Computerised Electrocardiogram Classification

A thesis submitted to the University of Wales, Cardiff
in candidature for the degree of

Doctor of Philosophy

by

Zakria Zaki Mahrousa, B.Eng.

Manufacturing Engineering Centre
School of Engineering
Cardiff University
United Kingdom

2004

In the Name of Allah, The Beneficent, The Merciful

SYNOPSIS

Advances in computing have resulted in many engineering processes being automated. Electrocardiogram (ECG) classification is one such process. The analysis of ECGs can benefit from the wide availability and power of modern computers.

This study presents the usage of computer technology in the field of computerised ECG classification. Computerised electrocardiogram classification can help to reduce healthcare costs by enabling suitably equipped general practitioners to refer to hospital only those people with serious heart problems. Computerised ECG classification can also be very useful in shortening hospital waiting lists and saving life by discovering heart diseases early.

The thesis investigates the automatic classification of ECGs into different disease categories using Artificial Intelligence (AI) techniques. A comparison of the use of different feature sets and AI classifiers is presented. The feature sets include conventional cardiological features, as well as features taken directly from time domain samples of an ECG. The benchmark AI classifiers tested include those based on neural network, k-Nearest Neighbour and inductive learning techniques.

The research proposes two modifications to the learning vector quantisation (LVQ) neural network, namely the All Weights Updating-LVQ (AWU-LVQ) algorithm and the Neighbouring Weights Updating-LVQ (NWU-LVQ) algorithm, yielding an "intelligent" diagnostic heart system with higher accuracy and reduced training time compared to existing AI techniques.

To my parents,

my wife Rawha Lababidi

and my daughters Sara and Sidra,

without their loving support

this would not have happened.

Acknowledgements

First of all I thank Allah (my Lord) the all high, the all great who made it possible for me to complete this work.

I would like to thank my supervisor Professor **D. T. Pham** for his excellent supervision, continuous encouragement and support. He was a brilliant supervisor. He really deserves more thanks than I can properly express for the great deal of time he gave me, and for making that time the most enjoyable and rewarding hours of my academic life.

I would like to thank **Aleppo University** for sponsoring me during my Ph.D. studies.

I reserve my deepest gratitude for my parents, **Mohammed Zaki Mahrousa** and **Fawzia Rehawai** who gave and have given me continuous support and encouragement throughout my studies, and my parents-in-law **M. Amine Lababidi** and **Naila Kouka** for their encouragement. Many thanks also to my brothers, sisters, brothers-in-law, sisters-in-law and a special thanks goes to brother **M. Yehaya Mahrousa** for his brotherhood advice.

My thanks are also due to all those members of the Intelligent System Laboratory who contributed to my work through their valuable help, advice and technical support. In particular, I would like to mention Dr.E. E. Eldukhri.

Thanks go to Dr. R. S. Gault, Dr. B. Peat, and Mr. A. Afify for their proof reading.

Finally, I am indebted to all those colleagues whose friendship and support I valued more than any technical help. Special thanks also go to Dr. Y. Dadam, Dr. Z. Salem, Dr. B. Al Mourad, Mr. S. Otri, Mr. M. R. Assi and all those who made my stay in the laboratory particularly pleasant.

Contents

List of Figures

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 5

List of Tables

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Abbreviations

a	Aberrated Atrial Premature Beat
A	Atrial Premature Beat
AI	Artificial Intelligence
ANN	Artifical Neural Network
ART	Adaptive Resonance Theory
AV	AtrioVentricular
AWU-LVQ	All Weights Updating – LVQ
BIH	Beth Israel Hospital
BP	Back Propagation
E	Ventricular Escape Beat
ECG	Electrocardiogram
EEG	Electro Encephalogram
F	Fusion of paced and normal beat
FS-LVQ	Fuzzy Soft-Learning Vector Quantisation
j	Nodal (junctional) Escape Beat
k-NN	k-Nearest Neighbour
L	Left Bundle Branch Block Beat
LA	Left Arm wrist
LL	Left Leg ankle
LVQ	Learning Vector Quantisation
MIT-BIH	Massachusetts Institute of Technology- Beth Israel Hospital
MLP	Multi-Layer Perceptron

N	Normal Beat
NN	Neural network
NWU-LVQ	Neighbouring Weights Updating - LVQ
P	Paced Beat
R	Right Bundle Branch Block Beat
RA	Right Arm wrist
RBF	Radial Basis Function
RL	Right Leg ankle
SA	Sino Atrial
SLPs	Single-layered Perceptrons
SOFM	Self Organising Feature Map
SOM	Self Organising Map
SVMs	Support Vector Machines
V	Premature Ventricular Contraction
ZISC	Zero Instruction Set Computer

NOMENCLATURE

Chapter 2

F_s	Scaled value after normalisation
F_{min}	Minimum value in the data set
F_{max}	Maximum value in the data set
F_u	Unscaled value before normalisation
λ	Mean of the data set
ϕ	Standard deviation of the data set

Chapter 3

X_u	Unlabeled sample
$x^{(i)}$	Training pattern vector
n_l	Number of training pattern from class l
$K_l(k,n)$	Number of patterns from class w_l among the k-NN of pattern x
Class (x)	k-NN decision rule
$D(x,x^{(i)})$	Euclidean distance between two pattern vectors x and $x^{(i)}$
d	Number of features
$M(x,x^{(i)})$	Matching score for the test pattern x and the training pattern x^i
y_j	Output of the neuron j
$f(.)$	Activation function
Δw_{ji}	Weights adjustment between neurons i and j in the forward direction

η	Learning rate
δ_j	Error factor of neuron *j*
net_j	Total of all the inputs to neuron *j*
d_j	Desired output of neuron *j*
o_{jp}	Gaussian function
σ_j^2	Variance of the Gaussian function
x_p	Input vector for index pattern *p*
τ	Momentom convergence
k	Output unit index
K	Total number of output units
y_{kp}	Output of the RBF network
P	Total number of patterns
P_{kp}	Degree to which pattern *p* belongs to class *k*
t	Time step
t_p	Target for pattern *p*
w_{ki}	Weight from the i th hidden unit to the k th output
Δw_{ki}	Weight updates for the winning output unit y_{kp}
v_{A_n}	Discrete attribute *A* with *n* possible values
C_i	The i th class of a labelled data set *S* with *k* classes
$P(C_i,S)$	proportion of instances in *S* which are in class C_i
S	labelled data set
Ent(S)	Entropy of a labelled data set *S*

Chapter 4

d_i	Euclidean distance between the input vector x and the weight vector w_i
d_k	Euclidean distance between the next closest neuron of the winning neuron and the input vector
w_{ij}	j^{th} components of the weight
x_j	j^{th} components of the input vector x
x	Input vector
w	Weight vector
α	Learning rate
h	Neighbourhood function
σ	Width of the kernel of the Gaussian function
β	A parameter used to eliminate negative corrections in supervised learning

Chapter 5

x	Input vector
w	Weight vector
α	Learning rate
h	Neighbourhood function of the winner j
r_j, r_k	Vectorial locations for the display grid
σ	Width of the kernel of the Gaussian function

$\alpha\,(0)$	Starting value of the learning rate
α_0	Rate of decrease
μ	Fuzzy C Mean (FCM) membership functions
$f(t)$	Function of t that controls the approximated excitation of the FS-LVQ neurons
h	Neighbourhood function
β	A parameter used to eliminate negative corrections in supervised learning

Chapter 1

Introduction

1.1 Computerised Electrocardiogram Classification

Since the advent of computerised ECG classification systems in 1957 [Titomir, 2000], research in the field has proliferated. Each year, increasing numbers of ECGs are recorded using ECG recording systems of on-board ECG diagnosis facilities, for example to detect significant changes in the patients' physiological state. The aim of ECG classification is to determine if the patient is ill and requires treatment or if the patient has no cardiac abnormalities and requires no treatment. In general, computerised ECG systems can assist the non-specialist with patient diagnosis, offering greater consistency through avoidance of observer variability. A computerised system normally consists of three processing modules: beat detection, feature extraction and classification, as shown in Figure 1.1. Beat detection concerns the identification and location of each cardiac cycle. The reference markers generated by the beat detection module are processed by the feature extraction module to produce a feature vector describing the morphology of the recorded ECG. The feature vector is then examined by the classification module, giving a hypothesis based on the available evidence.

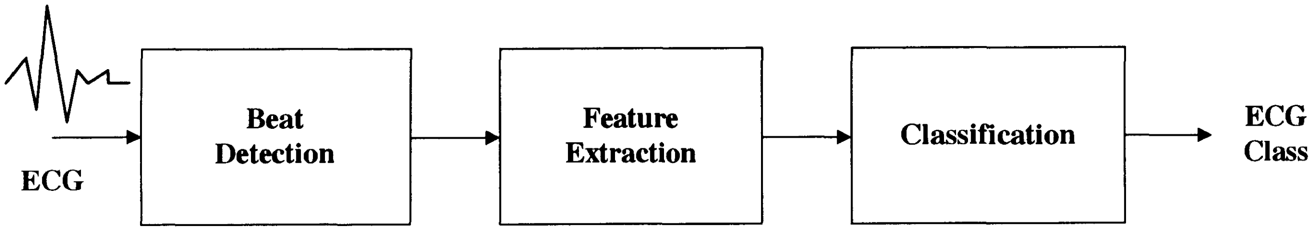


Figure 1.1: Three processing modules of the computerised ECG system

During the last two decades, research into beat detection and feature extraction has to some extent reached a plateau with regard to achieved performance levels. In order to enhance the overall performance of the system so that it could be used reliably, the classification stage is therefore now being targeted.

1.2 Research Topic

This research concerns the automatic classification of ECGs for arrhythmia analysis. A comparison was carried out of different classification techniques drawn mainly from the field of Artificial Intelligence (AI). AI classification techniques were considered as they can learn the classification task automatically. They do not require a *priori* modelling and are simpler to apply, yet promise to yield good performance.

The work focused on developing improved AI classification tools for distinguishing between abnormal ECGs (indicating the presence of arrhythmias) and normal ECGs (showing no signs of arrhythmias).

The AI techniques investigated in this research were neural networks and inductive learning. Neural networks were focused upon due to their superior abilities in handling continuous inputs such as those derived from an ECG. Different neural networks were studied and improvements were made to the Learning Vector Quantisation (LVQ) network, a well-known classification tool.

1.3 Objectives

The main objectives of this research were:

- To compare the performance of AI techniques such as neural networks and inductive learning for computerised ECG classification. The k-Nearest Neighbour method was also considered in the comparison.

- To compare the classification performance using different features extracted from ECG signals and different signal sampling rates. The aim was to determine the features and sampling rates that characterise ECG signals well and can be used successfully for classification purposes.

- To develop improved Learning Vector Quantisation (LVQ) algorithms in order to achieve better classification accuracy with a short training time. Two modifications were made to the standard LVQ algorithm to yield new LVQ algorithms. The first modification to the LVQ algorithm is called the All Weights Updating LVQ (AWU-LVQ). This gave better classification accuracy with a short training time compared to the standard LVQ algorithm. The second modification is called the Neighbouring Weights Updating LVQ (NWU-LVQ). This has a higher classification accuracy and required a shorter training time compared to other LVQ Algorithms.

1.4 Outline of the Thesis

This thesis comprises six chapters and two appendix.

Chapter 2 reviews the background literature relevant to the work presented in the thesis.

Chapter 3 presents a comparison of different classification techniques including decision tree inductive learner, k-Nearest Neighbour classifier, multilayer perceptrons and radial basis function neural networks.

Chapter 4 describes an enhancement of the Learning Vector Quantisation technique called All Weights Updating.

Chapter 5 describes another enhancement of the Learning Vector Quantisation technique called Neighbouring Weights Updating.

Chapter 6 presents the conclusions of the research and recommendations for further study.

Appendix A details the examples data extracted from the Massachusetts Institute of Technology-Beth Israel Hospital (MIT-BIH) database as used in this study.

Appendix B Statistical Formulae used in Feature Selection.

Chapter 2

Review of Automated ECG Classification

2.1 Preliminaries

An electrocardiogram (ECG) is an important tool in clinical diagnosis relating to disorders of the human heart. It records the electrical activity of the heart in terms of voltage changes transmitted to the body surface by electrical events in the heart.

Automatic ECG beat detection and classification is an essential tool for clinical settings, such as an intensive care unit. Such systems, however, must be able accurately to detect and classify problems on a real time basis. It is known that several arrhythmias are potentially dangerous and even life threatening if not detected within a few seconds of their onset.

One major problem facing today's automatic ECG analysis equipment is the wide variation in the morphologies of ECG waveforms of different patients and patient groups.

This chapter gives a review of the physiology and functionality of the human heart and how ECG signals are measured including the placement of the leads, and the nature and characteristics of ECG signals. The Chapter also provides a description of the database used in this work and a literature review on ECG arrhythmia classification.

2.2 The Heart

The heart is a muscle weighing approximately 300 grams. It consists mainly of two types of cells, those which form the working muscle, and those which control electrical signals. Its main function is to maintain the supply of blood in the body and to deliver it to all parts of the body via the arteries. In order to achieve this, the heart muscle must pump approximately 7200 litres of blood per day. At rest, the normal heart beats around 72 times and pumps around 5 litres of blood per minute. According to the demands of the body, it is capable of rapidly increasing this to 15 litres per minute with a heart rate of up to 200 beats per minute or more.

Figure 2.1 shows the physical structure of the heart. The heart consists of two chambers [McCulloch and Bastion, 1999], which act as two synchronised pump stages. The first chamber comprises the atria, which consist of the left and right atria. The second chamber is made up of the ventricles, which consist of the left and right ventricles. The right atrium and right ventricle supply blood to the lungs for oxygenation (pulmonary circulation), whereas the left atrium and left ventricle supply blood to the rest of the body (systemic circulation).

Excitation of the heart does not come directly from the central nervous system but is initiated in the SinoAtrial (SA) node (see Figure 2.2), or pacemaker, which are a special group of excitable cells.

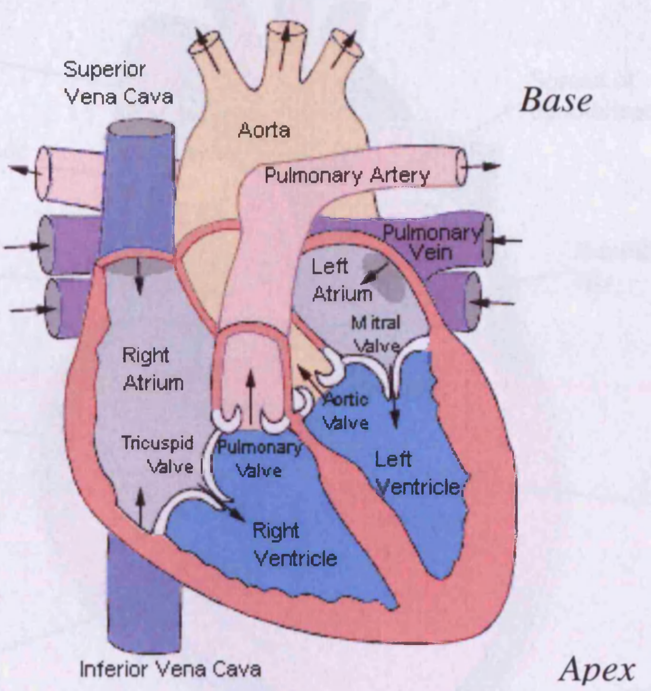


Figure 2.1: Cutaway drawing of the human heart showing the chambers, valves and connecting blood vessels (adapted from [Texas Heart Institute, 2001])

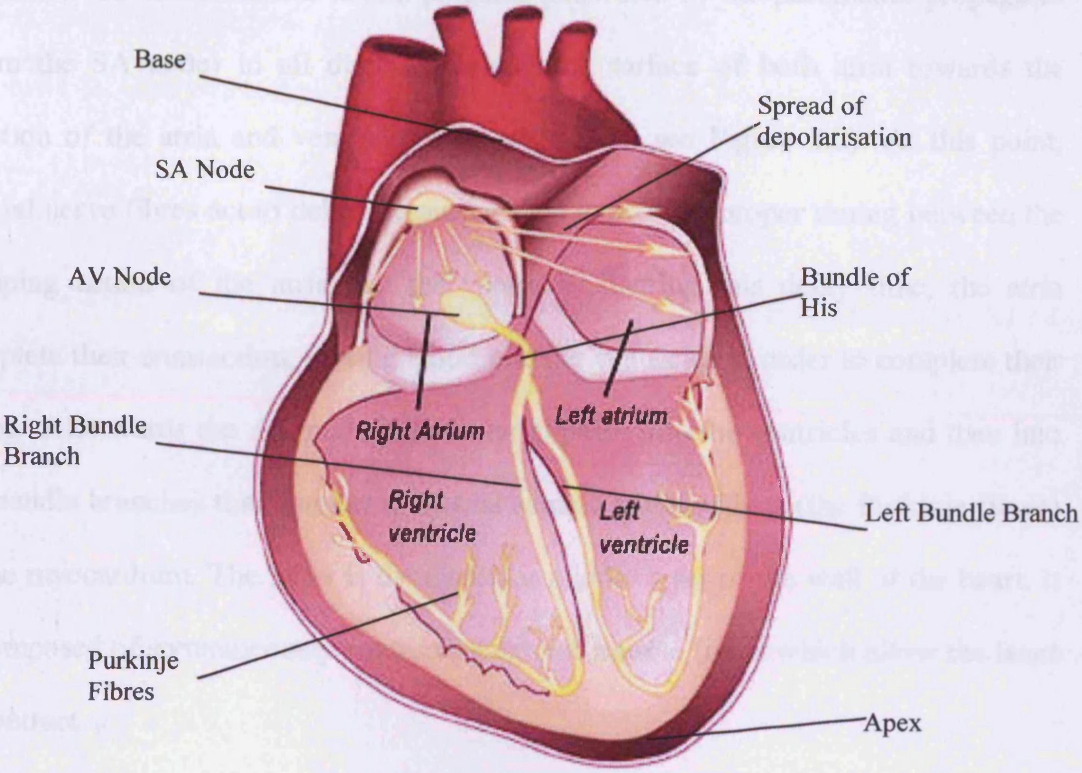


Figure 2.2: Electrical conduction pathways within the heart (adapted from [Wahl, 1999])

The pacemaker spontaneously generates action potentials at a regular rate, although the rate is influenced by nerves that accelerate or inhibit its value.

To initiate the heartbeat, the action potential generated by the pacemaker propagates (from the SA node) in all directions along the surface of both atria towards the junction of the atria and ventricles (the AV node - see Figure 2.2). At this point, special nerve fibres act to delay the propagation to provide proper timing between the pumping action of the atria and the ventricles. During this delay time, the atria complete their contraction, forcing blood into the ventricles in order to complete their filling. Afterwards the AV node initiates an impulse into the ventricles and then into the bundle branches that connect to special cardiac muscle fibres (the Purkinje fibres) in the myocardium. The latter is the muscular middle layer of the wall of the heart. It is composed of spontaneously contracting cardiac muscle fibres which allow the heart to contract.

The front wave in the ventricles, however, does not follow along the surface but is perpendicular to it and moves from the inside to the outside of the ventricular wall, until the entire ventricle becomes depolarised. Then the ventricles contract, forcing blood into the pulmonary and systemic circulatory systems. A wave of depolarisation follows the repolarisation wave by about 0.2 to 0.4 second. This depolarisation, however, is not initiated from neighbouring muscle cells but occurs as each cell returns to its resting potential independently.

2.2.1 Electricity in the Heart

The cardiac muscle contracts when it is stimulated by tiny electrical impulses carried in conductive fibres within the muscle. Unlike other muscles, the heart is unique. It generates its own electricity rather than receiving signals initiating in the brain and central nervous system. This allows the most important muscle in the body to function independently. In a normal healthy heart, these electrical impulses, of the order of 1 mV, originate in the SA node and follow a pathway around the heart in the conductive fibres. Conduction occurs due to potassium and sodium ions passing through cell walls [McCulloch and Bastion, 1999]. Figure 2.2 shows the electrical pathways within the heart.

To measure the heart's electrical activity, small metal leads are used. When a lead makes contact with the surface of the skin, which acts as an electrolyte, a difference of potential between the lead and the skin is produced.

The top layer of skin consists largely of dead cells along with a certain amount of oil and grime. Therefore, the skin's natural electrical resistance is high compared to the resistance of the body's fluids. An electrolytic jelly or paste is usually applied between the lead and the skin to ensure a low value of lead-skin interface resistance and minimise the drop in potential across the interface. Larger surface leads tend to have lower resistance compared to small needle leads.

