Wallace D. J. (1986). Learning and Memory Properties in Fully Connected Networks. Proc. Conf. on Neural Networks for Computing, Snowbird, UT (AIP Conf. Proc. 151) ed J. S. Denker (New York: AIP).

Carpenter G. & Grossberg S. (1987). A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine. Computer Vision, Graphics and Image Processing, 37, 54-115.

Caton R. (1875). The electric currents of the brain. Brit. Med. J., ii, 278.

Clark E. & O'Malley C. D. (1968). The human brain and spinal cord: A historical study illustrated by writings from antiquity to the twentieth century. Berkely: University of California Press.

Cohen M. A. & Grossberg S. (1983). Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. IEEE Transactions SMC-13, 815-826.

Crystal T. H. & Jackson C. L. (1970). Extracting and Processing Vocal Pitch for Laryngeal Disorder Detection. The Journal of the Acoustical Society of America, 48, 118.

Davis S. B. (1976). Computer Evaluation of Laryngeal Pathology based on Inverse Filtering of Speech. Speech Communication Research Laboratory Monograph, 13.

Davis S. B. (1979). Acoustic characterisations of normal and pathological voices. In Lass N. J. (Ed.), Speech and Language: Advances in Basic Research and Practice, New York: Academic Press, 1, 273-338.

Derthick M. (1984). Variations on the Boltzmann Machine Learning Algorithm. (Tech. Rep. No. CMU-CS-84-120). Pittsburgh: Carnegie-Mellon University, Department of Computer Science.

Elman J. L. & Zipser D. (1987). Learning the Hidden Structure of Speech. Institute for Cognitive Science, University of California at San Diego, ICS Report 8701.

Fairbanks G. (1960). Voice and Articulation Drill Book. New York: Harper Brothers.

Feldman J. A. & Ballard D. H. (1982). Connectionist models and their properties. Cognitive Science, 6, 205-254.

Ferlito A. (1974). Histological classification of larynx and hypopharynx cancer. Acta Otolar. Suppl., 342, 17.

Forrest B. M. (1988). Content-addressability and learning in neural networks. J. Phys. A: Math. Gen. 21, 245-255.

Friedmann I. & Osborn D. A. (1978). The Larynx, in W. St. C. Symmers (Ed.), Systemic pathology, Vol. 1, 248-267.

Gardner E. & Derrida B. (1988). Optimal storage properties of neural network models. J. Phys. A21, 271.

Gardner E. (1987). Maximum storage capacity of neural networks. Europhys. Lett. 4, 481.

Geman S. & Geman D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans. on Pattern Analysis and Machine Intelligence, 6, 721-741.

Gold B. & Rabiner L. R. (1969). Parallel processing techniques for estimating pitch periods of speech in the time domain. JASA, 46, 442-448.

Gold B. (1962). Computer program for pitch extraction. JASA, 34,916-921.

Gold B. (1964). Note on buzz-hiss detection. JASA, 36, 1659-1661.

Grant P. M. & Sage J. P. (1986). A comparison of Neural network and matched filter processing for detecting lines in images. In Denker J. S. (Ed.) AIP Conference Proceedings 151, Neural Networks for Computing, Snowbird, Utah.

Greene M. C. L. (1972). The voice and its disorder. (3rd edn.) Philadelphia: Lippincott.

Grossberg S. (1976). Adaptive pattern classification and universal recoding: Part 1. Parallel development and coding of neural feature detectors. Biological Cybernetics, 23, 121-134.

Grossberg S. (1982). Studies of Mind and Brain. D. Reidal Publ. Co., Dordrecht.

Grossberg S. (1987). The Adaptive Brain. Vols I and II, North Holland.

Hall S. I. & Colman B. H. (1975). Diseases of the Nose, Throat and Ear: a handbook for students and practitioners. Edinburgh: Churchill Livingstone.

Hanson R. J. (1978). A two-state model of F0 control. J. Acoust. Soc. Am.,64, 543-544.

Hardcastle W. J. (1976) The physiology of speech production. New York: Academic Press.

Harmon L. D. (1964). In Neural Theory and Modeling, ed. Reiss R. F. (Stanford Univ. Press, Stanford, CA), 23-24.

Hebb D. (1949). Organisation of behaviour. New York: Wiley

Hecker M. & Kreul E. (1971). Descriptions of the speech of patients with cancer of the vocal folds. Part 1: measures of fundamental frequency. JASA, 49, 1275-1282.

Heilberger V. L. & Horii Y. (1982). Jitter and shimmer in sustained phonation. In Lass N. (Ed.) Speech and Language: Advances in Basic Research and Practice. Vol. 7:299-332, Academic Press, New York.

Hiki S., Sugawara K. & Oizumi J. (1968). On the Rapid Fluctuation of Voiced Pitch, Reports of the Research Institute of Electrical Communication, Tohoku University, Japan, 19, 237-239.

Hiller S. M. (1985). Automatic Acoustic Analysis of Waveform Perturbations. PhD thesis, University of Edinburgh.

Hinton G. E., Sejnowski T. J., & Ackley D. H. (1984). Boltzmann Machines: Constraint Satisfaction Networks that Learn. (Tech. Rep. No. CMU-CS-84-119). Pittsburgh, PA: Carnegie-Mellon University.