2.2.2 Electrocardiogram

An ECG records the electrical activity of the heart, in particular the propagation of the electric potential through the cardiac muscles, as monitored by sensors at the limb extremities. It is considered a representative signal of the cardiac physiology, useful in diagnosing cardiac disorder. The state of cardiac health is generally reflected in the shape of the ECG waveform and heart rate. It may contain important pointers to the nature of diseases afflicting the heart [Acharya et al., 2003]. The biopotentials generated by the muscles of the heart result in the ECG.

2.2.3 Recording ECGs

The first accurate recording of an ECG was made by Einthoven in 1895 using a string galvanometer [Jenkins, 2001]. In order to record the ECG from electrodes placed vertically as well as horizontally on the human body, he had the electrodes placed not only on the chest but also in both arms and one leg. The leg selected was the left one, probably because it terminates vertically below the heart. Nowadays, an ECG is recorded by affixing up to five electrodes to the body [DeMarre and Michaels, 1983]. Usually two electrodes, or one electrode plus a group of up to three others, are connected to an amplifier. The electronic amplifier requires an additional connection to the body as a ground reference. The free right leg is used for this purpose. The lead placements are the right arm wrist (RA), left arm wrist (LA), left leg ankle (LL), right leg ankle (RL) and chest. As for the chest leads, different positions have been used.

Figure 2.3 shows the twelve standard lead positions over the body, and Table 2.1 lists the connections between the leads and their types (bipolar, and unipolar limb leads, unipolar chest leads).

2.2.4 ECG Wave-Form

The ECG wave-form for one cardiac cycle is shown in Figure 2.4. The letter designations given to each of the prominent features are those conventionally adapted.

The waveform is interpreted as follows [Hampton, 1998]:

The **P-wave** indicates the SA node function, which is produced by atrial depolarisation. During atrial depolarisation, the potential's action travels from the SA node towards the atrioventricular (AV) node.

The **R-wave** represents the depolarisation of most (but not quite all) of the remaining ventricular musculature. Because the ventricular muscle is massive compared to the atrial muscle, the R-wave amplitude is much higher than the P-wave amplitude. The R-wave, like the P-wave, appears above the baseline and is usually the most prominent feature in the ECG. The normal peak value of the R-wave is approximately 1 mV when measured at the surface of the body and about 40 mV when measured inside the heart.

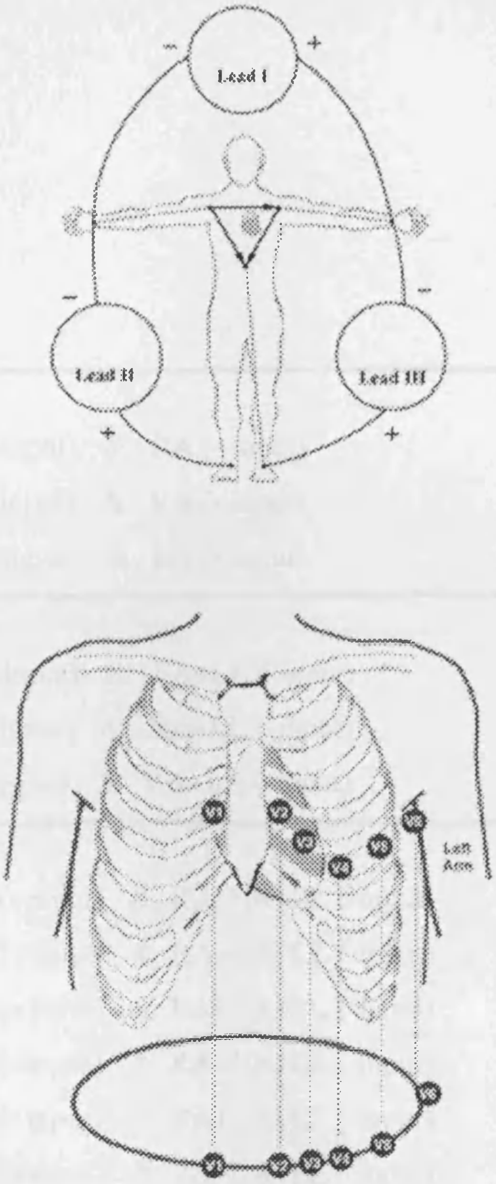


Figure 2.3: The twelve standard lead positions for recording the ECG

(adapted from [www.biopac.com])

Bipolar Limb Leads	
Lead I	LA (+input) & RA (- input)
Lead II	LL (+input) & RA (- input)
Lead III	LL (+input) & LA (- input)
Unipolar Limb Leads	
Lead aVR	RA (+input) & LA+LL (-input)
Lead aVL	LA (+input) & RA+LL (-input)
Lead aVF	LL (+input) & RA+LA (-input)
Unipolar Chest Leads	
V1	Chest (+input) & RA+LA+LL (-input)
V2	Chest (+input) & RA+LA+LL (-input)
V3	Chest (+input) & RA+LA+LL (-input)
V4	Chest (+input) & RA+LA+LL (-input)
V5	Chest (+input) & RA+LA+LL (-input)
V6	Chest (+input) & RA+LA+LL (-input)

RA = right arm wrist, LA = left arm wrist, LL = left leg ankle

+ input refers to the positive terminal of the ECG recorder

- input refers to the negative terminal of the ECG recorder

Table 2.1: The twelve standard lead positions used in ECG

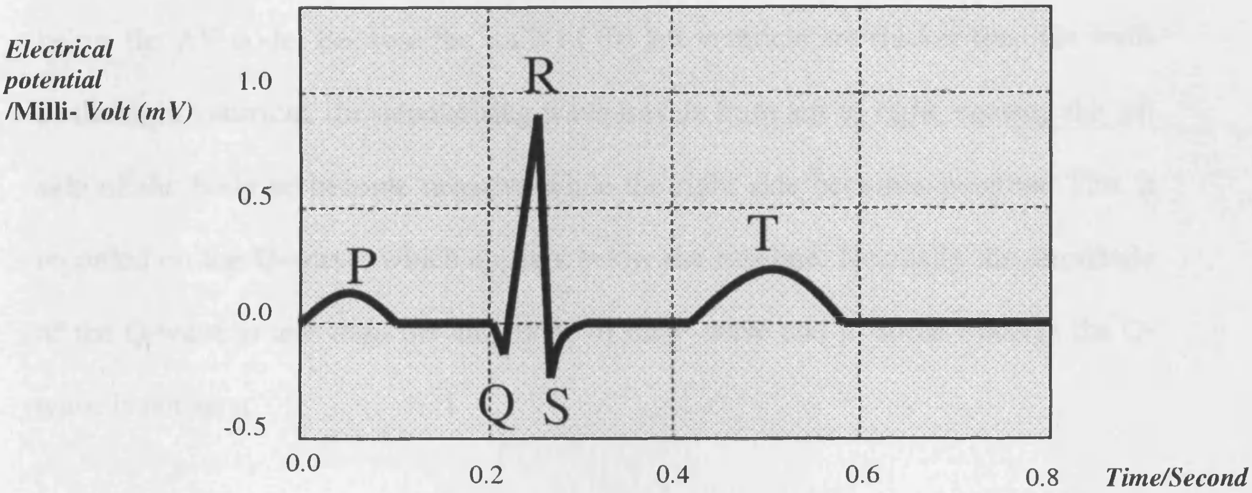


Figure 2.4: Idealised representation of an ECG trace (lead II) for one normal heartbeat

The **PR interval** indicates AV conduction time. The interval is measured from the onset of the P-wave to the beginning of the QRS complex. A normal PR interval is 0.18 to 0.20 seconds. A short PR interval indicates that the impulse originates in an area other than the SA node, and a long PR interval indicates that the impulse is delayed as it passes through the AV node. Whereas atrial depolarisation occurs in one direction, the ventricles depolarise in three directions. Immediately after the impulse delay period, initial depolarisation of the ventricles begins in the septal area just below the AV node. Because the walls of the left ventricle are thicker than the walls of the right ventricle, the depolarising wave travels from left to right, causing the left side of the body to become negative while the right side becomes positive. This is recorded on the **Q-wave**, which appears below the baseline. Normally the amplitude of the Q-wave is less than the amplitude of the P-wave and in some tracings the Q-wave is not seen.

The **S-wave** represents the depolarisation of the remaining portion of the ventricles. Since for this wave, the apex (see Figure 2.1) becomes negative while the AV node area becomes positive, the recorded S-wave appears below the baseline. In general, the amplitude of the S-wave is greater than that of the Q-wave.

However, for some patients, the S-wave amplitude is so small that it is not observable. The QRS complex is the combined result of the repolarisation of the ventricles. Prior to this time interval, the atria are depolarising. However, because of its small amplitude, the depolarising wave pattern of the atria is not measurable at the body's surface. Thus the baseline normally remains flat between the end of the P-wave and the start of the QRS complex.

As the ventricles begin to depolarise, they contract. Later they depolarise. Ventricular depolarisation is represented by the **T-wave**.

Occasionally, another wave will appear after the T-wave. This is called the **U-wave,** and is generally believed to be the result of after potentials in the ventricular muscle.

The **U-wave** is more frequently seen in tracings from infants and patients who have low serum potassium levels or an enlarged heart. Following depolarisation, the ventricles relax.

Table 2.2 summarises the values of different intervals and segments of a normal ECG signal with different heart rates. This table shows that the heart rate affects the ECG signals even if the person is healthy.

2.3 Cardiac Arrhythmia

Arrhythmias, also known as dysrhythmias, refer to any disorder of heart rate or rhythm. In a normal heart, the atria and ventricles contract in a co-ordinated manner. The depolarisation wave spreads from the SA node, through the atria, to the AV node and through the bundle of His (see Figure 2.2) and bundle branches into the ventricles. A conduction abnormality can occur at any of these points.

2.3.1 MIT-BIH Arrhythmia Database

The Massachusetts Institute of Technology Beth Israel Hospital (MIT-BIH) arrhythmia database [Moody and Darker, 1989] is a well-established source of

P-R Interval	QRS interval	Rate	Q-T Interval	S-T Segment
0.18 to 0.20 Second	0.07 to 0.01 Second	60	0.33 to 0.43 Second	0.14 to 0.16 Second
		70	0.31 to 0.41 Second	0.13 to 0.15 Second
		80	0.29 to 0.38 Second	0.12 to 0.14 Second
		90	0.28 to 0.36 Second	0.11 to 0.13 Second
		100	0.27 to 0.35 Second	0.10 to 0.11 Second
		120	0.25 to 0.32 Second	0.06 to 0.07 Second

Table 2.2: Summarised values of different intervals and segments of the normal ECG signal with different heart rates

ECG data for researchers studying ECG classification techniques. It contains digitised ECG signals, which have been transferred from Holter tape recordings taken from various in-patients at the Arrhythmia hospital laboratory at the Beth Israel Hospital between 1975 and 1979. From 4000 Holter tape records, 48 annotated records divided into two groups were kept. Group one consists of 23 records (the 1xx series) and contains examples that an arrhythmia detector might encounter in routine clinical use. The second group consists of 25 records (the 2xx series) and contains examples of complex arrhythmias that could pose difficulties to arrhythmia detectors or of rare clinical cases. The subjects were 25 men aged 32 to 89 years, and 22 women aged 22 to 89 years. About 60% of the records were obtained from in-patients. Each record is slightly over 30 minutes in length. The signals were sampled at the same frequency, 360 Hertz, but not necessarily at the same gain because during collection different equipment was used with different electrical gains for digitisation of the various records. Moreover, the digital amplitude values range between [0, 2047], where 1024 represents 0 volts. Therefore, the signals require normalisation before use.

The variety of the patients and variation in their ages and physical conditions makes the MIT-BIH database suitable for investigations into ECG classification techniques.

Lead II was the lead type used to record most of the ECG signals in the MIT-BIH database.

The MIT-BIH Arrhythmia database contains software to enable extraction of the digitised records. For the purpose of this study the following ECG types were selected from the MIT-BIH database:

1- Normal Sinus Rhythm (N): this is the term for the normal condition (see Figure 2.5).

2- Left Bundle Branch Block Beat (L): this arrhythmia is caused by a problem in conduction in the His bundle in the left side ventricle. This is seen as a widening of the QRS complex. This ECG type is invariably an indication of heart disease [Hampton, 1998]. Figure 2.6 indicates that the QRS complex is notably wider than that shown in Figure 2.5. This is due to the extra time taken for depolarisation caused by poor electrical conduction (block).

3- Right Bundle Branch Block Beat (R): the cause of this arrhythmia is similar to **(L).** However, the conduction problem now occurs on the right side of the His bundle branch and the ECG indicates a problem in the heart but also can be seen in a healthy heart. This type of arrhythmia is identified by a wide bimodal QRS complex (see Figure 2.7).

4- Paced Beat (P): this problem arises in patients that have been fitted with an artificial pacemaker. Pacemakers are used when a person has bradycardia (a very slow heart rhythm), which causes poor circulation and cannot be corrected by treatment with drugs. Pacemakers stimulate the heart muscle. This type of arrhythmia is indicated by the occasional missing of the P-wave and the presence of a spike

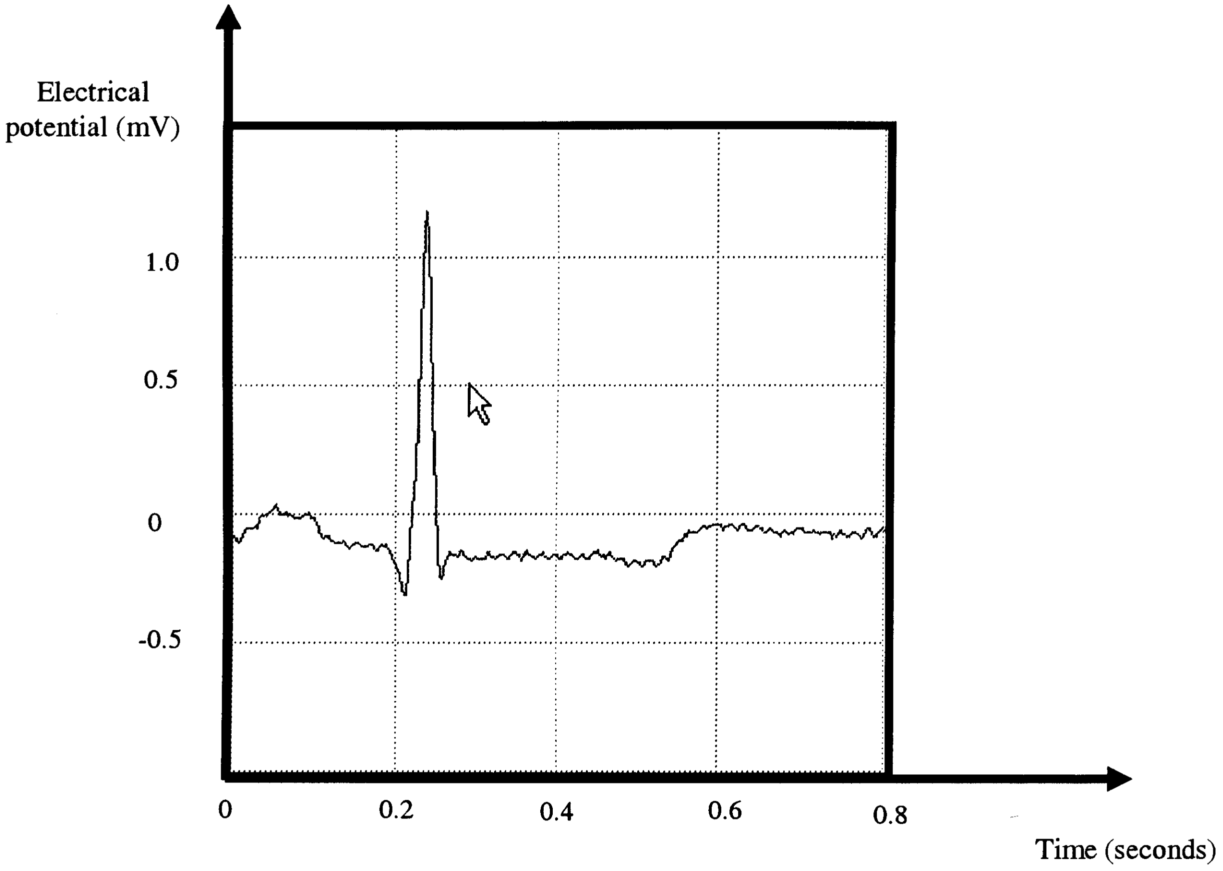


Figure 2.5: Normal sinus rhythm (N) Type

(MIT-BIH database, record 100)

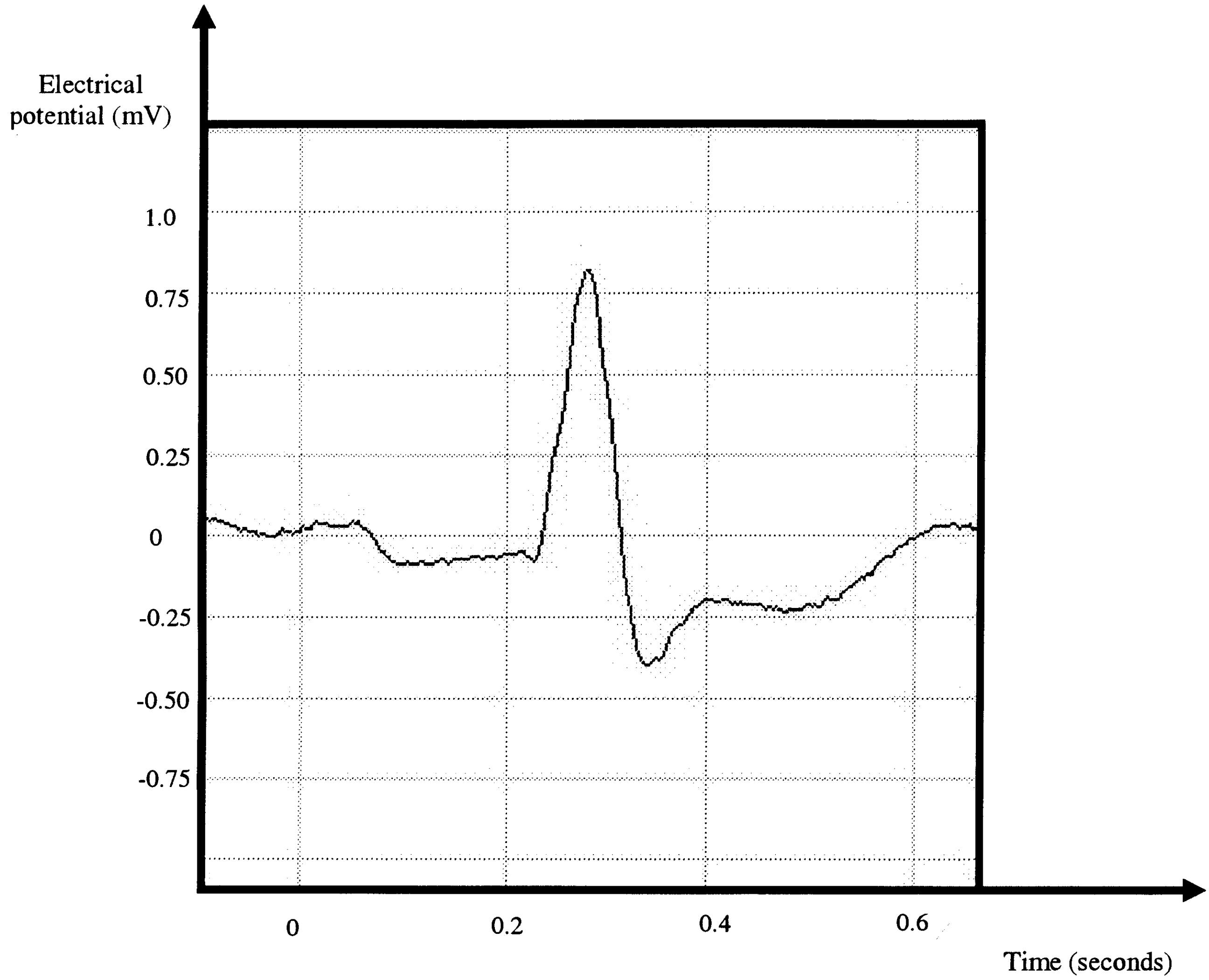


Figure 2.6: Left bundle branch block (L) type

(MIT-BIH database, record 109)