Hinton G. E. & Sejnowski T. J. (1983). Analysing cooperative computation. Proceedings of the Fifth Annual Conference of the Cognitive Science Society.

Hinton G. E. (1984). Distributed Representations. Tech. Rep. CMU-CS-84-157, Carngie-Mellon University, Department of Computer Science.

Hirano M., Kakita Y., Ohmaru K. & Kurita S. (1982). Structure and mechanical properties of the vocal fold. In N. Lass (Ed.), Speech and Language: Advances in Basic Research and Practice. New York: Academic Press, 211-297.

Hirano M. (1981). Clinical Examination of Voice. New York: Springer Verlag.

Hodgkin A. L. & Huxley A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. J. Physiol. (London), 117, 500-544.

Hopfield J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences of the United States of America, 79, 2554-2558.

Hopfield J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. Proceedings of the National Academy of Sciences of the United States of America, 81, 3088-3092.

Ising E. (1925). Z. Physik 31, 253.

Jackson J. H. (1958). On localisation. In Selected writings (Vol. 2). New York, Basic Books. (Original work published 1869).

Junge D. (1981). Nerve and muscle excitation. 2nd ed. Sunderland, Mass.: Sinauer.

Kandel E. R. & Schwartz J. H. (1981). Principles of neural science. Amsterdam: Elsevier-North Holland.

Kanerva P. (1984). Self-propagating search: a unified theory. PhD Thesis CSLI-84-7, Stanford University Centre for the Study of Language and Information.

Kaplan H. M. (1960). Anatomy and physiology of speech. New York: McGraw Hill.

Kindermann R. & Snell J. L. (1980).Markov Random Fields and their Applications. Vol 1. Amer. Math. Soc.

Kinzel W. (1985). Learning and pattern recognition in spin glass models. Z. Physik, B60, 205-213.

Kirkpatrick S., Gelatt C. D. Jr. & Vecchi M. P. (1983). Optimisation by simulated annealing. Science, 220, 671-680.

Kirkpatrick S. & Sherrington D. (1978). Phys. Rev. 17, 4384-4403.

Kitajima K., Tanabe M. & Isshiki N. (1975). Pitch perturbations in normal and pathological voice. Studia Phon., 9, 25-32.

Kitajima K. & Gould W. J. (1976). Vocal shimmer in sustained phonations of normal and pathological voices. Annals. Otol. 85, 377-381.

Knuth D. E. (1969). The Art of Computer Programming. Vol. 2. Seminumerical Algorithms. Addison Wesley.

Kohonen T. (1977). Associative memory: A system theoretic approach. New York: Springer.

Kohonen T. (1982). Self-organised formation of topologically correct feature maps. Biological Cybernetics, Vol. 43, 59-69.

Kohonen T. (1984). Self-organisation and associative memory. New York: Springer-Verlag.

Kohonen T. (1987). Adaptive, associative, and self-organising functions in neural computing. Applied Optics, Vol. 26, pp4910-4918.

Koike Y., Takahashi H. & Calcaterra T. C. (1977). Acoustic measures for detecting laryngeal pathology. Acta. Otolar., 84, 105-117.

Koike Y. (1967). Application of some acoustic measures for the evaluation of laryngeal dysfunction. JASA 42:1209.

Koike Y. (1969). Vowel Amplitude Modulations in Patients with Laryngeal Diseases. The Journal of the Acoustical Society of America, 45, 839-844.

Koike Y. (1973). Application of some acoustic measures for the evaluation of laryngeal dysfunction. Studia Phonetica, 7, 17-23.

Kuffler S. W., Nicholls J. G. & Martin A. R. (1984). From neuron to brain: cellular approach to the function of the nervous system. 2nd ed. Sunderland, Mass.: Sinauer.

Kullback S. (1959). Information theory and statistics. New York: Wiley.

Lang K. J., Waibel A. H. & Hinton G. E. (1990). A Time-Delay Neural Network Architecture for Isolated Word Recognition. Neural Networks, Vol. 3, 23-43.

Lashley K. S. (1924). Studies of cerebral function in learning: V. The retention of motor habits after destruction of the so-called motor area in primates. Arch. Neurol. Psychiat., 12:249-276.

Lashley K. S. (1950). In search of the engram. Society of Experimental Biology Symposium No. 4: Psychological mechanisms of animal behaviour. London: Cambridge University Press, 478-505.

Laver J., Hiller S., Mackenzie J. & Rooney E. (1986). An acoustic screening system for the detection of laryngeal pathology. Journal of Phonetics, 14, 517-524.

Laver J., Hiller S. & Hanson R. J. (1982). Comparative performances of pitch detection algorithms on dysphonic voices. Proceedings of IEEE Conference on Acoust. Speech and Signal Proc., 192-195.

Laver J., Mackenzie J., Hiller S. & Rooney E. (1985). Acoustic Screening for Vocal Pathology. Proceedings of the Voice Foundation, Denver.

Laver J. (1980). The phonetic description of voice quality. Cambridge University Press.

Le Cun Y., Denker J. S. & Solla S. A. (1990). Optimal Brain Damage. Advances in Neural Information Processing Systems 2. Ed. D. S. Touretzky, Morgan Kaufmann Publishers, San Mateo, California.

Liebermann P. (1961). Perturbations in vocal pitch. JASA, 33, 597-603.