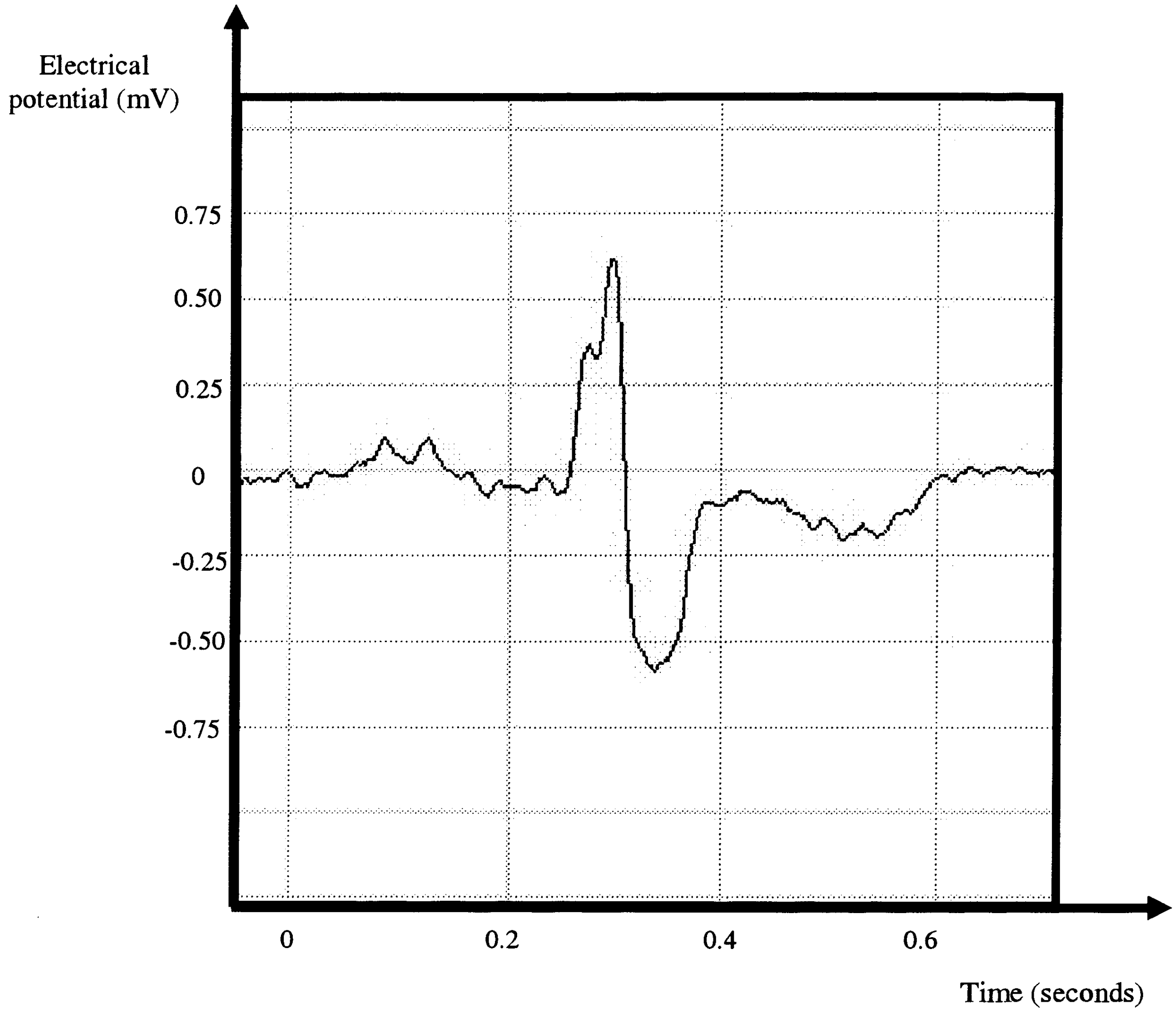


Figure 2.7: Right bundle branch block (R) type

(MIT-BIH database, record 118)

representing the stimulus from the pacemaker, followed by a wide QRS complex (see Figure 2.8).

5- Premature Ventricular Contraction (V): this arrhythmia occurs when the heart beats earlier than it should. This is because of the abnormal electrical activity of the ventricles which causes premature contraction of the lower chambers of heart, the ventricles. The premature contraction is followed by a pause as the heart's electrical system "resets" itself. The contraction following the pause is usually more forceful than normal. With this type, the QRS complex is misshapen and prolonged representing ventricular contraction without earlier atrial stimulation (see Figure 2.9).

6- Atrial Premature Beat (A): this arrhythmia is associated with early depolarisation of atrium. This type can be identified by a premature, small and distorted P-wave (see Figure 2.10).

7- Aberrated Atrial Premature Beat (a): early depolarisation of atria. This manifests itself as an abnormal P-wave (wide prolonged), narrow R-wave, and distorted QRS complex (see Figure 2.11).

8- Nodal (junctional) Escape Beat (j): the cause of this arrhythmia is that the region around the AV node takes over as the focus of the depolarisation; the rhythm is called "nodal" or 'junctional' escape. Figure 2.12 shows one beat cycle of this arrhythmia which has no Q- and S-waves. Also, the P-wave has an inverse polarity compared to that of the normal sinus rhythm.

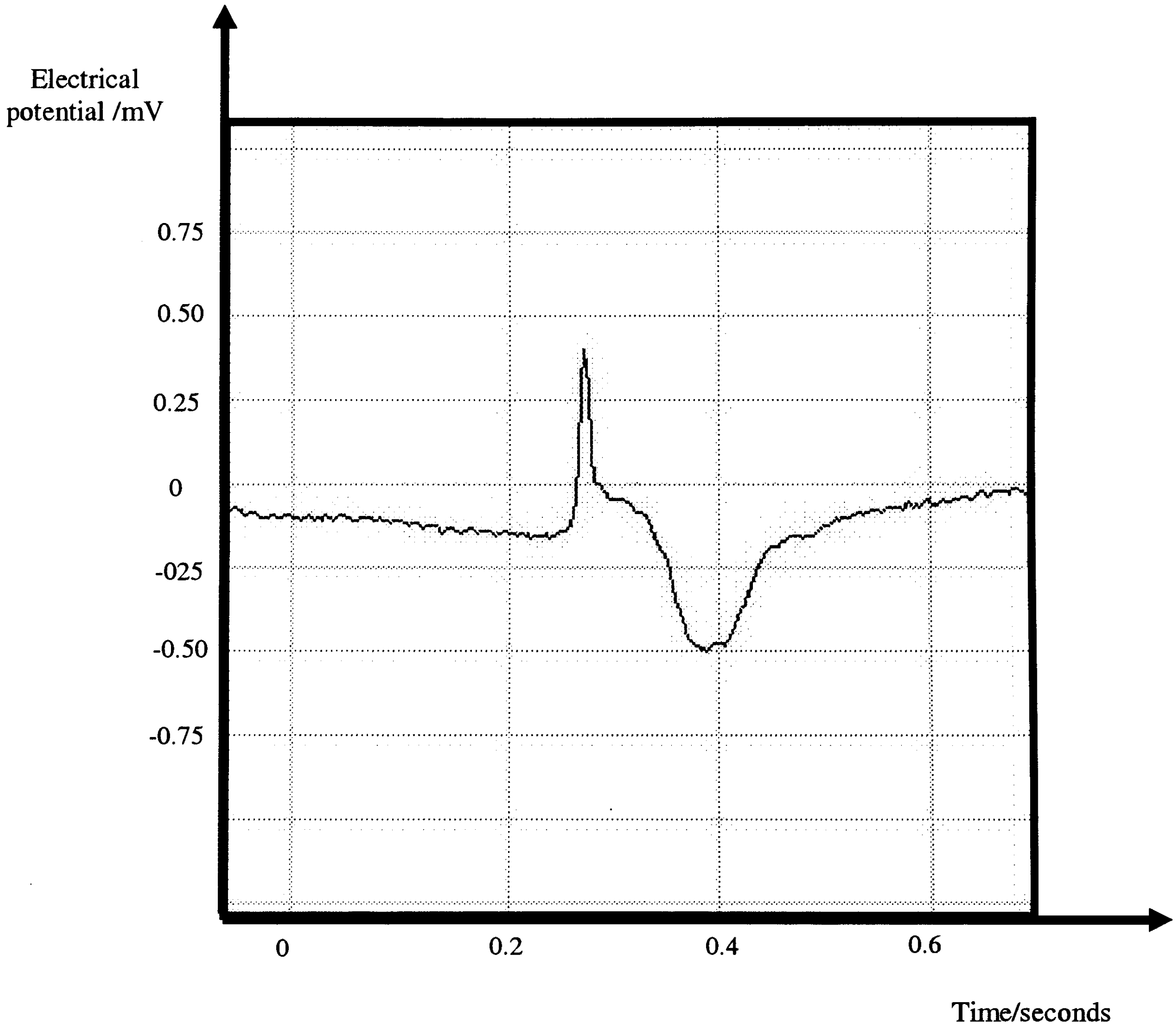


Figure 2.8: Beat stimulated by an artificial pacemaker ('Pace') type

(MIT-BIH database, record 104)

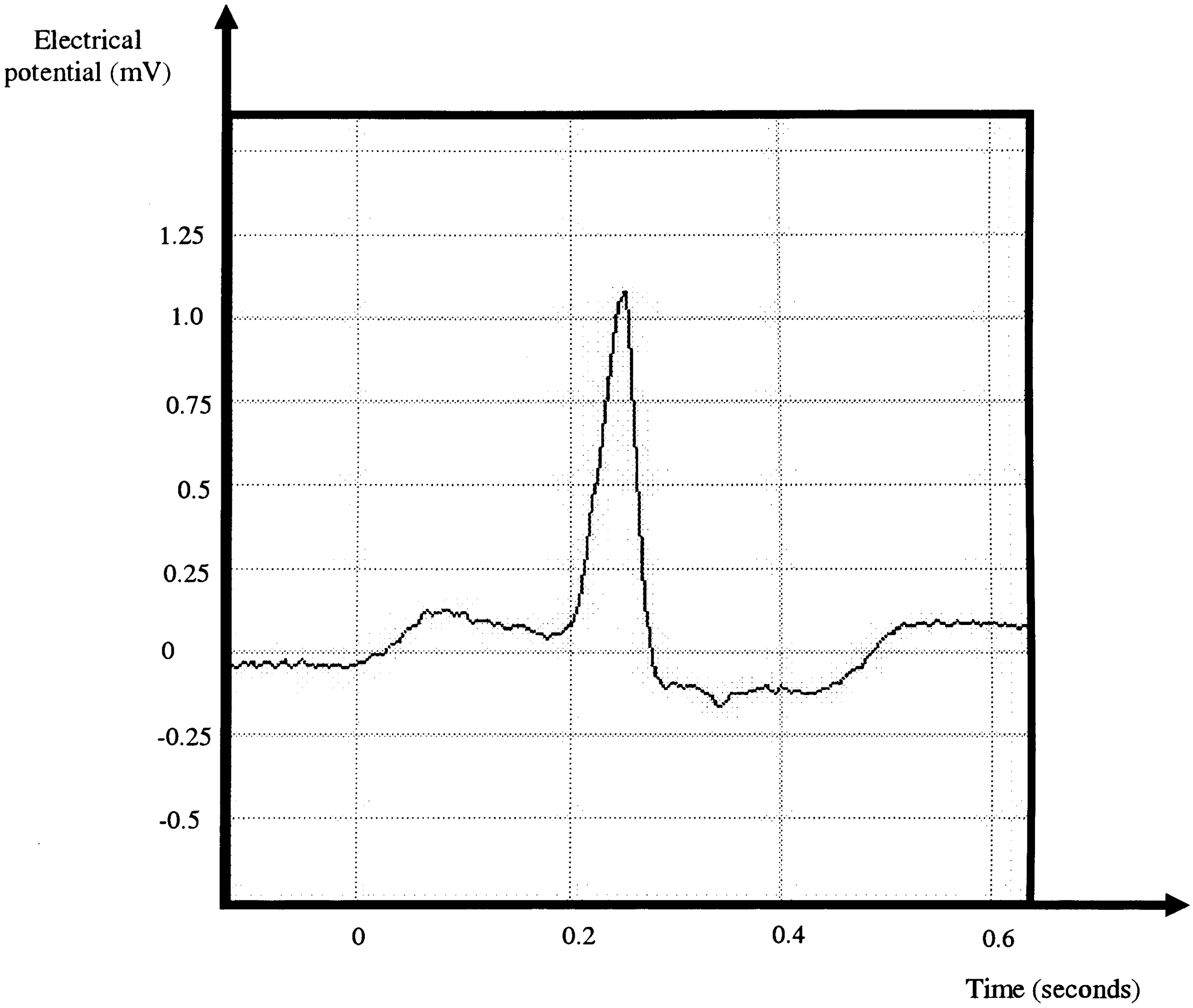


Figure 2.9: Premature ventricular contraction (V) type

(MIT-BIH database, Record 105)

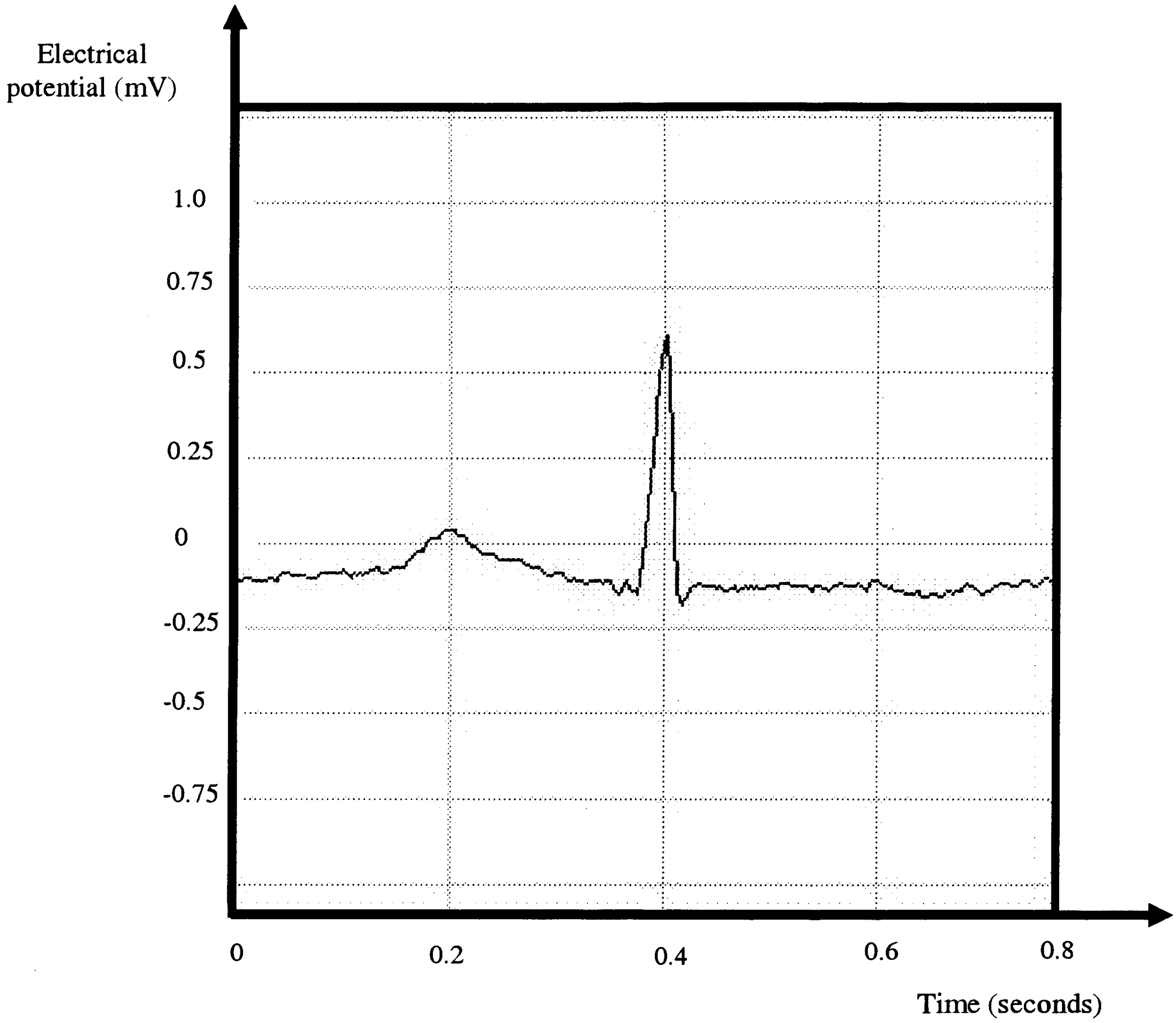


Figure2.10: Atrial premature beat (A) type

(MIT-BIH database, Record 100)

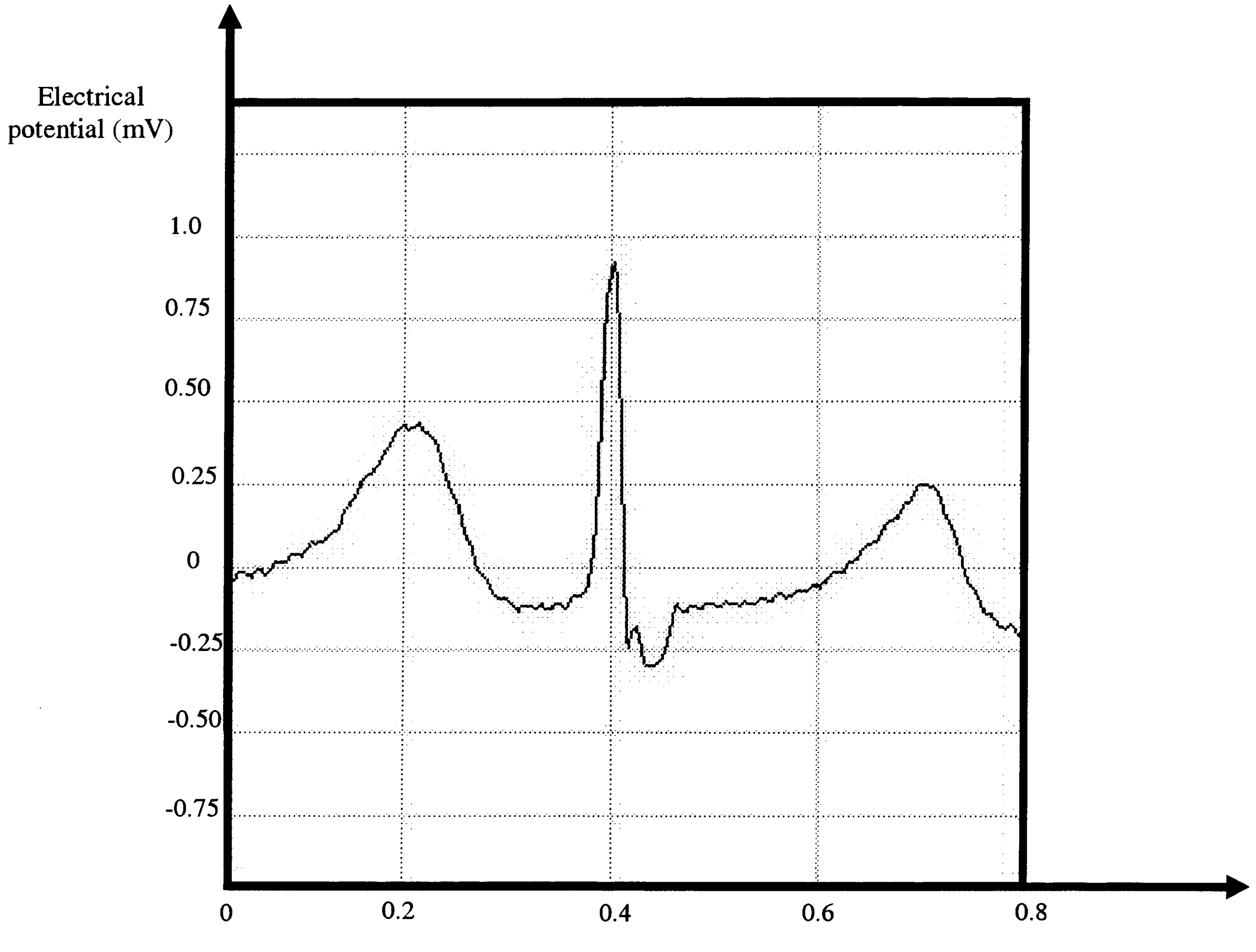


Figure 2.11: Aberrated atrial premature beat (a) type

(MIT-BIH database, Record 105)