Liebermann P. (1963). Some acoustic measures of the fundamental periodicity of normal and pathological larynges. JASA, 35, 344-353.

Little W. A., & Shaw G. L. (1978). Analytic study of the memory storage capacity of a neural network. Math. Biosci. 19, 101-120.

Little W. A. (1974). The existence of persistent states in the brain. Math. Biosci., 19, 101-120.

Luchsinger R. & Arnold G. E. (1965). Voice-Speech-Language. Clinical Communicology: Its Physiology and Pathology, Wadsworth, Belmont.

Mackenzie J., Laver J. & Hiller S. (1983). Structural pathologies of the vocal folds and phonation. Edinburgh University Dept. of Linguistics, Work in Progress 16, 80-116.

Maw A. R., Cullen R. J. & Bradfield J. W. B. (1982) Verrucous carcinoma of the larynx. Clinical Otolar., 7, 305-311.

McClellan J. (1975). FIR Design Program. In Rabiner L. R. & Gold B. (Eds.) Theory and Application of Digital Signal Processing. New Jersey: Prentice-Hall, 194-204.

McCulloch W. S. & Pitts W. (1943). A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 5, 115-133.

Metropolis N., Rosenbluth A. W., Rosenbluth M. N., Teller A. H. & Teller E. (1953). Equations of state calculations for fast computing machines. Journal of Chemical Physics, 6, 1087.

Michaels L. (1976). Histopathology of nose and throat. In R. Hinchcliffe & D. Hamson (Eds.), Scientific Foundations of otolaryngology, 667-700. London: William Heinemann Medical Books Ltd.

Michel J. (1964). Vocal Fry and Harshness. Phd dissertation, University of Florida.

Minsky M. & Papert S. (1969). Perceptrons. Cambridge, MA: MIT Press.

Moore P. & Leden H. Von (1958). Dynamic variations of the vibratory pattern in the normal larynx. Folio Phoniatricia 10:205-238.

Moore P. (1962). Observations on the physiology of hoarseness. Proceedings of the 4th International Congress of Phonetic Sciences, Helsinki, 92-95.

Moussouris J. (1974). Gibbs and Markov random systems with constraints. Journal of Statistical Physics, 10, 11-33.

Newell G. F. & Montroll E. W. (1953). On the Theory of the Ising Model of Ferromagnetism. Rev. of Mod. Phys., 25, 353-389.

Olsen G. H. (1982). Modern Electronics made simple. London: Heinemann.

Peeling S. M. & Moore R. K. (1987). Experiments in Isolated Digit Recognition using the Multi-Layer Perceptron. Royal Signals & Radar Establishment Memorandum No. 4073, Malvern, Worcs.

Peretto P. (1984). Collective Properties of Neural Networks: A Statistical Physics Approach. Bio Cybern. 50, 51-62.

Perkel D. H. & Bullock T. H. (1969). Neurosci. Res. Symp. Summ. 3, 405-527.

Perkins H. (1977). Speech Pathology, An Applied Behavioural Science. St. Louis: The C. V. Mosby Co.

Pineda J. (1987). Generalization of backpropagation to recurrent neural networks. Proc. of IEEE Conf. on neural Inf. Processing Sys.- Natural and synthetic, Boulder, Colorado.

Poincaré H. (1913). Foundations of Science (G.B. Halstead, Trans.). New York: Science Press.

Prager R. W., Harrison T. D. & Fallside F. (1986). Boltzmann machines for speech recognition. Computer Speech and Language 1, 3-27.

Rabiner L. R., Sambur M. R. & Schmidt C. E. (1975). Applications of nonlinear smoothing algorithm to speech processing. IEEE Trans. ASSP-23, 552-557.

Rabiner L. R. & Schafer R. W. (1978). Digital Processing of Speech Signals. New Jersey: Prentice Hall.

Ramig L. A. & Ringel R. L. (1983). Effects of physiological aging on selected acoustic characteristics of voice. JSHR, 26, 22-30.

Renyi A. (1962). Probability Theory. Amsterdam: North-Holland, 1962.

Romanes G. J. (Ed.)(1978) Cunningham's Manual of Practical Anatomy, Vol 3, Head and Neck and Brain (14th edn.). Oxford University Press.

Rosenblatt F. (1959). Two theorems of statistical separability in the Perceptron. In Mechanisation of Thought Processes: Proceedings of a symposium held at the National Physical Laboratory, November 1958. Vol. 1, 421-456. London: HM Stationary Office.

Rosenblatt F. (1962). Principles of neurodynamics. New York: Spartan.

Rumelhart D. E., Hinton G. E. and McClelland J. L. (1986). A General Framework for Parallel Distributed Processing. In Rumelhart D. E. & McClelland J. L. (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations. MIT press (1986).

Rumelhart D. E., Hinton G. E. and Williams R. J. (1986). Learning Internal Representations by Error Propagation. In Rumelhart D. E. & McClelland J. L. (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations. MIT press (1986).

Saunders W. H. (1964). The Larynx. New Jersey: CIBA Corp.

Sejnowski T. & Rosenberg C. R. (1986). NETtalk: A parallel Network that Learns to Read Aloud. John Hopkins Univ. Technical Report JHU/EECS-86/01.

Shaw H. (1979). Tumours of the larynx. In J. Ballantyne & J. Groves (Eds.), Scott-Brown's Diseases of the Ear, Nose and Throat. (4th edn), Vol. 4, 421-508.

Smith D. R., & Davidson, C. H. (1962). J. Assoc. Comput. Mach., 9, 268-279.

Smith W. R. & Liebermann P. (1964). Studies in Pathologic Speech Production, Final Report, AFCRL-64-379, Air Force Cambridge Research Laboratories, L. G. Hanscom Field, MA.

Smith W. R. & Liebermann P. (1969). Computer diagnosis of laryngeal lesion. Computers and Biomedical Research 2:291-303

Smolensky P., & Riley M. S. (1984). Harmony theory: Problem solving, parallel cognitive models, and thermal physics. (Tech. Rep. No. 8404). La Jolla: University of California, San Diego, Institute for Cognitive Science.

Smolensky P. (1984). The mathematical role of self-consistency in parallel computation. Proceedings of the Sixth Annual Conference on Artificial Intelligence AAAI-83, 109-113.

Smolensky P. (1986). Information Processing in Dynamical Systems: Foundations of Harmony Theory. In Parallel Distributed Processing: Explorations in the Microstructures of Cognition. Volume 1: Foundations. Rumelhart D. E. & McClelland J. L. (Eds.). MIT Press (1986).

Smolensky P. (1986a). Neural and Conceptual Interpretation of PDP Models. In Parallel Distributed Processing: Explorations in the Microstructures of Cognition. Volume 2: Physiological and Biological Models. Rumelhart D. E. & McClelland J. L. (Eds.). MIT Press (1986).

Szu H. (1986). Fast Simulated Annealing. In Denker J. S. (Ed.) AIP Conference Proceedings 151, Neural Networks for Computing, Snowbird, Utah.

Thompson R. S., & Gibson W. G. (1981). Neural Model with probabilistic firing behaviour. I. General considerations. Math. Biosci. 56,239-253.

Thompson R. S., & Gibson W. G. (1981). Neural Model with probabilistic firing behaviour. II. One- and two-neuron networks. Math. Biosci. 56,255-285.

Titze I. R. (1973). The human vocal cords: a mathematical model, part 1. Phonetica, 28,129-170.

Toulouse G., Dehane S., & Changeux J-P. (1986). Spin glass model of learning by selection. Proceedings of the National Academy of Sciences of the United States of America, 83, 1695-1698.

Trehern J. F., Jack M. A., Laver J. & Hiller S. M. (1987). Acoustic Screening for Vocal Pathology with a Boltzmann Machine. Proc. European Conference on Speech Technology, Vol. 1. Edinburgh.

Trehern J. F., Jack M. A. & Laver J. (1986). Speech Processing with a Boltzmann Machine. Proc. ICASSP-86. Vol. 1., 721-724. Tokyo.

Von Leden H., Moore P. & Timcke R. (1960). Laryngeal Vibrations: Measurements of the Glottic Wave. Part III. The Pathologic Larynx. Archives of Otolaryngology, 71,16-35.

Von Neumann J. (1961). Taub A. H. (Ed.) Collected Works Vol. 5. New York: Pergamon Press.

Wallace D. J. (1985). Memory and Learning in a Class of Neural Models. In Bunk B. & Mutter K. H. (Eds.) Proceedings of the Workshop on Lattice Gauge Theory, Wuppertal, 1986, Plenum.

Wendahl R. W. (1963). Laryngeal Analog Synthesis of Harsh Voice Quality. Folia Phoneatrica, 15, 241-250.

Wendahl R. W. (1966). Laryngeal Analog Synthesis of Jitter and Shimmer. Auditory Parameter of Harshness. Folia Phoneatrica, 18,98-108.

Widrow B. & Hoff M. E. (1960). Adaptive switching circuits. Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, Part 4, 96-104.

Widrow B. & Stearns S. D. (1985). Adaptive Signal Processing, Prentice Hall.

Wilcox K. A. & Horii Y. (1980). Age and changes in vocal jitter. J. of Gerontology, 35, 194-198.

APPENDICES

APPENDIX 1

Speech Processing with a Boltzmann Machine

Trehern, Jack & Laver, 1986.

Proceedings of ICASSP-86 Vol. 1, 721-724. Tokyo.

SPEECH PROCESSING WITH A BOLTZMANN MACHINE

J. F. Trehern, M. A. Jack and J. Laver

Centre for Speech Technology Research
University of Edinburgh
80 South Bridge
Edinburgh

ABSTRACT

Recent work on the representation of 'knowledge' in massively parallel networks has resulted in the development of the Boltzmann Machine. In such networks, 'knowledge' is represented by the pattern and strengths of interconnections between simple processing elements. In the Boltzmann Machine noise is used to aid the search procedure in the network.

Developed primarily for static image processing, Boltzmann Machines are well suited to problems that can be formulated as constraint satisfaction searches. This paper describes some simple experiments using a Boltzmann Machine to process and classify static speech patterns.

INTRODUCTION

Advances in VLSI technology making possible the implementation of hundreds or thousands of cooperative processing elements, and increased knowledge about the structure of the brain has lead to a resurgence of interest in 'connectionist' models. These models enable a significant part of the knowledge of a system to be applied to a particular problem in a very short time. This is particularly attractive for speech recognition, where the need to segment and label the non-symbolic speech signal, using knowledge sources is expedient.