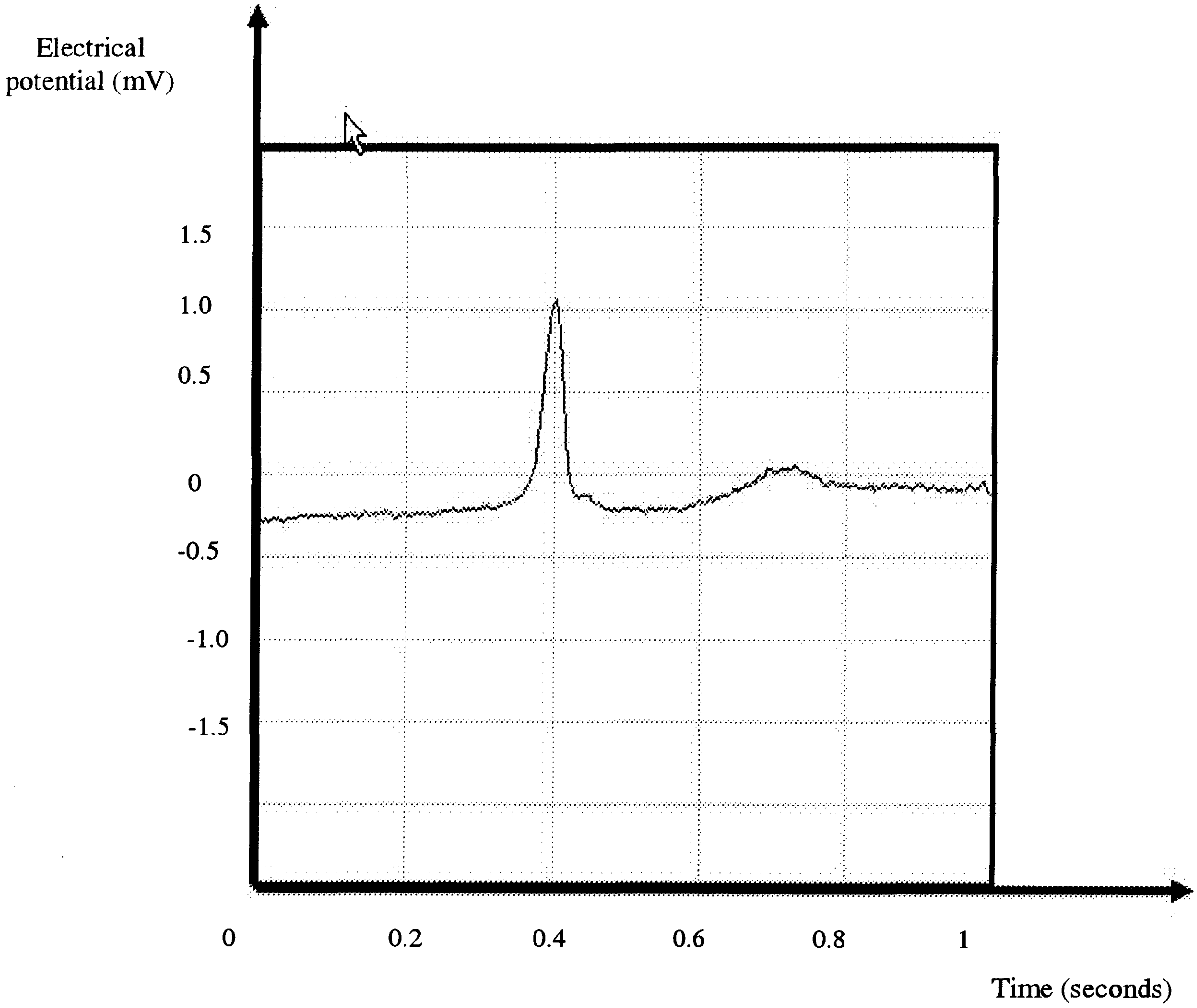


Figure 2.12: Nodal (junctional) escape beat (j) type

(MIT-BIH database, Record 201)

9- Ventricular Escape Beat (E): this most commonly occurs when the ventricle contracts without nodal stimulation. This is classically associated with complete heart blockage. The QRS complexes are wide whereas the P-waves are occasionally absent as demonstrated in Figure 2.13.

10- Fusion of paced and normal beats (f): this type of arrhythmia is a mixture of paced and normal beats. The P-waves have large amplitudes and are wide, and the QRS complexes are distorted, especially in the S-waves portion (see Figure 2.14).

Examples of the above arrhythmias and normal ECGs were extracted from records 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 111, 112, 113, 114, 115, 116, 117, 118, 119, 121, 122, 123, 124, 200, 201, 202, 203, 205, 207, 208, 209, 210, 212, 213, 214, 215, 217, 219, 220, 221, 222, 223, 228, 230, 231, 232, 233, 234.

There are two points to be taken into account concerning the above examples: intra-patient and inter-patient variability. Intra-patient variability occurs due to changes in the patient's emotional and physical states and inter-patient variability is due to different physical conditions between the different patients. As a result of intra- and inter-patient variability, different beat waveforms and different lengths of a beat cycle are observed.

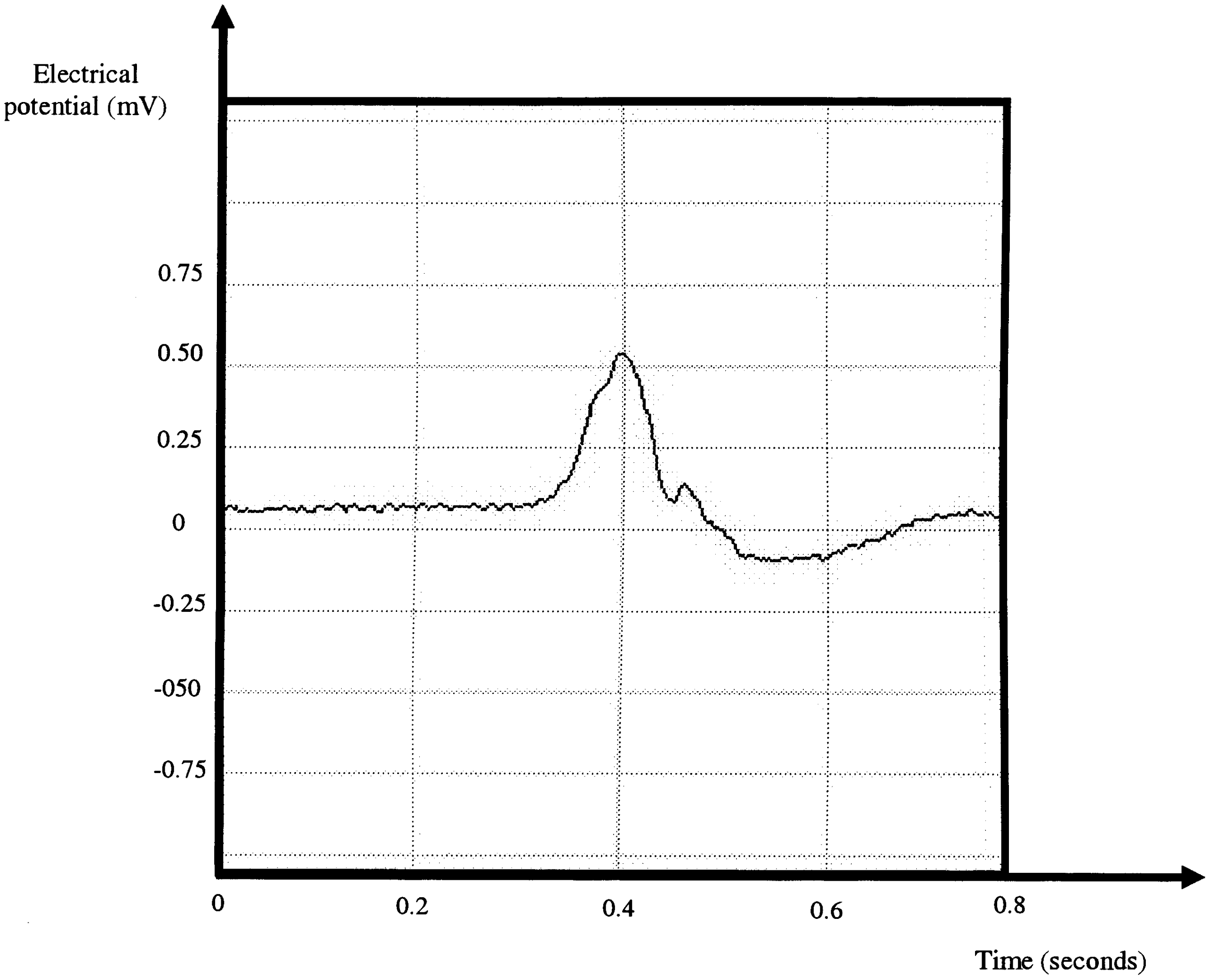


Figure 2.13: Ventricular escape beat (E) type

(MIT-BIH database, Record 207)

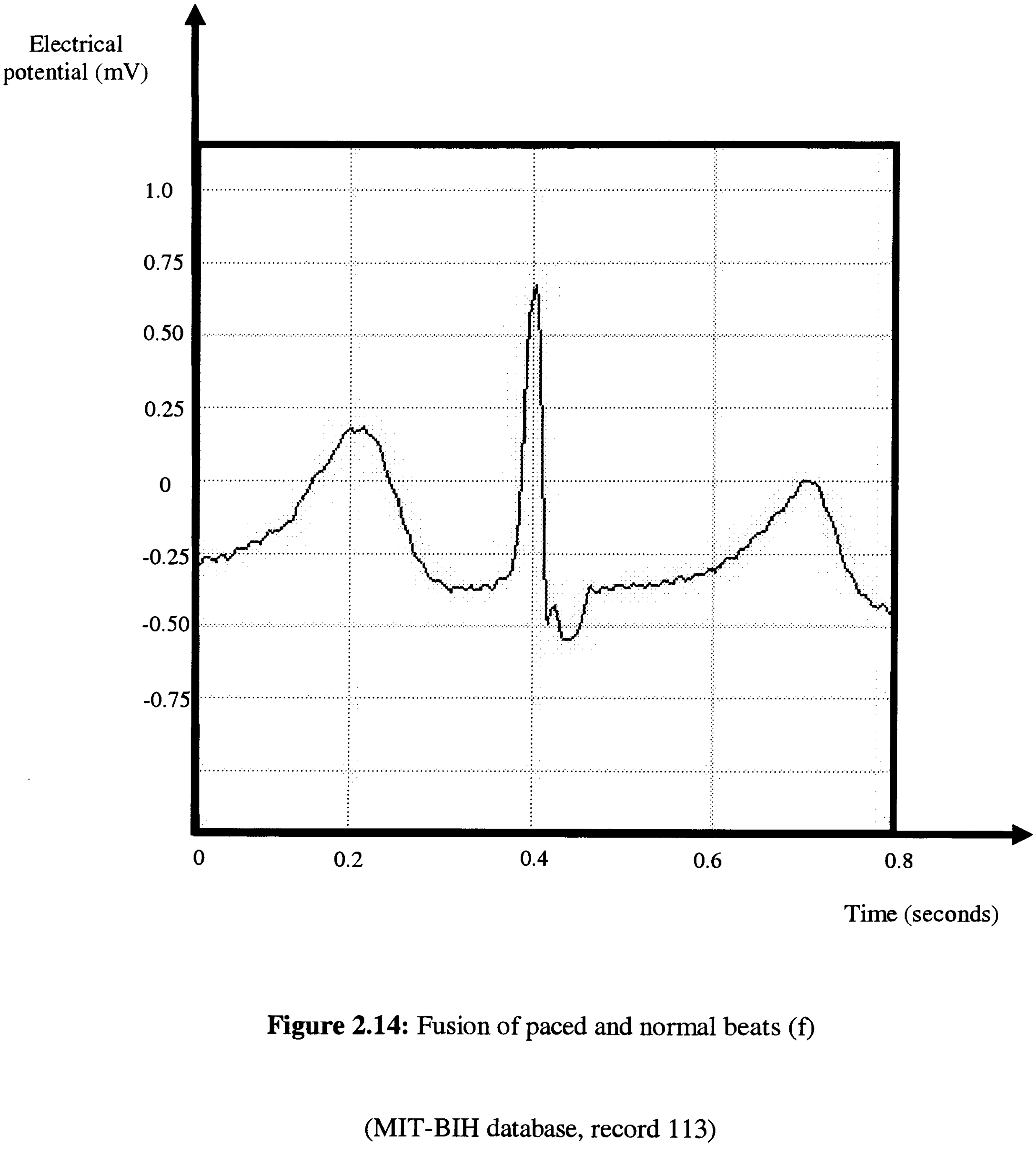


Figure 2.14: Fusion of paced and normal beats (f)

(MIT-BIH database, record 113)

2.4 Data Preparation

The application of a pattern classifier first requires the selection of features that must be tailored separately for each problem domain. Features should contain the information required to distinguish between classes, be insensitive to irrelevant variability in the input, and also limit the amount of training data required. Good classification performance requires selection of effective features and also selection of a classifier that can make good use of those features without demanding large amounts of training data, memory, and computing power [Lippmann, 1989].

Raw ECG data cannot be directly used for classification; it needs to undergo pre-processing operations such as filtering, digitisation, feature extraction and normalisation. The MIT-BIH Institute, which produced the ECG database used in this study, had carried out the filtering and digitisation.

The next section explains the preparation of ECG data in terms of feature extraction and normalisation.

2.4.1 Feature Extraction

Feature extraction techniques try to reduce the amount of data to be processed by a pattern classifier by extracting important features that can be used to represent the whole data set. Primary features are directly extracted from a data set. Additional features can be obtained by combining primary features by means of mathematical operations, for example taking the difference or the ratio of two feature values.

The MIT-BIH database consists of records, each pertaining to several ECG beats, with each digitised beat comprising hundreds of points representing one cycle of the ECG. It is very important to reduce these large data files to small files that retain sufficient information to enable differentiating between the different types of arrhythmia. The main problem when classifying an ECG is to determine which portion of the signal to use for diagnosis. Different portions and various numbers of features can be adopted. Many researchers use the QRS-complex as it represents ventricular depolarisation, and because it contains most of the information about the nature of disease [Conde, 1994; Suzuki, 1995; Hosseini and Nazeran, 1999; Biel, 2001]. However, the ST segment should also be taken into account as well as beat rhythm information [Weisner et al., 1982].

It has been noted by Suzuki [Suzuki, 1995] that since the QRS complex is a reference used to detect other waves, the first step in an automatic ECG interpretation system is recognition of this QRS complex. It is also true that the QRS complex contains a significant amount of information about the state of the heart, as it represents ventricular depolarisation [Biel, 2001].

Various sets of features were extracted for this study. In the work reported in chapter 3, two sets of extracted features (one with 18 and the other with 11 features) were used. Another set of 15 features was employed in the experiments reported in chapters 4 and 5. Figure 2.15 shows some of the features extracted from one ECG signal.

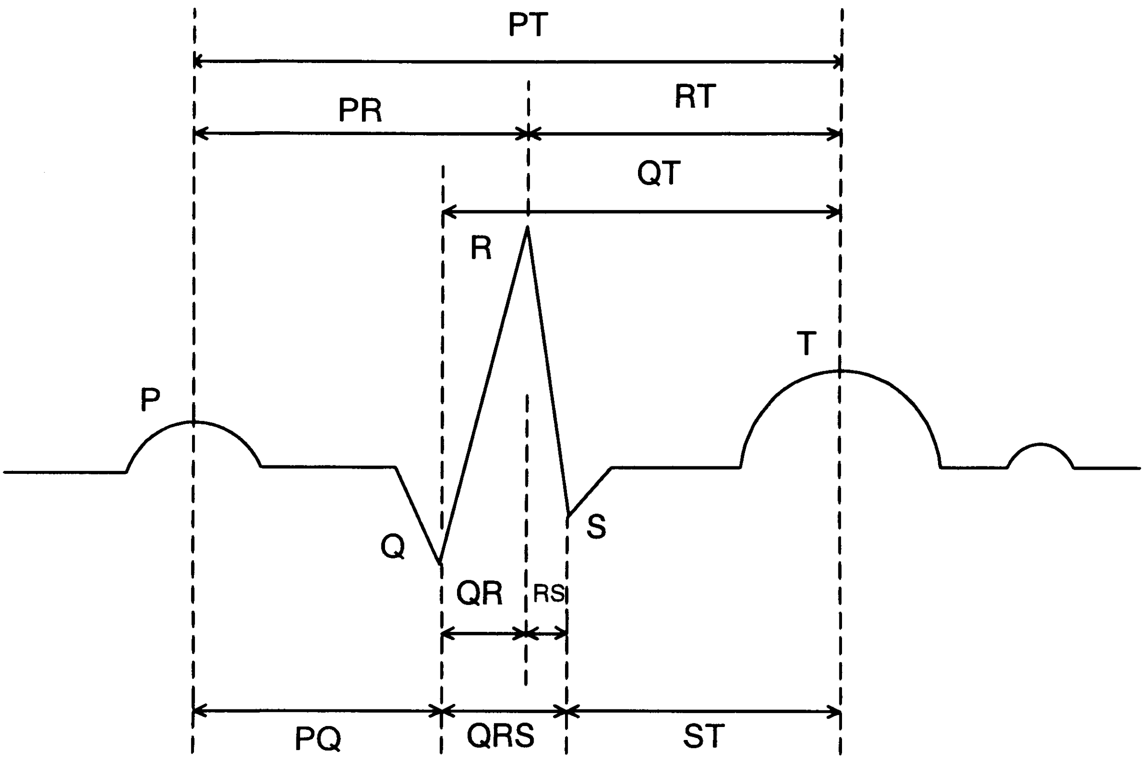


Figure 2.15: Some features extracted from an ECG

These features include some of the important information about the ECG signal such as the height and duration of different parts of the waveform. Table 2.3 and Table 2.4 respectively, show the 18 and 11 features used in the work described in chapter 3. The second set of features was obtained by removing seven less important features from the first set.

The process of physiological feature extraction starts with the identification of the R-wave, as the QRS complex is the most prominent part of the ECG as mentioned before. The R-wave is normally positive and generally shows a significantly greater peak than the other waves and the R peak can be detected as the highest value in the cycle.

However, the R-wave can sometimes be negative if the electrodes are attached with reversed polarity, and an inverted waveform is seen; in this case the Q- and S-waves are shown as positive.

On some occasions, the T-wave can have a greater magnitude than the R-wave. In this case, distinction is still possible because the R-wave has a pointed peak whereas T-waves are wider and more rounded.

As for the Q-wave and R-wave peaks, they may then be found by searching for a local maximum or minimum. In the case of the Q-wave, the search proceeds from the peak of the R-wave towards the left (i.e. the search is made backwards in time). For the S-wave, the search proceeds forward in time from the location of the R-wave peak.

Feature	Description
1- P peak height	The amplitude of the P-wave.
2- Q peak height	The amplitude of the Q-wave.
3- R peak height	The amplitude of the R-wave.
4- S peak height	The amplitude of the S-wave.
5- T peak height	The amplitude of the T-wave.
6- PT wave duration	Overall duration of the P & T-wave.
7- PR wave duration	Overall duration of the P & R-wave.
8- RT interval	Overall duration of the R & T-wave.
9- QRS interval	Overall duration of the QRS complex: from the onset of the Q-wave to the end of the S-wave. (Time taken for complete ventricular pumping action.)
10- QT interval	Overall duration of the Q & T-wave.
11- QR interval	Overall duration of the Q & R-wave.
12- Minimum	The minimum value of the ECG signal
13- Maximum	The maximum value of the ECG signal
14- RS interval	Overall duration of the R & S-wave.
15- PQ interval	Overall duration of the P & Q-wave.
16- ST interval	Overall duration of the S & T-wave.
17- Standard deviation	Standard deviation of the electrical signal from the Baseline
18- Mean	Mean of the electrical signal

Table 2.3: 18 Features of the ECG signal selected for classification

Feature	Description
1- PT wave duration	Overall duration of the P & T-wave.
2- PR wave duration	Overall duration of the P & R-wave.
3- RT interval	Overall duration of the R & T-wave.
4- QRS interval	Overall duration of the QRS complex: from the onset of the Q-wave to the end of the S-wave. (Time taken for complete ventricular pumping action.)
5- QT interval	Overall duration of the Q & T-wave.
6- QR interval	Overall duration of the Q & R-wave.
7- Minimum	The minimum value of the ECG signal
8- Maximum	The maximum value of the ECG signal
9- RS interval	Overall duration of the R & S-wave.
10- PQ interval	Overall duration of the P & Q-wave.
11- ST interval	Overall duration of the S & T-wave.

Table 2.4: 11 Features of the ECG signal selected for classification

Several variations from the ideal shape were found in the real examples from the MIT-BIH database even for the normal type. Some variations in the QRS complex were taken into account during the feature selection process see Figures (2.16(a), (b) and (c)), (2.17(a), (b) and (c)) and (2.18(a), (b), and (c)).

As shown in Table 2.3 and Table 2.4, a number of statistical features was also used.

Another method of reducing the data set used to describe the ECG was to re-sample at a reduced sampling frequency. This method is simpler and faster than the feature extracting methods.

For the experiments reported in chapter 3, two sets of re-sampled data were used. The re-sampling frequencies were 100 Hz and 50 Hz for the two sets respectively, giving 64 points or 33 points for one cycle of the ECG signal.

2.4.2 Normalisation

As mentioned previously, during collection different equipment was employed with different electrical gains for digitisation of the various records. Consequently, the signals required normalisation before use, and for convenience in neural network training, the data was normalised before being presented to the neural network.