Generally, connectionist machines are unprogrammed, with the networks finding their solutions by settling into stable states rather than following detailed algorithms. The Boltzmann Machine [3], is a particular type of massively parallel network that is well suited to 'weak' constraint satisfaction searches. In finding the most plausible interpretations for perceptual data, the best solution may well violate some of the constraints of the problem domain. A more expedient approach is to use weak constraints that incur a cost when they are violated. The quality of the interpretation is then determined by the total cost of the infringed constraints.

In the Boltzmann Machine the constraints of the problem domain are encoded in the connections between simple processing elements of the network, with the magnitude of the connection strengths representing the cost of violating that constraint. The search for good global states (i.e. low cost interpretations) is performed by repeatedly allowing each processing element to adopt whichever of its permitted two states minimises the total cost for the current state of the network.

This procedure may result in local minima that are not globally optimal, but by making the decision rule probabilistic, it is possible to escape from these local minima. The probability of the system being in a particular global state, and the energy of that state, are related by the Boltzmann distribution. This has made possible the development of a simple learning rule for modifying the connection strengths of the network so as to embody knowledge about the problem domain.

We present here preliminary results of experiments using a very simple Boltzmann Machine to process and classify static speech patterns.

THE BOLTZMANN MACHINE

The Boltzmann Machine consists of elementary processing elements called units. These units can be in one of two states, *on* or *off*. Which state is adopted is a probabilistic function of the states of its contiguous units, and the weights on the links to them. Significantly, link weights are symmetric, having the same strength in both directions. Each unit also has a bias (or threshold value) associated with it.

It has been shown [1], that each global state of a network of symmetrically connected binary processors can be assigned an 'energy' value. If some of the units of the network are coerced into particular states, the system will then find the minimum energy configuration that is compatible with that input.

The energy of a global state is defined as:

$$E = - \sum_{i<j} w_{ij} s_i s_j \qquad (1)$$

where w_{ij} is the strength of the connection between units i and j, s_i is 1 if the unit is *on* , and s_i is 0 if the unit is *off*. (We assume here that the threshold value associated with the unit is accounted for in the strength of a link between i and a unit that is always in the *on* state).

Because the connections are symmetric, the difference in global energy for the *on* and *off* states of the k^{th} unit can be determined locally by that unit. unit. The energy difference is:

$$\Delta E_k = \sum_i w_{ki} s_i \qquad (2)$$

The search for good (low energy) global states is done by selecting a unit at random and setting its state to that which provides the least energy contribution, given the current states of the other units.

However, this simple search algorithm suffers from becoming stuck in local minima that are not globally optimal. One way around this problem is to allow occasional uphill steps to configurations of higher energy, thus allowing the system to escape from local minima.

The Metropolis algorithm [6] has this desired property, and has recently been adapted [2],[5] to constraint satisfaction problems. The probabilistic decision rule is as follows: If the energy gap of the k^{th} unit is ΔE_k, then regardless of the previous state, set $s_k=1$ with probability

$$p_k = \frac{1}{1 + e^{-\Delta E_k / T}} \qquad (3)$$

where T is a parameter that behaves like temperature. (See Fig. 1). A system following this decision rule obeys a Boltzmann distribution. At low temperatures, the time required to make the jump out of a local energy minima may be long, where as at high temperatures these barriers are easily jumped, but this results in only a coarse level search.

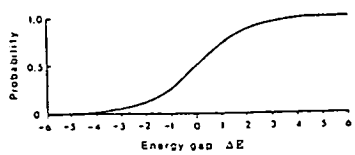

Figure 1

Probability for a unit to be
on
as a function of energy gap ΔE for T=1

To reliably find good minima in a limited time, the system is started at a high temperature, and then the temperature is gradually lowered. This technique is known as optimisation by simulated annealing.

The properties of the Boltzmann distribution allow analysis of the stochastic search process, and has resulted in the development of a learning algorithm. This allows the strengths of connections to be adapted so that the network can capture the essential properties of its environment.

To minimise the distance between the probability distributions for when the system is clamped by the environment , and for when it is free-running, statistics about how often pairs of units are *on* together are collected when the system is at thermal equilibrium. Each weight is then changed by an amount proportional to the difference between these probabilities. If p_{ij} is the average probability of two connected units both being in the *on* state when the environment is clamping, and p'_{ij} is the corresponding probability when the network is running freely, (both probabilities measured at thermal equilibrium), then each weight is changed by an amount:

$$\Delta w_{ij} = \varepsilon(p_{ij} - p'_{ij}) \qquad (4)$$

where ε scales the size of each weight change.

When this distance has been minimised, the network will have generated a set of weights that make up a locally optimal model of the constraints in the problem domain.

SPEECH PROCESSING

In order to explore the Boltzmann Machine concept for speech recognition tasks, a small, elementary network has been considered in addressing the problem of reliable recognition of isolated words for one speaker, having trained the system on one utterance of each word. Of particular interest is the capability of the Boltzmann Machine in making the generalisations required to recognise previously unseen utterances of the trained words.

The speech data used comprised the English digits (0-4) for a speaker from the South of England. The speech was recorded under studio conditions, bandlimited to 4kHz, and digitised at 10kHz. Features were then extracted from 32mS blocks of the digitised speech to yield a set of eight equally spaced (normalised log amplitude) spectral coefficients.

The Boltzmann Machine structure used consisted of a two-dimensional array of units coupled to five output units (i.e. one for each of the spoken digits). The two-dimensional layer comprised 8×16 units, with each of the units being connected to its twenty-four nearest neighbours. Each unit of this planar array was connected to each of the output units.

The processed speech data was applied directly to the units of the two-dimensional array such that the data, in effect, acts as a variable weight vector between the units of the array and a set of permanently *on* units. Alternatively, the speech data could have been quantised to sixteen discrete levels, and a unit assigned to each of these levels in the network [4]. This option was rejected in the present work, as the number of units would have increased significantly beyond available computational capabilities.

The learning procedure for the network was as follows. For each learning cycle, two speech patterns were selected at random, and applied in turn to the input layer. The speech data was noisily clamped by randomly varying the sample values over the range ±3% of the maximum possible value of the speech data. The output units were noisily clamped as follows, each *on* unit of the output clamp vector was switched *off* with a probability of 0.2 , and each *off* unit of the output vector was switched *on* with a probability of 0.15 .

At the start of the learning phase, all the bias settings and weights were set to zero, except for the bias value of the layer units, which was set to -2. For each of the speech patterns applied, the network was allowed to reach equilibrium once.
Co-occurrence statistics were then gathered for ten units of time. (One unit of time being defined as the time required for each unit to be given on average one chance to change its state). During the unclamped phase of learning the output units of the network were randomised with an equal probability of being *on* or *off*. The network was then allowed to reach equilibrium again, and co-occurrence statistics again gathered for ten units of time.

Instead of adapting the weights by an amount proportional to $p_{ij} - p'_{ij}$, they were incremented by a fixed step of 1.0 if $p_{ij} > p'_{ij}$, and decremented by a fixed step of 1.0 if $p_{ij} < p'_{ij}$.

Each time the network was taken to equilibrium the units were initially randomised with equal probability of being *on* or *off*. The annealing schedule followed, allowed the network to run for the following times at the following temperatures:- (2 units of time with T=10; 2 units with T=7; 2units with T=3; 4 units with T=1), after which it was assumed that the network had reached equilibrium.

RESULTS

After typically 1000 learning cycles the network was able to provide 100% discrimination (as measured at the output units) of the five utterances used for training. On being presented with a subsequent utterance of the trained words by the same speaker the network proved to be capable of making the correct decision for only 64% of the time.

By observing the patterns of the states of the units in the input layer of the network, it is possible to pick out some of the features of the input data. Figure 2 and figure 3 depicts the states of the units for the utterance pattern. The similarity between these two figures forms the basis for continuing investigation of speaker-independent operation of the Boltzmann Machine.

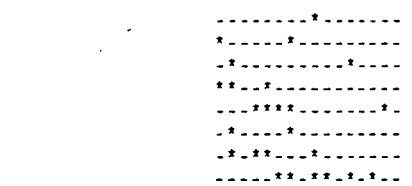

Figure 2

Unit states for trained word 'two'.

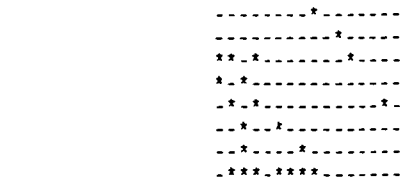

Figure 3

Unit states for word 'two'.

CONCLUSIONS

A very simple Boltzmann Machine network has been considered. The structure has been demonstrated as partially successful in identifying previously unseen speech patterns. The model currently incorporates only a simple type of learning as none of the units are hidden from the environment. The effects of varying the connectivity patterns of the model, and the inclusion of extra layers of units in the network remain to be explored.

There are many parameters that need closer investigation. Parameters observed to have a profound affect on the learning ability of the network are those of the weight step value used in adapting the link weight values, and the annealing schedule used. Current work includes investigation of the effects of parameter variations, and carrying out more detailed simulations using an array processor implementation.

REFERENCES

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," Proc. Natl. Acad. Sci. USA, Vol 79, pp. 2554-2558,April, 1982.

[2] S. Kirkpatrick, C. D. Gelatt, M. P Vecchi, "Optimisation by Simulated Annealing," Science, Vol. 220, pp. 671-680, May, 1983.

[3] G. E. Hinton, T. J. Sejnowski, D. H. Ackley, "Boltzmann Machines: Constraint Satisfaction Networks that Learn," Technical report CMU-CS-84-119, Carnegie-Mellon University, May, 1984.

[4] J. S. Bridle, R. K. Moore, "Boltzmann Machines for Speech Pattern Processing," Proc. I.O.A., Vol. 6 Part 4, pp. 315-322, 1984.

[5] S. Geman, D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No. 6, pp. 721-741, November, 1984.

[6] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, E. Teller, Journal of Chemical Physics, 6, 1087, 1953.

APPENDIX 2

Acoustic Screening for Vocal Pathology with a Boltzmann Machine

Trehern, Jack, Laver & Hiller, 1987.

Proceedings of European Conference on Speech Technology, Vol. 1, Edinburgh.