Usually, the data would be normalised between 0 and 1 or between -1 and 1. The main advantage of normalisation is to eliminate the effects of different scales and ranges. In this study, the data was normalised in the range [–1, 1]:

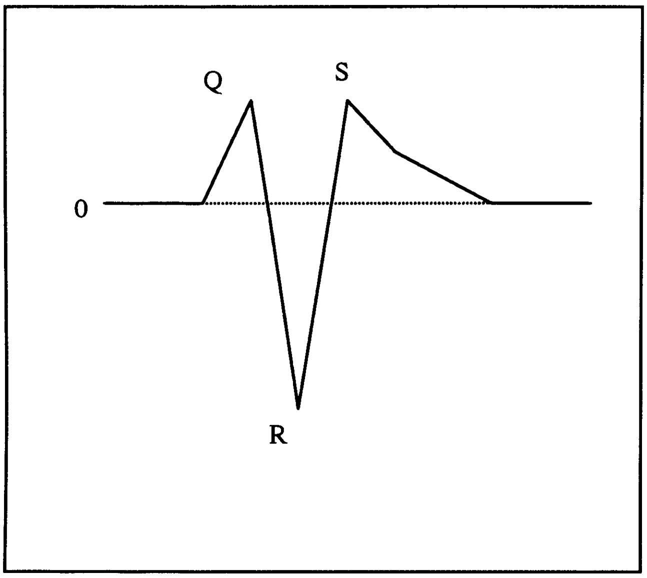


Figure 2.16 (a): Variations in the R-wave part of QRS complex showing inverted R-wave

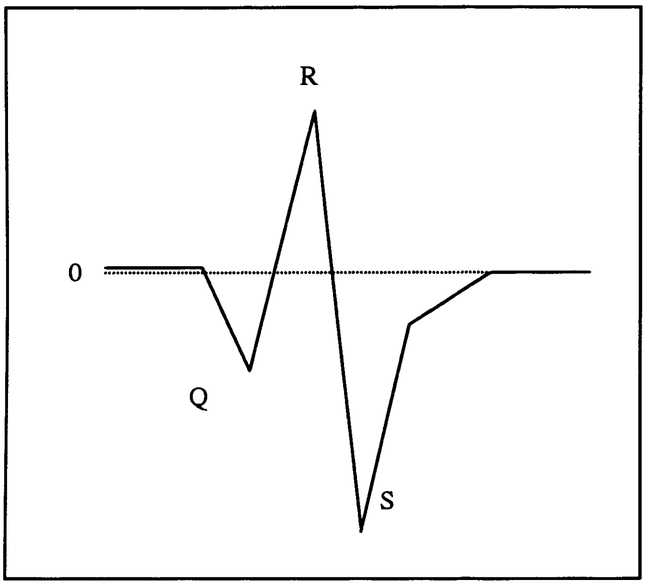


Figure 2.16 (b): Variations in the R-wave part of QRS complex showing S-wave has greater magnitude than R-wave

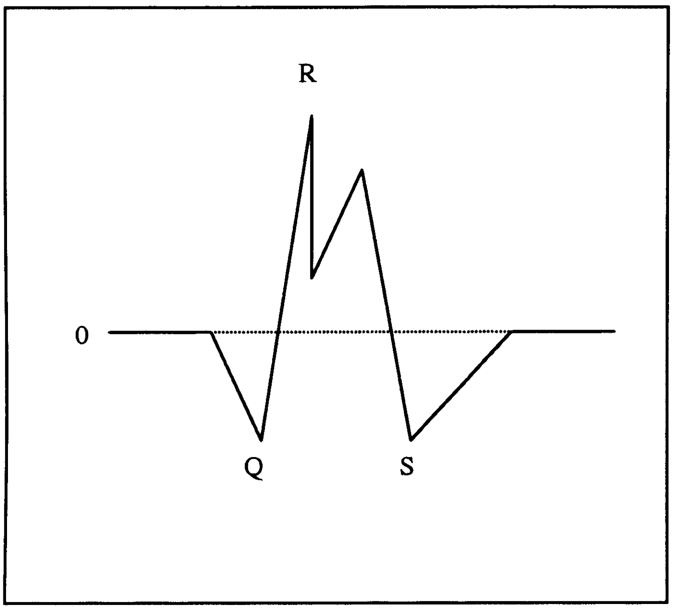


Figure 2.16 (c): Variations in the R-wave part of QRS complex showing apparent double R peak

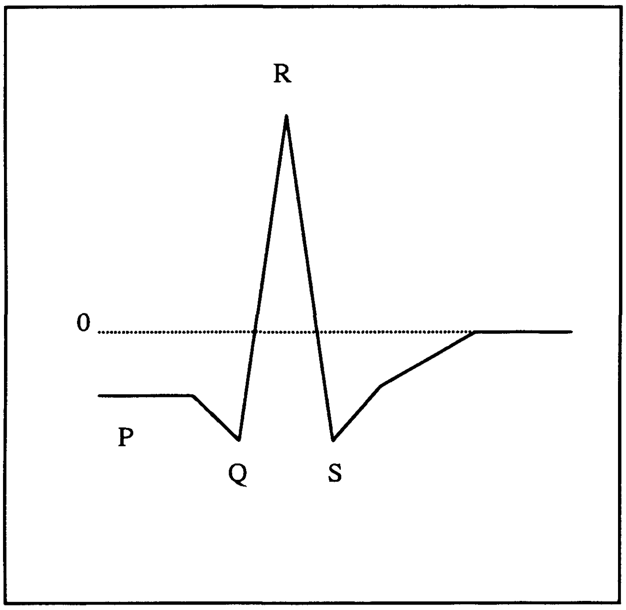


Figure 2.17 (a): Variations in the Q-wave part of QRS complex showing depressed PR segment

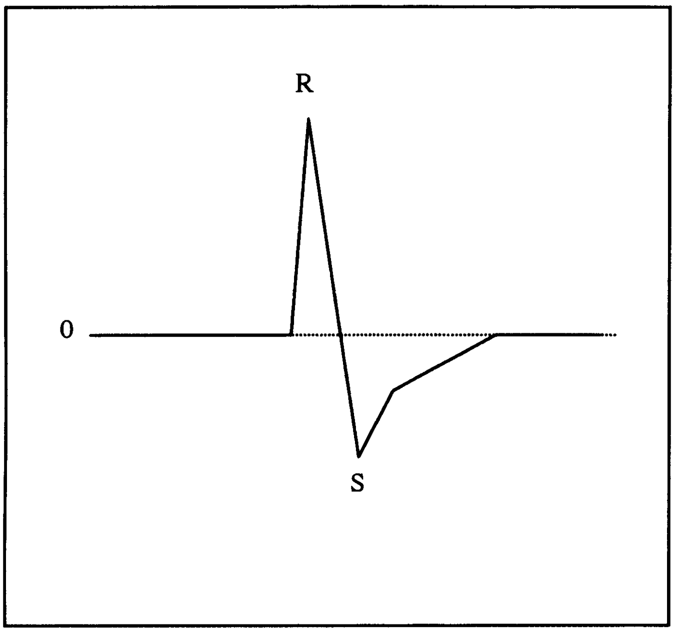


Figure 2.17 (b): Variations in the Q-wave part of QRS complex showing no Q-wave present

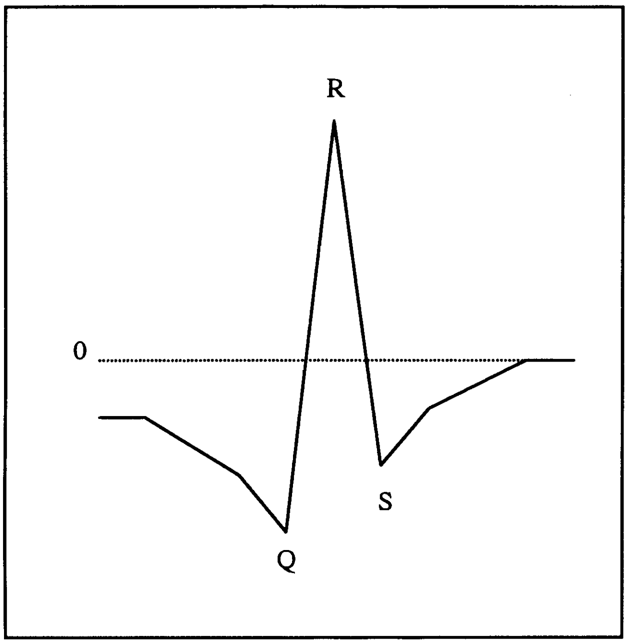


Figure 2.17 (c): Variations in the Q-wave part of QRS complex showing indeterminate onset to Q-wave

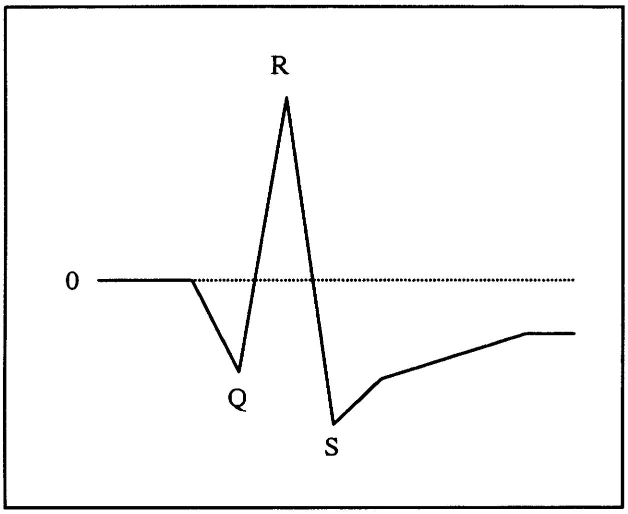


Figure 2.18 (a): Variations in the S-wave part of QRS complex showing indeterminate end to S-wave

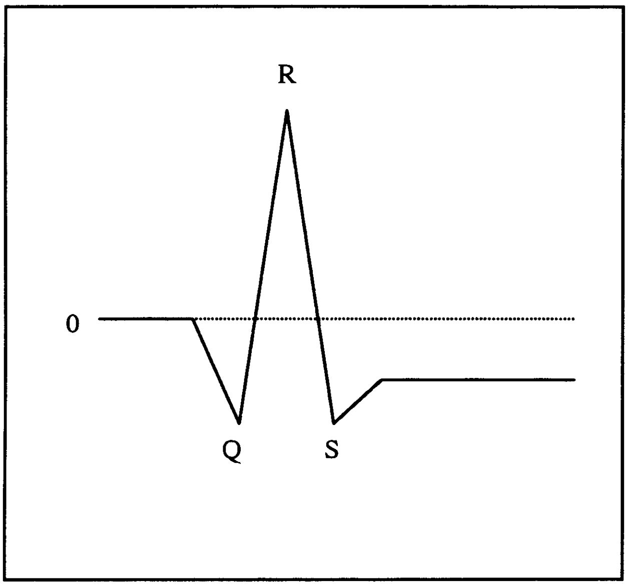


Figure 2.18 (b): Variations in the S-wave part of QRS complex showing depressed ST segment

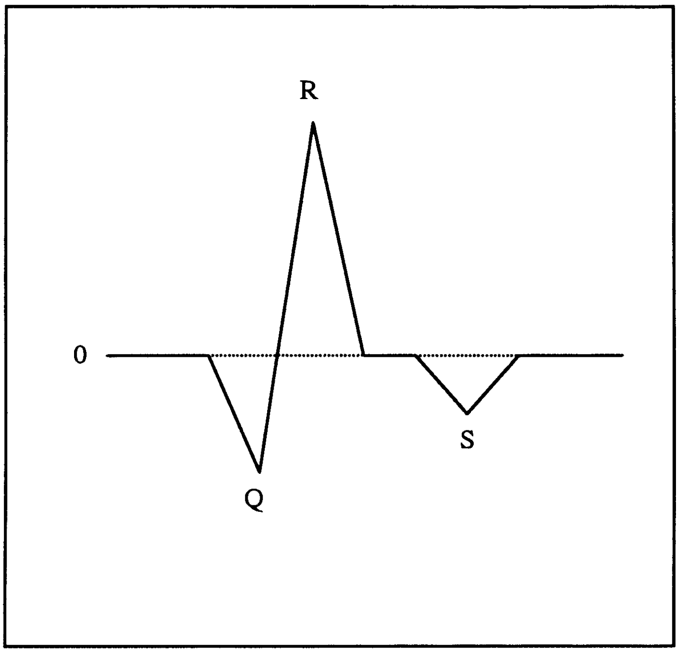


Figure 2.18 (c) Variations in the S-wave part of QRS complex showing apparent separation of S-wave from QR part of complex

$$F_s = \left(2 * \frac{(F_u - F_{\min})}{(F_{\max} - F_{\min})}\right) - 1 \qquad (2.1)$$

where F_u is the unscaled value, $F_{\min}$ the minimum value in the data set, and $F_{\max}$ the maximum value in the data set.

Another effective normalisation method is to calculate the mean and the standard deviation of the attributes first, and then divide the difference between each attribute value and the mean by the standard deviation:

$$F_s = \frac{F_u - \lambda}{\phi} \qquad (2.2)$$

where λ is the mean and ϕ the standard deviation of the data set , F_u is the unscaled value of the feature and F_s is the new scaled value of the feature.

2.5 Previous Work on ECG/Arrhythmia Classification

Several authors have looked at ECG arrhythmia classification using different means such as statistical methods, expert systems, and supervised neural networks.

Automated interpretation of ECGs began more than 35 years ago [Pibperger and Stallman 1962; Pordy et al., 1968]. Since that time there has been continuous development of expert systems for automated interpretation of ECGs. Automated interpretation of ECGs includes three basic approaches. The first is based on decision logic where a rule-based expert system is used to mimic the decision processes of a cardiologist. The second approach utilises multivariate statistical pattern recognition to solve a pattern recognition problem [Klingeman and Pipberger, 1967]. The third approaches employing neural networks [Rasiah and Attikiouzel, 1994] and machine learning [Oates et al., 1988] have also been developed.

Over the last few years automated interpretation of ECGs has been widely used as decision support for physicians, with the best interpretation programs performing almost as well as humans. Recent papers have shown that neural networks may be used to improve automated ECG interpretation for myocardial infarction.

Neural networks have been utilised with positive results in various medical diagnoses [Gallant, 1988; Frenster, 1990; Peng and Reggia, 1989]. In computerised ECG, the developed applications have concentrated mainly on beat and diagnostic classification [Gallant, 1988; Degani and Bortolan, 1990; Yeap et al, 1990]. According to Lippmann [Lippmann, 1989], recent interest in neural networks is directed towards practical research. This includes areas of study encompassing pattern recognition and artificial intelligence applications where real-time response is required. Both areas are relevant to ECG classification.

Pedrycz et al. [Pedrycz et al., 1991] used a combination of two pattern recognition techniques, cluster analysis and feed-forward back propagation neural networks, for the diagnostic classification of a 12-lead ECG. The principle of cluster analysis based on the Euclidean distance in parameter space was also applied to the original learning set. The classification accuracy results varied between 51.9% and 84.0% for classifying 7 classes of ECG abnormality.

Silipo and Bortolan [Silipo and Bortolan, 1997] compared statistical methods and neural network architectures with supervised and unsupervised learning approaches in performing the automatic analysis of the diagnostic ECG, where seven beat types and 39 features were used. The classification results varied between 91.0% and 94.0% correct classification for all seven types, showing that a classifier based on neural networks can produce a performance at least comparable with those of traditional classifiers. As for the neural network architectures trained with unsupervised techniques, they produced a reasonable classification performance. Interestingly, two additional features used were the age and sex of the subjects. This information is not given in the MIT-BIH database.

A neural network based system, the C.Net 2000 ambulatory ECG monitor, was developed by Gamlyn et al. [Gamlyn et al., 1999]. This is a portable, battery-powered unit capable of analysing an ECG in real time. A panel of Kohonen networks is embedded in a 32-bit micro-controller. The system is able to detect variations in the heart rate and P-R interval, changes to the ST segment, 'ectopic' beats and certain arrhythmias. Features include 24-hour monitoring and printout of detailed reports. The product is now commercially available.

Hu et al. [Hu et al., 1997] used a patient-adaptable approach to classify ECG beats in the MIT-BIH arrhythmia database. They concentrated on four categories of ECG beats, namely, normal, ventricular premature beat, fusion of normal and ventricular beat and unclassifiable beat. They used a mixture of the Self Organising Feature Map (SOFM) and Learning Vector Quantisation (LVQ) algorithms to develop two expert programs, the global expert program capable of classifying ECG beats from the whole database and the local expert program, which is a patient-specific expert system. The classification accuracy varied between 62.2%-95.9% for different records. The main drawback of the method is the need to create a local expert program for each individual patient.

Edenbrandt et al. [Edenbrandt et al., 1992] used single output MLPs to classify seven different classes of ST-T segments found in the ECG. They used the ST slope and the positive and negative amplitudes of the T-wave as inputs to the MLP. They trained and tested ten MLPs with different configurations of hidden layers and neurons in the hidden layers. The average classification accuracy was between 90.0% and 94.4%.

Izeboudjen and Farah [Izeboudjen and Farah, 1998] proposed an arrhythmia classifier using two neural network classifiers based on the MLP model. The morphological classifier groups the P-waves and QRS complexes into normal or abnormal beats. The timing classifier takes as the input the output of the morphological classifier and the duration of the PP, PR and RR intervals (see Figure 2.19). An accuracy of 93.0% was reported in classifying 13 arrhythmia classes from 48 examples scanned from different ECG signals using a PC.

Dorffner et al. [Dorffner et al., 1994] compared the performance of neural networks with the performance of skilled cardiologists in classifying coronary artery disease during stress and exercise testing. He performed three experiments, two of which used recurrent networks, while the third one employed an MLP. This neural network approach produced results comparable to the diagnosis of experts. Only in some cases did the neural networks outperform the experts.

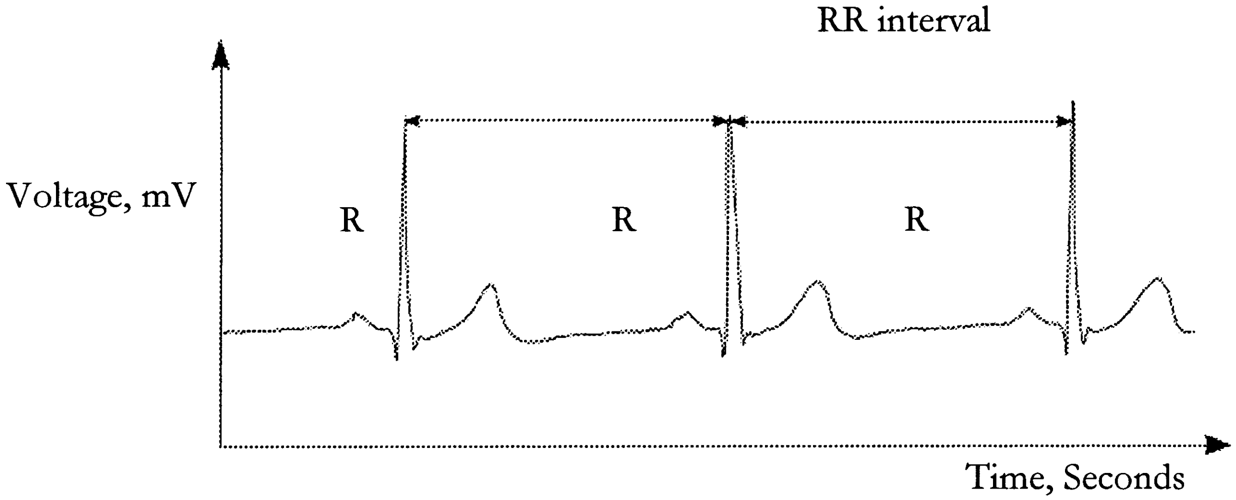


Figure 2.19: RR intervals from an ECG

Nugent et al. [Nugent at al., 1998] used single-output bi-group MLPs to detect the presence or absence of a specific ECG class. Three different feature selection techniques were adopted, namely, rule based, manual and statistical. The results of the bi-group neural networks were combined using orthogonal summation. The methodology was applied to recognise three classes, namely, normal, left ventricular hypertrophy and inferior myocardial infarction. On average, the classification accuracy was only 78.0%.

Biel et al. [Biel et al., 2001] suggested that the distinction between ECG signals of different people is sufficiently great to identify individuals using just one lead of an ECG.

Bortolan et al. [Bortolan et al., 1991] used a feed forward network with backpropagation to classify seven beat types using 39 features. Results of over 90.0% correct classification for all seven types were achieved. Interestingly, two features used were the age and sex of the subjects. Such information is not given in the MIT-BIH database. The same seven beat classes were investigated by Silipo et al. [Silipo et al., 1999] using a neural classifier with Radial Basis Function (RBF) pre-processing. Here again, correct beat type designation was consistently made for over 90.0 % for all classes.