ACOUSTIC SCREENING FOR VOCAL PATHOLOGY WITH A BOLTZMANN MACHINE

J. F. Trehern*, M. A. Jack*, J. Laver*, S. M. Hiller*.

ABSTRACT

The measurement of pitch perturbation has been used in acoustic screening for the detection of vocal disorders. As an alternative to applying conventional statistical techniques to the pitch perturbation parameters used for separation of control and pathological speakers, a parallel distributed processing model approach has been implemented using a Boltzmann Machine (BM). After training, the machine is able to separate the two groups and to classify the vocal disorders present in the pathological group of speakers. The connections developed by the Boltzmann Machine have also given some insights into the usefulness of each of the perturbation parameters employed.

INTRODUCTION

It has been suggested that the measurement of pitch perturbation may be of value in screening for vocal disorders (ref 1), and work in this field has proved this approach to be partially successful in separating pathological speakers from control speakers (ref 2). The basic methodology used to achieve this group separation has been to extract parameters from a sample of continuous speech for each speaker, and then subject these parameters to a statistical analysis using bi-variate plots or a multivariate technique (linear discriminant analysis).

In this paper, we consider the application of a new technique based on statistical mechanics, to separate the control speakers from the pathological speakers, and to differentiate between the various vocal disorders present. The Boltzmann Machine (ref 3), is a recent development in parallel distributed processing (PDP), and consists of a nonlinear network of stochastic binary processing units, which interact pairwise through symmetric connection strengths. An important feature of the BM is the existence of a domain independent learning algorithm. This allows the strengths of connections between processing units to be modified and enables the network to create internal representations that capture underlying features in the training data.

ACOUSTIC ANALYSIS SYSTEM

The analysis system produces a set of 10 parameters for each speaker, taken from measurements of fundamental frequency (F_0) and waveform perturbations in approximately 40 seconds of recorded text read from the 'Rainbow Passage' (ref 4). The measurement system uses a modified version of the Gold and Rabiner parallel processing pitch detection algorithm, with phase compensation for low frequency distortion introduced by tape recording techniques; low pass filtering to remove high-frequency resonance effects from the waveform (600 Hz males); non-linear smoothing to derive an intonational 'trendline' from the new pitch period estimates; parabolic interpolation at waveform peaks to provide greater resolution of pitch period values (ref 5).

Intonational data are derived from a smoothed F_0 trendline, giving its mean value (F_0-AV) and its range, represented as the standard deviation of trendline values (F_0-DEV). Statistical analyses are then made of pitch period perturbation (jitter) and amplitude perturbation at waveform peaks (shimmer). The following measures are taken for both jitter and shimmer:-

(1) Average magnitude of excursion of the raw F_0 contour from the local trendline (AVEX).

* Centre for Speech Technology Research, Univ. of Edinburgh, 80 South Bridge, Edinburgh, EH1 1HN.

(2) Standard deviation of (signed) excursion from trendline (DEVEX).

(3) Rate of excursion (RATEX); percentage of points in the sample where magnitude of excursion is greater than or equal to 3% of local trendline value. 3% was chosen as even the healthiest of voices, performing monotone steady-state vowels typically show a level of (jitter) perturbation of about 2% (ref 6).

(4) Directional perturbation factor (DPF). This measure, adapted from Hecker and Kreul (ref 7), is the percentage of changes in algebraic sign between adjacent pitch or amplitude estimates in the raw contour. A 3% threshold was also applied.

BOLTZMANN MACHINE

The Boltzmann Machine is a particular type of massively parallel network. Processing in the network is performed by elementary binary units, which are either in the *on* or *off* state. There are two types of unit in the network, visible and hidden. The visible units are those that interface the network to its environment, and the hidden units are those that have no environmental contact.

Units are connected to each other by bi-directional links, which have real value weights attached to them. The state of each unit is a stochastic function of the states of its contiguous units, and the strengths of its connections to them. Processing proceeds with each unit asynchronously updating its state until the network eventually reaches equilibrium. The strength of the connections between units is used to store 'knowledge', which is used to provide an interpretation of any input applied to the network.

A cost function called 'energy' has been defined by Hopfield (ref 8), for binary symmetric networks. Searching such a network may thus be regarded as minimising this cost function. One strategy for minimising the energy is for each unit to update its state by switching itself into whichever state minimises its contribution to the global energy value for the network. However, this can result in the network becoming stuck in local minima that are not globally optimal. To avoid this happening occasional uphill jumps in energy value are allowed so that these local minima may be escaped from. The decision rule for the units is now as follows: if the energy gap between the *on* and *off* states of the k^{th} unit is ΔE_k then regardless of the previous state set state of $s_k = 1$ with probability

$$p_k = \frac{1}{1 + e^{\frac{-\Delta E_k}{T}}} \tag{1}$$

T is a parameter that acts like temperature. (When $T = 0$ function p_k is a step function, i.e. the search will follow a normal gradient descent).

The temperature T determines the size of uphill jumps in energy allowed. A reliable method of taking a network to thermal equilibrium in a given time is achieved by at first allowing large jumps in energy (high temperature), resulting in a coarse search of the energy landscape. Then gradually reducing the temperature to a low value, better minima may be found within the coarse scale minima. This process is called simulated annealing.

For a system following the above decision rule, the relative probability between two global states of the network will follow a Boltzmann distribution. This has allowed the development of a domain independent learning algorithm (ref 3) that is able to modify the strength of connections by 'experience', so that a network is able to develop an internal model of its training environment.