The influence of various network parameters on multilayer neural network performance were researched by Edenbrandt et al. [Edenbrandt et al., 1992]. ECG ST-T segments were the basis of the study which found that increasing the number of input features did not necessarily improve classification. Similarly, increasing the number of neurons in the hidden layer beyond five gave no benefit. It was also reported that networks with two hidden layers showed only a very slight improvement over those with one hidden layer. Problems were encountered with training networks to recognise uncommon patterns, the best results being obtained, as expected, for those beats with the most examples in the training set.

Magleveras et al. [Magleveras et al., 1998] advised against using digital filtering of signals at the pre-processing stage to avoid corrupting the components of the ECG. However, others have done so in their work [Hamilton et al., 1986; Suzuki, 1995; Dokur et al., 1997].

Modular neural networks were applied to ECG classification [Kidwai, 2001]. These employed a more logical step-by-step approach by breaking the problem of classification down into stages rather than using a one-hit approach.

Suzuki [Suzuki, 1995] and Hamilton and Tompkins [Hamilton and Tompkins, 1986] researched methods of QRS complex detection. Their aim was reliably to break down a continuous ECG signal into individual beats. This is in contrast to supplying information from a database where signals have already been pre-divided into beats, such as the MIT-BIH database. Recognition of the QRS complex was proposed by Suzuki as the first step in the development of a real-time ECG analysis system. His self-organising neural network was capable of detecting R-waves in real time, in order to divide the ECG into cardiac cycles. An Adaptive Resonance Theory (ART) network then performed classification according to QRS complex features. Hamilton and Tompkins [Hamilton and Tompkins, 1986] claimed that their system carried out

QRS detection at 100 times the rate of the cardiac cycle, and gave a 99.8 % success rate for QRS identification.

Dokur et al. [Dokur et al., 1997] used a Kohonen neural network to detect four ECG waveforms: Normal beat (N), Premature ventricular contraction (V), Paced beat (P) and Left bundle branch block (L). The network was trained with data from the MIT-BIH arrhythmia database and gave a 90.0% classification accuracy.

2.6 Summary

This chapter has reviewed background material relevant to the work presented in this thesis. An overview of the functionality of the heart has been given. Special attention was paid to feature extraction, which constitutes the most important phase in ECG classification. An introduction to ECG pattern recognition has been given and a review of research on ECG classification carried out.

Chapter 3

Comparison of Different ECG Classification Techniques

3.1 Pattern Classification

Pattern classification encompasses a wide spectrum of significant information processing problems. Important technological applications range from speech recognition and hand-written character recognition, to fault detection in machinery and medical diagnosis.

The basic assumption in this field is that 'objects' can be characterised by a set of relevant measurements, called features. After measuring those particular features, an object can be classified from the measured feature values.

Pattern recognition techniques are used to classify input patterns into classes. Input patterns can be viewed as points in the multidimensional space defined by the input feature measurements. The purpose of a pattern classifier is to partition this multidimensional space into decision regions that indicate to which class any object belongs. As mentioned previously, the first step in pattern classification is the selection of features. These must be chosen carefully for each problem domain, and should contain the information required to distinguish between classes.

Good classification performance requires a recogniser that can make good use of those features with limited training data, memory, and computing power [Lippmann, 1989].

There are two phases to pattern classification, the training phase, where a recogniser is constructed, and the application or test phase. In the training phase, the training set is used to decide how the parameters ought to be weighted and combined in order to separate the various classes. In the application phase, the weights determined in the training set are applied to a set of unknown objects in order to determine what their classes are likely to be.

Classification is usually an easy problem when there are only two or three features. For example, with two features one can often simply construct a scatter-plot of the feature values and determine graphically how to divide those values into various regions where objects of the same class are grouped together. The classification problem becomes very hard when there are many features to consider.

Pattern classification can be in two ways, according to the data provided to the recogniser during the training phase. With the first method, known as unsupervised classification or clustering, the recogniser is required to cluster the data, by uncovering similarities between the data in order to group them. With the second method, known as supervised classification, the data comprise both input patterns and corresponding classes, and the recogniser is required to uncover a mapping between the two. In both cases, an unknown input pattern is subsequently assigned to one of the possible classes.

The automatic interpretation and classification of ECG signals begins with the collection of the ECG data from patients with various types of normal and abnormal heart beats. This data is then pre-processed by filtering to remove noise and by digitisation. The next step is feature extraction, where measurements are made of key parameters of the ECG, such as the height and duration of the P-waves, T-waves and QRS complexes. Finally, the training set is normalised, and then input to the classifier, as shown in Figure 3.1.

A number of machine learning techniques have been employed for pattern recognition. These enable computer programs automatically to improve their performance of some tasks through experience.

Machine learning algorithms have been classified according to Pham et al. [Pham et al., 2002] into inductive learning algorithms, artificial neural networks, genetic algorithms, and instance-based learning algorithms, as illustrated in Figure 3.2.

This chapter gives a comparison between three different types of machine learning: instance-based learning, neural network learning, and inductive learning. Genetic algorithms were not considered as this form of learning is generally too slow for real time applications.

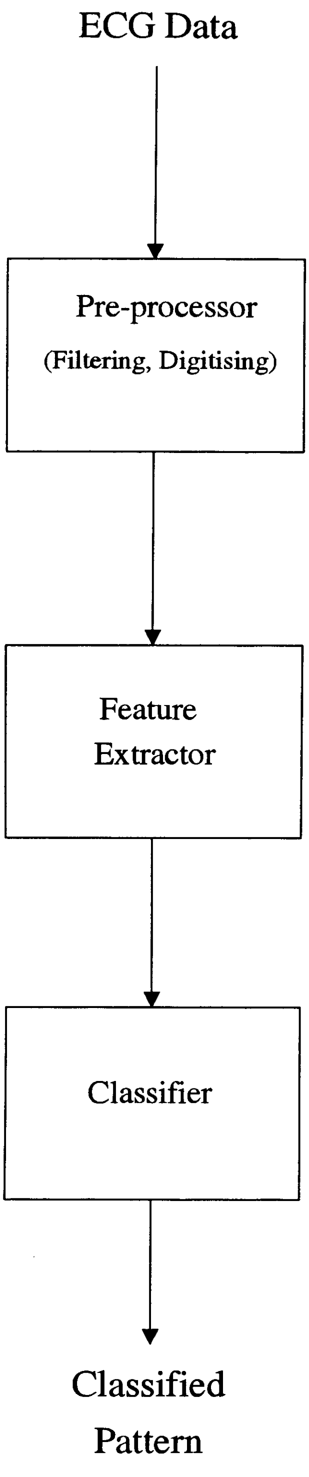


Figure 3.1: Components of an ECG pattern classification system

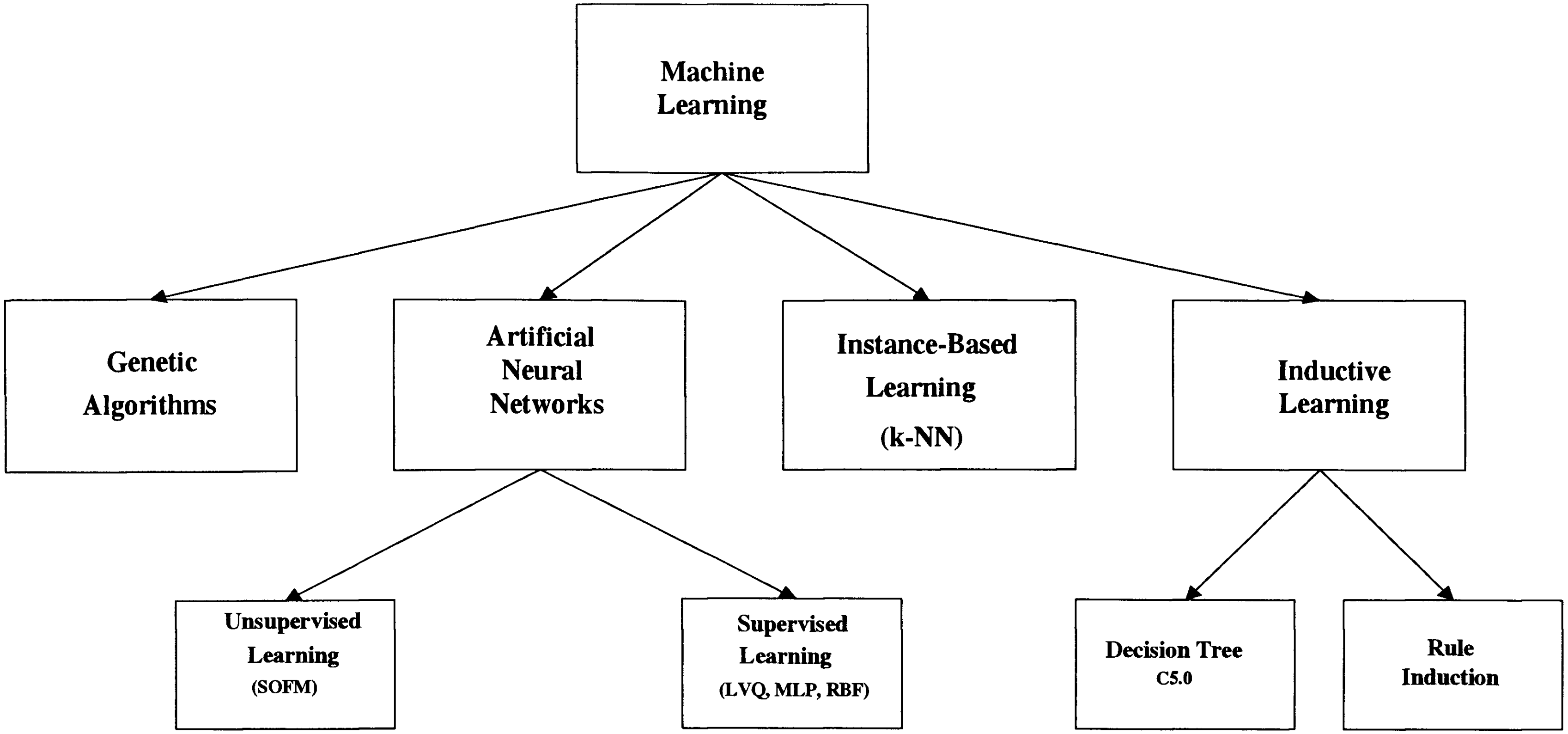

Machine Learning
Genetic Algorithms
Artificial Neural Networks
Instance-Based Learning (k-NN)
Inductive Learning
Unsupervised Learning (SOFM)
Supervised Learning (LVQ, MLP, RBF)
Decision Tree C5.0
Rule Induction

Figure 3.2: Machine learning techniques classification

KNN: k-nearest neighbour
LVQ: learning vector quantisation
MLP: multilayer perceptron
RBF: radial basis function
SOFM: self organising feature map

3.2 Instance-based Learning Classifiers

Instance-based learning consists of two stages. The first stage computes matching scores for each class according to how closely the input matches the exemplar pattern that best represents that class. The second stage selects the class with the maximum score.

A popular instance-based classifier is the k-Nearest Neighbour classifier (k-NN). This is a conventional nonparametric estimation classifier [Therrien, 1989], which leads directly to an approximation of class likelihood, minimises the probability of error, and provides good performance for optimal values of *k*. k-NN is a spontaneous method requiring no training, which classifies unlabeled examples based on their similarity to the examples in the training set.

The k-NN classifier, for a given unlabeled sample $X_u \in \mathrm{R}^n$, operates by finding the *k* "closest" labelled examples in the training data set and assigning X_u to the class that appears most frequently within the k-subset. In this case, the entire training set is stored in memory. A Euclidean distance metric is commonly employed [Mitchell, 1997]. If two or more 'nearest' classes exist, then the test sample is assigned to the class with the minimum average distance to it.

Figure 3.3 shows a simple example for classifying an unknown example X_u into one of three classes. In this case, a Euclidean distance metric is employed and $k = 5$. Of the five closest neighbours, four belong to w_1 and one belongs to w_3, so X_u is assigned to w_1, the predominant class.

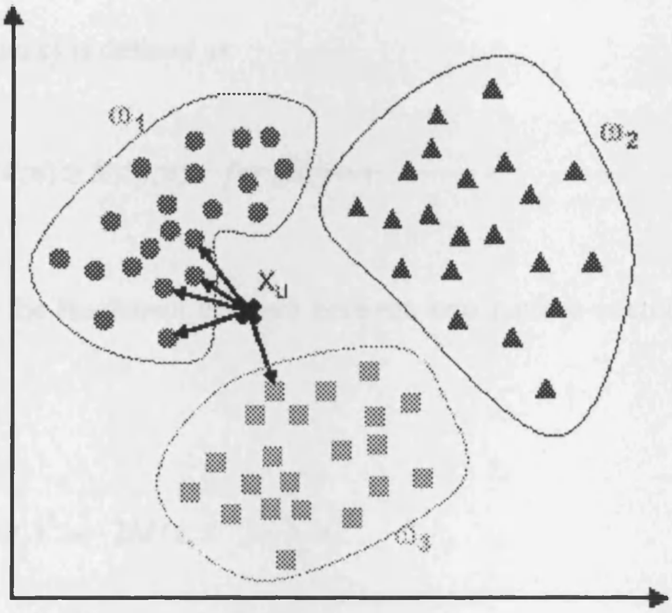


Figure 3.3: An example of k-NN classification

Let n training pattern vectors be denoted as: $x^{(i)}\ (l), i = 1,\ldots\ldots\ldots, n_l\ , l = 1,\ldots\ldots\ldots, C,$ where n_l is the number of training patterns from class $l, \sum_{l=1}^{C} n_l = n$, and C is the total number of categories. Let $K_l(k,n)$ be the number of patterns from class l among the k-NN of pattern x.

The nearest neighbours are computed from the n training patterns. According to the k-NN decision rule, class(x) is defined as:

$$\text{Class}\,(\text{x}) = j \;\; if \;\; K_j(k,n) \geq K_i(k,n) \quad for\ all\ j \neq i \tag{3.1}$$

Let $D(x,x^{(i)})$ denote the Euclidean distance between two pattern vectors, $x\ and\ x^{(i)}$, then:

$$D(x,x^{(i)}) = \sum_{j=1}^{d}(x_j - x_j^i)^2 = -2M(x,x^{(i)}) + \sum_{j=1}^{d} x_j^2 \tag{3.2}$$

$$M(x,x^{(i)}) = \sum_{j=1}^{d} x_j x_j^i - \frac{1}{2}\sum_{j=1}^{d}(x_j^i)^2 \tag{3.3}$$

where d is the number of features. $M(x,x^{(i)})$ is defined as the matching score between the test pattern x and the training pattern x^i. Finding the minimum Euclidean distance is therefore equivalent to finding the maximum matching score [Jain and Mao, 2000].

There are several advantages to the k-NN classifier: it is analytically traceable, new training examples can be easily added, it is simple to implement, and it utilises training data directly without the need to learn weights as parameters.

A severe drawback of the k-NN classifier, however, is that it can require a large amount of computation time to classify new examples because all computation takes place at the classification time rather than the training time as there is no training. The k-NN classifier is considered a 'lazy' learning algorithm [Aha, 1989], because it defers data processing until it receives a request to classify an unlabeled example. The k-NN classifier also needs large amounts of memory because all examples have to be stored.

3.3 Artificial Neural Networks

Neural networks (NNs) differ from conventional pattern recognition techniques in their ability to generalise from a limited number of samples to new situations which have not been encountered during the learning phase [Rumelhart et al., 1987]. In this way, they are able to solve problems that do not have an algorithmic solution, or where the available solution is too complex to be found. They have been successfully applied to many pattern classification problems [Lippman, 1989; Duda et al., 2001].

Computational models for neural networks are based on the human brain in consisting of inter-connected processing elements called 'neurons' [Lippmann, 1987; Hush and Horne, 1993; Haykin, 1994]. They may be classified [Pham and Liu, 1995] according to the nature of the connections between the neurons, i.e. their structure, and according to the means of adjusting the connections, i.e. their learning algorithm.

3.3.1 Neural Network Structure

Neural networks may be categorised according to their structure as either feedforward networks or recurrent networks.

i) Feedforward Networks

In feedforward networks, the neurons are grouped into layers. The neurons are connected between the layers with data flowing only in one direction, from the input layer to the output layer. The neurons in feedforward networks have a static memory, remembering only the current input at any given time. Some examples of feedforward networks are the Multilayer Perceptron (MLP) [Pandya and Macy, 1996], the Learning Vector Quantisation (LVQ) [Abramson and Wechsler, 2001; Engelbrecht, 2002] and the Radial Basis Function (RBF) network [Webb, 1999].

ii) Recurrent networks

In recurrent networks, the flow of signals can be both forwards and backwards. This is made possible by allowing the output of the neurons to be fed back, either to themselves or to neurons in preceding layers. In other words, a recurrent network has at least one feedback loop, i.e. cyclic connection, so that at least one of its neurons feeds its output back to the input of one or more preceding neurons. The behaviour of such networks may be extremely complex. The neurons may then have a dynamic memory, i.e. they not only remember the current inputs but also previous inputs and outputs. Examples of recurrent networks include the Hopfield Network [Negnevitsky, 2002], the Elman Network and the Jordan Network [Pham and Liu, 1995].

3.3.2 Learning in Neural Networks

NNs can be classified according to their mode of learning, namely supervised learning and unsupervised learning.

i) Supervised Learning

In supervised learning, as shown in Figure 3.4, the set of training inputs and their corresponding outputs must be presented to the network. The data set consisting of training input vectors and targets (desired outputs) is called the training set. The aim of training is to adjust the weight values in order that the difference between the real output and the target output is minimised. In this scheme, the system should be directed by an external signal (teacher) to achieve the desired performance.

The connections between the neurons are adjusted according to a supervised learning algorithm such as the Delta Rule, the Back Propagation (BP) algorithm and the Learning Vector Quantisation (LVQ) algorithm. The aim of these algorithms is to train the NN to achieve the desired outputs for the given inputs.

ii) Unsupervised Learning

Unsupervised learning does not require the desired output patterns to be presented to

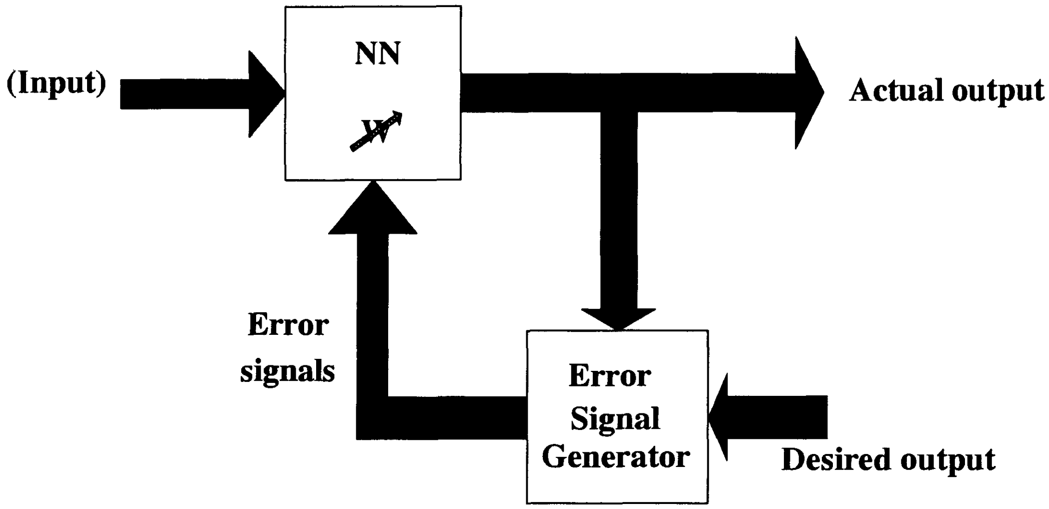


Figure 3.4: Simplified representation of supervised learning for NNs

the network, and the aim is to discover pattern features in the input with no external assistance.

A simplified representation of unsupervised learning is shown in Figure 3.5. No a *priori* knowledge is assumed to be available regarding an input's membership of a particular class. Rather, gradually detected characteristics together with the training history are used to assist the network in defining classes and possible boundaries between them.

The NN automatically clusters the input data into groups or classes. Such unsupervised learning systems are normally employed for pattern clustering. Well-known unsupervised learning systems include the Self Organising Feature Map (SOFM) [Kohonen, 1982] and the Adaptive Resonance Theory (ART) network [Carpenter et al., 1991].

For ECG classification, this research focused on the MLP, LVQ and RBF supervised learning networks, which had been found to have reliably strong performances [Lippmann, 1989; Kulkarni et al., 1998; Engelbrecht, 2002].