APPLICATION TO ACOUSTIC SCREENING FOR VOCAL PATHOLOGY

To assess the BM for vocal pathology screening, experiments were performed on totally connected networks having a 10×8 input array, 2 output units for the control/pathology discrimination, and 5 output units for pathology classification. During the training procedure each input training pattern in turn was clamped over the input units, and the desired output

pattern clamped over the output units. The machine was then taken to equilibrium and statistics were gathered about how often pairs of units were on together. This was then repeated, but with the output units unclamped. Co-occurrence statistics were again gathered. The clamped/unclamped statistics were used in the adaption of the connection strengths. It should be noted that hidden units are never clamped by the environment.

The data employed consisted of a pathological group of 37 male speakers, whose laryngeal state had been established by medical examination, and a control group of 39 male speakers, who although not subjected to a laryngeal examination reported no history of laryngeal disorders, or other relevant complaints. The laryngeal disorders present in the pathological group were classified as shown in Table 1.

Table 1. Classification of laryngeal disorders and number of cases.

PATHOLOGY	#
Disorder of ligamental area:	
Squamous cell carcinoma	6
Sessile vocal polyps	6
Not classified	20
Disorders of cartilaginous area:	5

The maximum global range for each of the 10 perturbation parameters employed was determined and divided into octiles. The parameters were presented in binary form as a direct mapping of the occupancy state of the octiles, to the 10×8 input array of the BM.

The training data employed consisted of 20 control speakers and 20 pathological speakers. (The pathological training data set comprised the following: 3 carcinomas; 3 polyps; 3 cartilaginous; and 11 not-classified). For the problem of control/pathological group separation a machine with 2 output units corresponding to each group was used. Machines with (and without) hidden units proved to be capable of providing 100% discrimination between the groups for the training data set. Table 2. summarises the results for 'unseen' test data. Although the hidden units have not had any effect on the control recognition rate, the pathological recognition rate has been improved.

Table 2. Control/pathological group separation results for test data.

GROUP	TEST DATA % ERROR	
	no hidden	with hidden (8)
Controls	10.5	10.5
Pathologicals	35.3	17.6

For the problem of pathological classification a BM with 5 output units was used, with each unit representing one of the classes. Again, the machine was capable of 100% discrimination between classes in the training data. Table 3. summarises the results for 'unseen' test data.

Table 3. Control/pathological classification results for test data.

GROUP	TEST DATA % ERROR	
	no hidden	with hidden (4)
Controls	5.3	0.0
ALL Pathologicals	88.2	76.5

These results show an improvement in both the control and pathological class separation results with a machine using hidden units. The recognition rates for the individual pathology

classes are not shown because in most cases the machine was unable to identify data outside of its training set. This is probably due to the very small amount of data available - 3 patterns for each class except for the not-classified and control groups. In fact the not-classified group was able to record a percentage error of 66.7% with 'unseen' test data.

Further preliminary experiments showed that the 2 output unit BM was capable of distinguishing between the control and pathological groups correctly when trained on the complete corpus of data. This compares very favourably with the results previously obtained by linear discriminant functions (ref 2) viz. 14.5% incorrect classifications for the pathological group and 7.9% incorrect classifications for the control group.

For the two group separation problem it was observed that particularly strong connections are made to the F_0-AV, J-DEVEX, J-AVEX, S-RATEX, J-DPF and S-DPF parameters. This supports previous findings indicating the importance of functions of perturbation to this type of classification (ref 2).

CONCLUSIONS

The separation of the two groups of subjects and the classification of different pathologies has been shown using a BM. These results compare favourably with those obtained by linear discriminant analysis.

There appears to be some underlying structure to the data as determined by the small improvement gained by using hidden units in the BM. The power of the BM lies in its ability to capture any underlying structure using hidden units. Due to the fact that only limited data samples were available for some of the groups, increased training data is expected to show better separation performance with hidden units.

This work was supported in part under the Anglo-Portuguese Joint Research Programme -Treaty of Windsor.

REFERENCES

[1] S. B. Davis, Acoustic Characteristics of Normal and Pathological Voices. In N. J. Lass (Ed), Speech and Language: Advance in Basic Research and Practice (Academic Press, NY, 1979) p273-338.

[2] J. Laver, S. Hiller, J. Mackenzie, E. Rooney, An acoustic screening system for the detection of laryngeal pathology, Journal of Phonetics 14, p517-524 (1986).

[3] G. E. Hinton, T. J. Sejnowski, D. H. Ackley, Boltzmann Machines: Constraint Satisfaction Networks that Learn, Tech. Rep. CMU-CS-84-119 (Carnegie-Mellon Univesrsity, May 1984).

[4] G. Fairbanks, Voice and articulation drill book, (New York: Harper Row, 1960).

[5] S. Hiller, J. Laver & J. Mackenzie, Automatic analysis of waveform perturbations in connected speech, Edinburgh University Dept. of Linguistics, Work in Progress 16, p40-68 (1983).

[6] R. Hanson, A two-state model of F_0 control, JASA 64 (1978) p543-544.

[7] M. Hecker, E Kreul, Descriptions of the speech of patients with cancer of the vocal folds. Part 1: measures of fundamental frequency, JASA 49 (1971) p1275-1282.

[8] J. J. Hopfield, Neural Networks and physical systems with emergent collective computational abilities, Proc. Nat. Acad. Sc. USA, 79, (1982), p2554-2558.