3.4 Multilayer Perceptron

The Multilayer perceptron is the most widely used architecture for feedforward NNs. Rosenblatt [Rosenblatt, 1958] first used the term perceptron for a single-layer network. In the late 1960s, Single-layered Perceptrons (SLPs) were proven mathematically inadequate for nonlinearly separable classification problems [Minsky

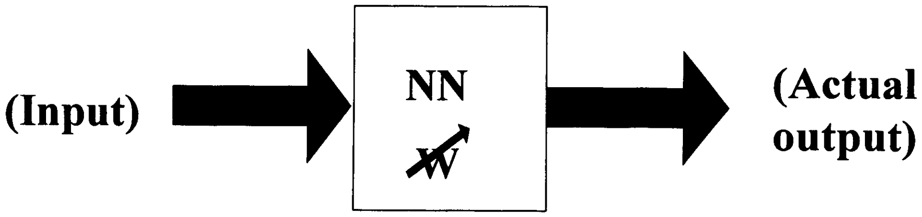


Figure 3.5: Simplified representation of unsupervised learning for NNs

and Papert, 1969]. However, in the mid 1980s, this problem was solved when the back propagation training algorithm was applied to a multilayer network.

This algorithm was first devised by Werbos (1974), but was popularised by Rumelhart et al. [Rumelhart et al., 1986].

A basic form of artificial neuron architecture is depicted in Figure 3.6. MLP feedforward networks consist of at least three layers of neurons: input, output and hidden layers, which are connected (an SLP has two layers counting the input layer). The layer which receives the input from the user is called the "Input Layer". The layer which provides information from the NN to the user is called the "Output Layer". Between the input and output layers lie one or more hidden layers. These hidden layers allow for the complex transformation of inputs to outputs. A fully connected MLP is shown in Figure 3.7.

The strengths of the connections between the neurons are referred to as "weights". Each neuron will produce an output after summing its weighted inputs. The output is then some function of the sum. For the neuron illustrated in Figure 3.6 the output y_j is calculated using Equation 3.4.

$$\mathrm{y_j} = \mathrm{f}\left(\sum_{i=1}^{N} \mathrm{w_{ji} x_i}\right) \tag{3.4}$$

The activation function *f(.)* can take different forms. Some common forms of activation functions [Pham and Liu, 1995] are listed in Table 3.1.

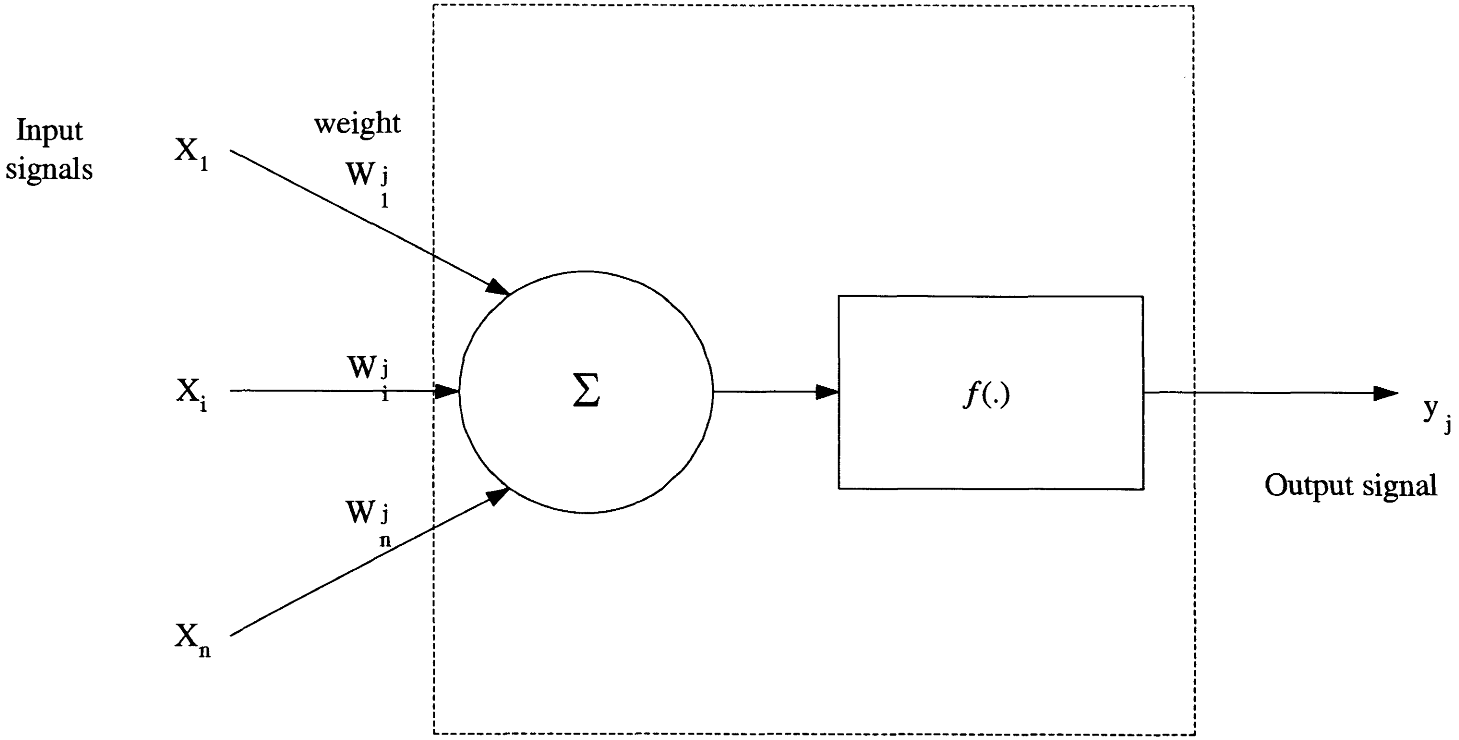


Figure 3.6: A neuron in a multilayer perceptron

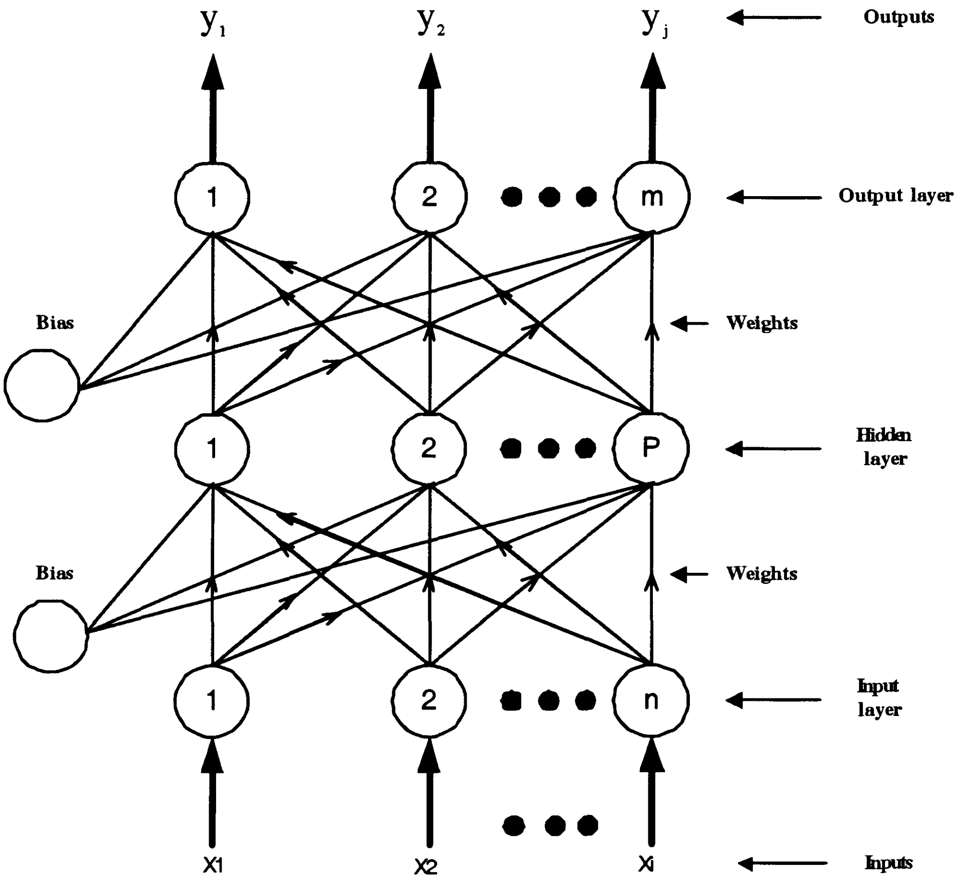


Figure 3.7: A fully connected multilayer perceptron (MLP)

Function Name	**Function Definition**
Linear	$f(\text{s}) = \text{s}$
Sigmoid	$f(\text{s}) = 1 / (1 + \exp(-\text{s}))$
Hyperbolic Tangent	$f(\text{s}) = (1 - \exp(-2\text{s})) / (1 + \exp(2\text{s}))$
Radial Basis Function	$f(\text{s}) = \exp(-\text{s}^2 / 2*\sigma^2)$

Table 3.1: Activation functions [Pham and Liu, 1995]

The most popular choices for *f(.)* are the hyperbolic tangent and sigmoid functions [Zurada, 1992].

The Back Propagation (BP) algorithm (Figure 3.8) is the most popular algorithm for training the MLP, and is a type of gradient descent algorithm. The BP algorithm compares the actual network outputs with the desired outputs corresponding to the presented input vectors. The differences between the two errors are then propagated back from the output layer to the preceding layers, and synaptic weights are modified to reduce these differences in the course of training. This process continues until the output error has met some predefined criterion. Learning iterations consist of two phases: the feedforward phase, which simply calculates the output value (s) of the NN, and the backward propagation phase, which propagates an error signal back from the output toward the input layer. Weights are adjusted as functions of the back propagated error signal.

The aim of the algorithm is to adjust the weights between the layers to make the output reach the desired value, i.e. reduce the error between the desired and actual output values. The adjustment in the weights is done using Equation 3.5:

$$\Delta w_{ji} = \eta \delta_j x_i \tag{3.5}$$

where w_{ji} represents a weight connection between neuron i and neuron j in the forward direction, x_i represents the output of neuron i, η is a gain factor called the 'learning rate' and δ_j is the error factor of neuron j that depends on whether neuron j is an output neuron or a hidden neuron.

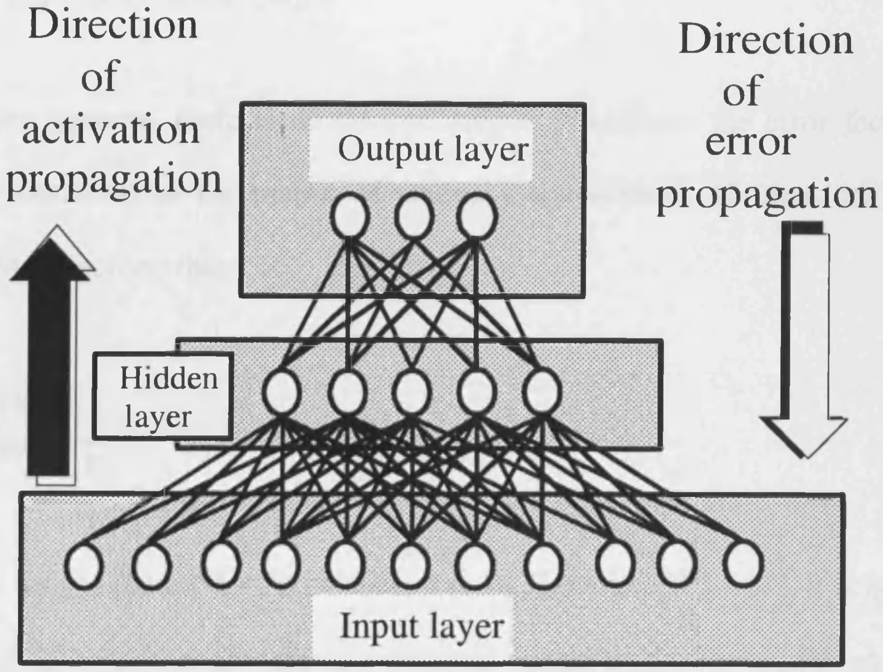

Direction of activation propagation
Output layer
Direction of error propagation
Hidden layer
Input layer

Figure 3.8: Back propagation multilayer perceptron

For an output neuron:

$$\delta_j = \left(\frac{\partial f}{\partial net_j} \right) (d_j - y_j) \tag{3.6}$$

where net_j is the total of all the weighted inputs to neuron j, d_j is the desired output of neuron j and y_j is the actual output.

For hidden neurons, there is no desired output. Therefore, the error factors of the neurons connected to the output of neuron j are utilised. If there are q neurons connected to neuron j then:

$$\delta_j = \left(\frac{\partial f}{\partial net_j} \right) \sum_q w_{qj} \delta_q \tag{3.7}$$

After the weight change Δw_{ji} is calculated using Equations 3.5 - 3.7, it is added to the existing weight $w_{ji}(k)$ to give the new adjusted weight $w_{ji}(k+1) = w_{ji}(k) + \Delta w_{ji.}$

Back propagation works by first initialising the weights to random values. The output of the neurons is calculated using Equation 3.6. If it is not the desired output, then the weights are changed as described in Equations 3.6 – 3.7. If after the weight change the output still has not reached the desired value, further weight adjustments are made. This process continues until the output becomes close to the desired value for each corresponding training input. One cycle of applying all the trainng data and changing the corresponding weights is refered to as an 'iteration' or 'epoch'.

The process of making the output closer to the desired value is called 'convergence'. The back propagation algorithm continues until a fixed number of epochs are reached or the difference between the desired and actual ouputs falls below a prescribed level. To speed up convergence, usually a momentum term "τ" is added:

$$\Delta w_{ji}(k+1) = \eta\delta_j x_i + \tau\,\Delta w_{ji}(k) \tag{3.8}$$

where $\Delta\, w_{ji}\,(k+1)$ and $\Delta\, w_{ji}\,(k)$ are the weight changes in the *(k+1)*th and *(k)*th epoch. Increasing τ increases convergence but this must be done with care as it may lead to instability.

The training parameters determine the weight changes. These include the initial weights, the learning rate and the momentum. The initial weights are usually randomised to small values in order to prevent premature convergence due to saturation of the activation functions. Small learning rates could not be set to ensure stable learning. However, weight changes from one epoch to the next will be smaller and consequently the rate of convergence will be smaller. A higher learning rate will increase the rate of convergence, but this may lead to oscillations in the weight values. The momentum term is designed to reduce the effect of these oscillations.

A selection of pattern recognition applications that utilise MLPs with the BP algorithm is given below:

- NETtalk: neural networks are shown to be capable of learning to pronounce English text [Sejnowski and Rosenberg, 1987].

- Hand written character recognition [Guyon, 1991].

- Detection of radar targets [Haykin and Deng, 1991].

- Diagnosis of electronic circuits broads used in digital telephone exchanges [Totton and Limb, 1991].

- Diagnosis of heart attacks [Harrison et al., 1991].

- Speech recognition [Renals et al., 1992].

- Control chart pattern recognition [Pham and Oztemel, 1993, 1994].

- Detection of high impedance arcing faults [Sultan et al., 1992].

- Identification of starches in manufacturing food products [Huang et al., 1993].

- Automatic classification of metaphase chromosomes [Errington and Graham, 1993].

- Classification of sonar targets [Svardstrom, 1993].

- Fingerprint classification [Pal and Mitra, 1996].

- Neural networks for classifying surface defects on automotive valve stem seals [Pham and Bayro-Corrochano, 1995].

- Identification of text independent Speaker using multiple classifiers [Chen et al., 1997].

- Classification of wood veneer defects [Pham and Sagiroglu, 2001].

- Airplanes shape recognition [Osowski and Nghia, 2002].

- Off-line handwritten word recognition [Liu and Gader, 2002].

- Classification of multispectral satellite images [Venkatesh and Raja, 2003].

- Text detection and recognition in images and video frames [Chen et al., 2004].

The design parameters determine the structure of the network. In an MLP, the design parameters are the number of hidden layers and the number of neurons in these layers. There is no 'fixed' rule to determine the number of hidden layers and the number of neurons that are needed to solve a particular problem. Usually one hidden layer will suffice. If the accuracy of the network is not satisfactory then the number of neurons in the hidden layer can be increased. If this does not improve the accuracy then a second hidden layer can be introduced. MLPs with three or more hidden layers are rare.

3.5 Radial Basis Function Neural Network

A radial basis function (RBF) neural network is a special kind of feedforward neural network. RBFs, however, store local information whereas conventional MLPs store global information. RBF networks belong to the group of kernel function networks that utilise simple kernel functions distributed in different neighbourhoods of the input space, whose responses are essentially local in nature. A RBF network is also capable of learning on-line, i.e. while it is still being employed for classification, unlike most other neural networks.

The architecture of a RBF network consists of three layers: an input layer, a hidden layer composed of J basis functions and an output layer composed of K neurons with linear activation functions as shown in Figure 3.9. A single-hidden-layer network has an advantage in terms of computing speed compared to multiple hidden layer networks. Each hidden node in an RBF network represents one of the kernel functions and corresponds to an individual point in the input space (called a centre), denoted as φ_j (for node j). The hidden nodes first calculate the Euclidean distances $\|x_p - \varphi_j\|$ between the input vector x_p for pattern p and the centres of the nodes. They then calculate an output signal from the node using the appropriate kernel functions and distance values. The output nodes simply compute a weighted summation of the hidden node outputs.

A kernel function is a local function whose range of effect is determined by its width. Its output is high when the input is close to the centre and decreases rapidly to near zero as the distance from the input to the centre increases. One of the most common functions used for o_{jp} is the Gaussian function and was the function adopted in this work. [Moody and Darken, 1989] have reported that Gaussian type functions have the desirable feature of allowing the hidden units to be locally tuned. Gaussian type

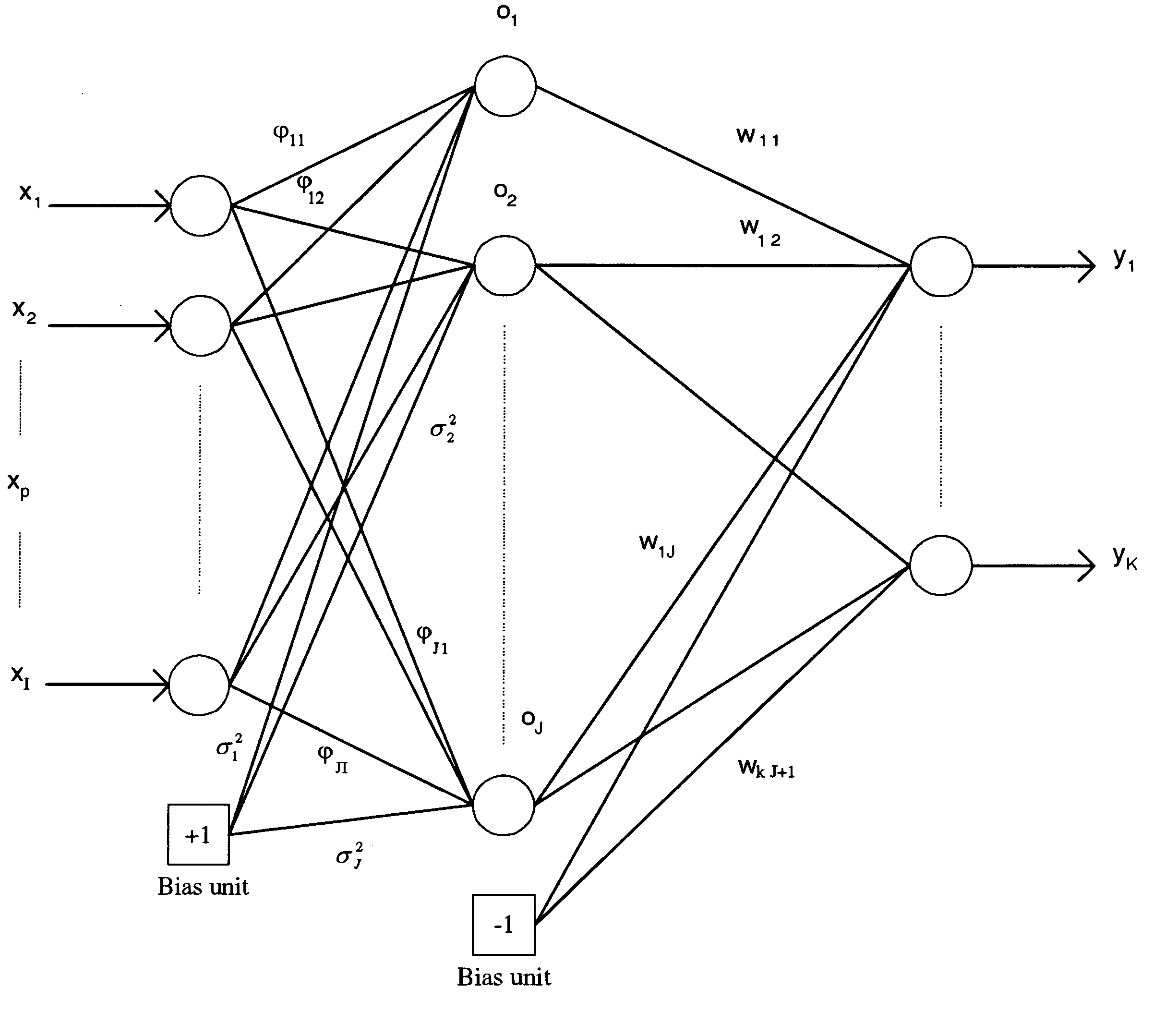


Figure 3.9: Radial basis function network

functions can be expressed as:

$$o_{jp} = \exp\left[-\frac{\left\|x_p - \varphi_j\right\|^2}{2\sigma_j^2}\right] \tag{3.9}$$

This function has a maximum value of 1 if $\left\|x_p - \varphi_j\right\|$is 0, and falls to 0 as $\left\|x_p - \varphi_j\right\|$ approaches infinity, where $\left\|x_p - \varphi_j\right\|$ is a distance measure, usually taken to be the Euclidean norm. Each basis function, o_{jp} , is centred at some point, φ_j in the input space.

Mathematically, the overall response function of such a network is given by:

$$y_{kp}(x) = \sum_{j=1}^{J} w_{kj} * o_{jp} \tag{3.10}$$

where y_{kp} is the k^{th} output of the network, J is the number of hidden units, $x_p \in R^n$ is an input vector, o_{jp} is the j^{th} radial basis function whose output is maximum at the centre and decreases rapidly to zero as the input's distance from the centre increases, and w_{kj} is the weight from the j^{th} hidden unit to the k^{th} output.

Training of a RBF network is achieved in two steps. In the first step, there is unsupervised learning of the weights φ_{ji} between the input and hidden layers using the equation developed by Kohonen [Kohonen, 1989]:

$$\Delta w_{ki} = \begin{cases} \eta(t)[x_{ip} - w_{ki}(t-i)] & if \quad y_{kp} = t_{kp} \\ -\eta(t)[x_{ip} - w_{ki}(t-i)] & if \quad y_{kp} \neq t_{kp} \end{cases} \tag{3.11}$$

where Δw_{ki} is the weight updates for the winning output unit y_k , $\eta(t)$ is a decaying learning rate and t_{kp} is the target for k^{th} output unit for pattern p.

In the next step there is supervised training of the w_{kj} weights between the hidden and output layers using Gaussian density function (Equation 3.9).

The RBF algorithm works as follows [Engelbrecht, 2002]:

1. (a) Initialise all φ_{ji} weights to the average value of all the inputs in the training set.

(b) Initialise all variances σ_j^2 to the variance of all the values over the training set.

(c) Initialise all w_{kj} weights to small random values.

2. Learn the centroids φ_j using Equation 3.10.

After each epoch, set the variance for all the winning o_j's mean weight vector φ_j to the input patterns for which o_j was selected as the winner.

3. Learn the hidden-to-output weights w_{kj} using the adjustment Equation:

$$\Delta w_{kj}(t) = \eta \sum_{k=1}^{k} (t_{kp} - y_{kp})\, o_{jp} \tag{3.12}$$

where t is the time step.

4. Stop training when the Gaussian density converges.

The activation function of values of the hidden and output units can be used to compute the degree P_{kp} to which a pattern p belongs to each class k:

$$P_{kp} = \frac{y_{kp}}{\sum_{j=1}^{J} o_{jp}} \tag{3.13}$$

Research has been conducted using RBF networks in many areas such as:

- Pattern recognition [Moody and Darken, 1989].
- System identification [Nie and Linkens, 1995].
- Control and signal processing [McLoone and Irwin, 1998].
- Classification of microcalcifications in digital mammograms [Tsujii et al., 1999].
- Learning of view-invariant pattern recogniser with temporal context [Inoue and Urahama, 2000].
- Off-line handwritten word recognition [Liu and Gader, 2002].

- Supervised image classification [Foody, 2004].
- Fault diagnosis of rotating machinery [Yang et al., 2004].
- Classification of gear faults [Wuxing et al., 2004].

3.6 Inductive Learning

The inductive learning process involves the operation of generalisation, correcting and refining knowledge representation [Michalski, 1990]. Induction is a method of moving from the particular to the general, from specific examples to general rules [Quinlan, 1989]. The general task of induction is to develop classification decision trees or rules that can determine the class of any object from the values of its attributes [Quinlan, 1986].

Inductive learning techniques can be divided into two main categories, namely, decision tree induction and rule induction.

A decision tree consists of internal nodes and leaf nodes. Each internal node represents a test. For a discrete attribute A with n possible values $v_{A_1}, v_{A_2}, \ldots v_{A_n}$, there are normally n different branches from an internal node. For a continuous attribute A, a binary test is often carried out, and a branch $A \leq v$ is created, with a second branch corresponding to $A > v$, where v is a threshold in the domain of A. Each leaf node has an associated class to which an instance can be assigned if it satisfies the conjunction of all tests along the path starting from the root and ending with this leaf node. When classifying a new instance, a path is identified based on values of the attributes of the instance until a leaf node is reached. The label of this leaf node is taken to be the predicted class of this new instance.

A simple example of how to decide whether to buy a second hand car is shown in Figure 3.10. Decision trees such as the one in Figure 3.10 can be converted into a collection of "IF-THEN" rules (a "rule set"), which may show the information in a more modular form. The rule set representation is useful in order to show particular groups of items related to one conclusion. For example, the following rules give a "profile" for a group of cars worth buying:

First Rule:

If mot = 'yes' and mileage = 'low' THEN 'BUY'.

Second Rule:

IF mot = 'yes' and mileage = 'high' and age = 'recent' THEN 'BUY'.

Third Rule:

IF mot = 'yes' and mileage = 'high' and age = 'old' THEN 'Do NOT BUY'.

Fourth Rule:

IF mot = 'no' THEN 'Do NOT BUY'.

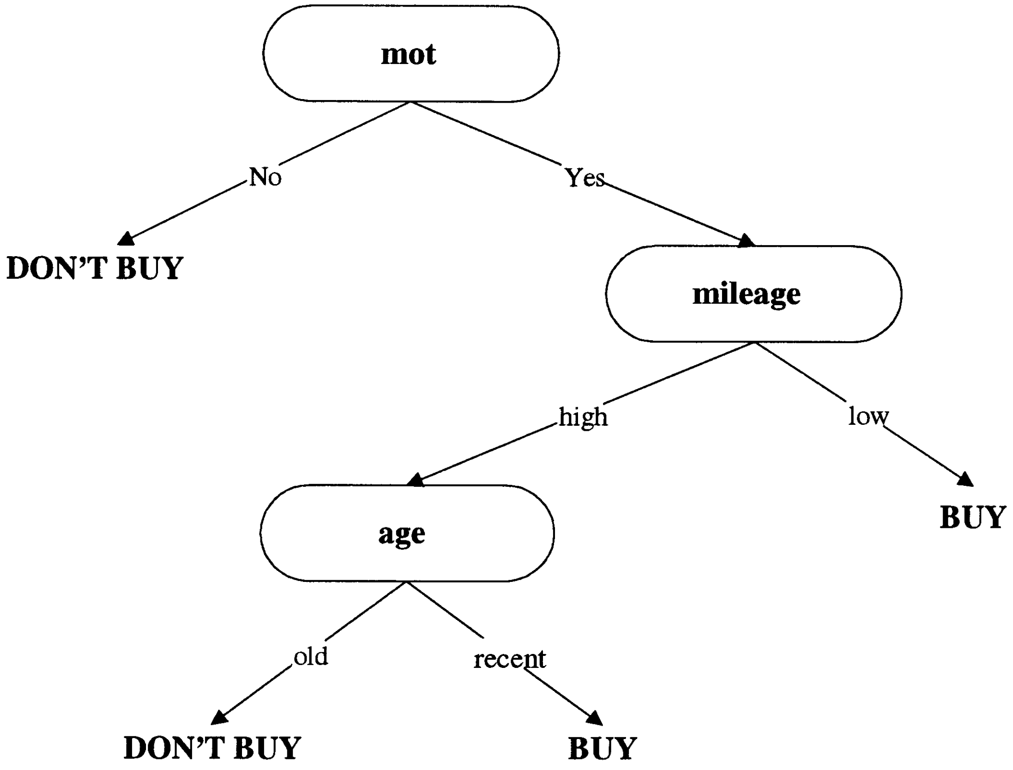


Figure 3.10: A Simple Decision Tree Example

In contrast with decision tree learning, rule induction directly generates IF-THEN rules. Each rule can be represented in the following form: $A_1 \wedge A_2 \wedge ... \wedge A_n \rightarrow C_i$, where the antecedent consists of a conjunction of predicates A_i. Each predicate takes the form $[A_i = v_i]$ or $[v_{j_1} < A_i \leq v_{j_2}]$ depending on the property of the attribute A_i. If A_i is a discrete attribute, v_i is a valid discrete value that A_i can take. If A_i is a continuous attribute, v_{j_1} and v_{j2} are two thresholds in the domain of attribute A_i. The consequent is the class to which instances satisfying the antecedent can be labelled. Since it is possible for some instances to be covered by more than one rule after the rule set is generated, when classifying a new instance, some conflict resolution approach must be employed.

Several algorithms for learning decision trees have been proposed. Among them, ID3 and its descendants C4.5, C5.0 [Quinlan, 1986; ISL, 1998] have achieved notable successes. This learning system is categorised as "divide-and-conquer" inductive system [Quinlan, 1986]. Here, the inductive process of C4.5, a top-down decision tree learning algorithm, is described. The tree is constructed in a recursive fashion.

First, the root node test is decided by considering all possible tests. The one that maximises the "information gain ratio" [Quinlan, 1993] is selected. Second, the whole data set is split into several subsets, each one corresponding to a test outcome. This tree building strategy is applied recursively to each subset until subsets obtained consist only of instances belonging to the same class. The gain ratio used in C4.5 requires definition of the entropy of a labelled data set S with k classes. Let k classes be C_1, C_2...C_k and let $P(C_i, S)$ be the proportion of instances in S which are in class C_i. Then the entropy of S is defined as:

$$Ent(S) = -\sum_{i=1}^{k} P(C_i, S)\log_2 P(C_i, S) \tag{3.14}$$

Let a test T with n outcomes partition the data set S into S_1, S_2,...S_n. The total entropy of the partitioned data set is defined as the weighted sum of the entropies of the subset.

$$Ent(S,T) = \sum_{i=1}^{n} \frac{|S_i|}{|S|} Ent(S_i) \tag{3.15}$$

To reduce the bias of the gain criterion introduced in ID3 towards attributes with many values, the split entropy as defined below is employed:

$$Ent(T) = -\sum_{i=1}^{n} \frac{|S_i|}{|S|} \cdot \log_2\left(\frac{|S_i|}{|S|}\right) \tag{3.16}$$

The gain ratio is then given by:

$$GainRatio(T) = \frac{Ent(S) - Ent(S,T)}{Ent(T)} \tag{3.17}$$

In Equation 3.17, the numerator is the "gain" in entropy as a result of partitioning the data set into mutually exclusive subsets based on test T. The denominator can be regarded as the cost of selecting a given attribute as a test. The gain ratio computation for a discrete attribute test is relatively straightforward. For continuous attributes, the *n* values appearing in the subset associated with an internal node are sorted. Then, all *n-1* possible splits on this continuous attribute are examined. The one that maximises the gain ratio criterion is selected as a threshold. After the decision tree is constructed, error-based post-pruning [Utgoff et al., 1997] is applied to prevent it from over-fitting the data. Pruning is done by examining each subtree and replacing it with one of its branches or leaf nodes if such a replacement does not degrade the accuracy of the subtree.

C5.0 is an improved version of C4.5 [Quinlan, 1993]. Like C4.5, C5.0 also allows automated extraction of production rules from the decision trees that it generates.

Improvements in C5.0 include the possibility of discrete and ordered attributes and the speed and quality of rule generation.

Figure 3.11 gives an example of a decision tree extracted by C5.0 [ISL., 1998].

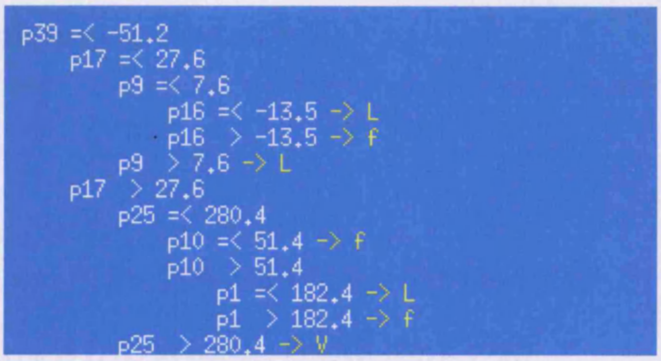
The decision tree can be converted into the following seven rules:

First Rule:

IF P39 =< -51.2 and P17 =< 27.6 and P9 =< 7.6 and P16 =< -13.5 THEN the class type is L (Left bundle branch block beat).

Second Rule:

IF P39 =< -51.2 and P17 =< 27.6 and P9 =< 7.6 and P16 > -13.5 THEN the class type is f (Fusion of paced and normal beat).



```
p39 =< -51.2
    p17 =< 27.6
        p9 =< 7.6
            p16 =< -13.5 -> L
            p16  > -13.5 -> f
        p9  > 7.6 -> L
    p17  > 27.6
        p25 =< 280.4
            p10 =< 51.4 -> f
            p10  > 51.4
                p1 =< 182.4 -> L
                p1  > 182.4 -> f
        p25  > 280.4 -> V
```

Figure 3.11 Example of a decision tree extracted by C5.0

Third Rule:

IF P39 =< -51.2 and P17 =< 27.6 and P9 > 7.6 THEN the class type is L (Left bundle branch block beat).

Fourth Rule:

If P39 =< –51.2 and P17 > 27.6 and P25 =< 280.4 and P10 =< 51.4 THEN the class type is f (Fusion of paced and normal beat).

Fifth Rule:

If P39 = < –51.2 and P17 > 27.6 and P25 =< 280.4 and P10> 51.4 and P1 =< 182.4 THEN the class type is L (Left bundle branch block beat).

Sixth Rule:

If P39 =< –51.2 and P17 > 27.6 and P25 = <280.4 and P10 > 51.4 and P1 >182.4 THEN the class type is f (Fusion of paced and normal beat).

Seventh Rule:

If P39 =< –51.2 and P17 > 27.6 and P25 > 280.4 THEN the class type is V (premature ventricular contraction).

3.7 Simulation Results and Comparisons

This section compares the performances of four classification techniques (C5.0, MLP, RBF and k-NN) applied to four different ECG data sets.

The first and the second data sets utilise 18 and 11 features respectively. The details of these features were explained in chapter 2. The third and the fourth data sets reduce the dimension of an ECG beat to 33 and 64 respectively without any feature extraction.

The C5.0 and MLP implementations in the Clementine data mining software [ISL., 1998] were employed in this work. The reason for using a commercially available program was that both techniques were known techniques and it would not have been useful expending efforts on developing software implementing them specially for this research.

Similarly, commercially available hardware implementations of the RBF network and the k-NN algorithm were adopted. The system employed was a Zero Instruction Set Computer (ZISC) board developed by IBM [Costa et al., 1996]. This is a CMOS PC-compatible card that can be plugged into an ISA or a PCI bus slot. The card can accept up to 64 8-bits input and can classify vectors belonging to up to 16382 categories. RBF networks with up to 576 neurons in the hidden layer can be realised.

Ten ECG classes were used. Table 3.2 shows the numbers of examples in each class in the training and test sets. Appendix A lists the examples taken from each record in the MIT database. The learning rate for both MLP and RBF neural network was 0.1.

The classification accuracy was calculated using the following equation:

$$\text{Accuracy}\,(\%) = \frac{\text{Nunmers of test patterns correctly classified}}{\text{Total numbers of pattern tested}} * 100 \qquad (3.18)$$

Table 3.3 gives a summary of the results obtained with the different techniques. For ease of visualisation, the results are also plotted in Figure 3.12. It was found that:

- C5.0 gave the best classification accuracy when compared to the MLP, RBF and the k-NN classifiers.

- The results for the MLP classifier were better than those for the RBF network.

- The k-NN classifier was the poorest classifier of all.

- Using the 18 features provided the highest classification accuracy.

- The 33 re-sampled data set gave better classification accuracy when compared with the 64 re-sampled data set.

- Feature extraction gave better classification accuracies when compared with simple re-sampling.

Class	Training Set	Test Set
N	600	200
V	600	200
P	600	200
L	600	200
R	600	200
A	31	16
f	14	6
a	75	38
E	68	35
j	64	32
Total	3252	1127

Table 3.2: Numbers of training and test examples taken from the MIT database

Classifier	Features Extracted		Re-sampling only	
	Data set 1	Data set 2	Data set 3	Data set 4
	11	18	33	64
C5.0	85.96%	88.36%	84.98%	84.92%
MLP	82.07%	85.86%	81.25%	80.50%
RBF	75.65%	79.15%	78.69%	77.90%
k-NN	72.52%	74.25%	73.19%	73.00%

Table 3.3 Summary of the classification accuracies obtained with different classification techniques

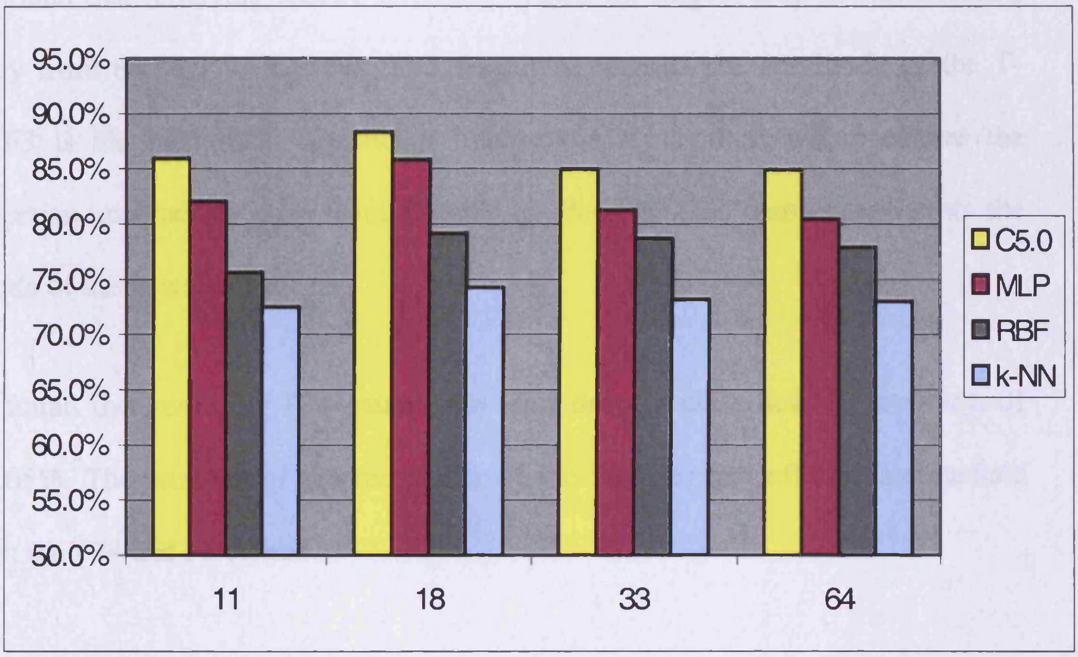


Figure 3.12: Graphical representation of the data in table 3.4

As C5.0 gave the best classifier, it was decided to employ it in further experimentation with the data. An experiment using C5.0 was carried out using 18 features data set by removing one feature at a time to study the effect on the classification accuracy. Table 3.4 and Figure 3.13 showed the classification results obtained.

It was found that removing feature five (F5) caused the largest drop in classification accuracy from 88.36% to 82.91%. This feature represents the amplitude of the T-wave. F3 is the next most significant feature the removal of which caused the classification accuracy to drop from 88.36% to 85.64%. This feature represents the amplitude of the R-wave.

It was found that removing F14 caused the least drop in classification accuracy, of only 0.05%. The removal of the remainder of the features had effects intermediate between the F14 and F5 effects.

Removed feature	Classification accuracy
F1	86.64%
F2	87.64%
F3	88.64%
F4	87.36%
F5	82.91%
F6	86.69%
F7	87.70%
F8	87.47%
F9	87.97%
F10	87.42%
F11	86.42%
F12	85.97%
F13	86.08%
F14	88.31%
F15	87.42%
F16	87.31%
F17	86.30%
F18	86.47%

Table 3.4: Effect of removing features on the classification results using C5.0

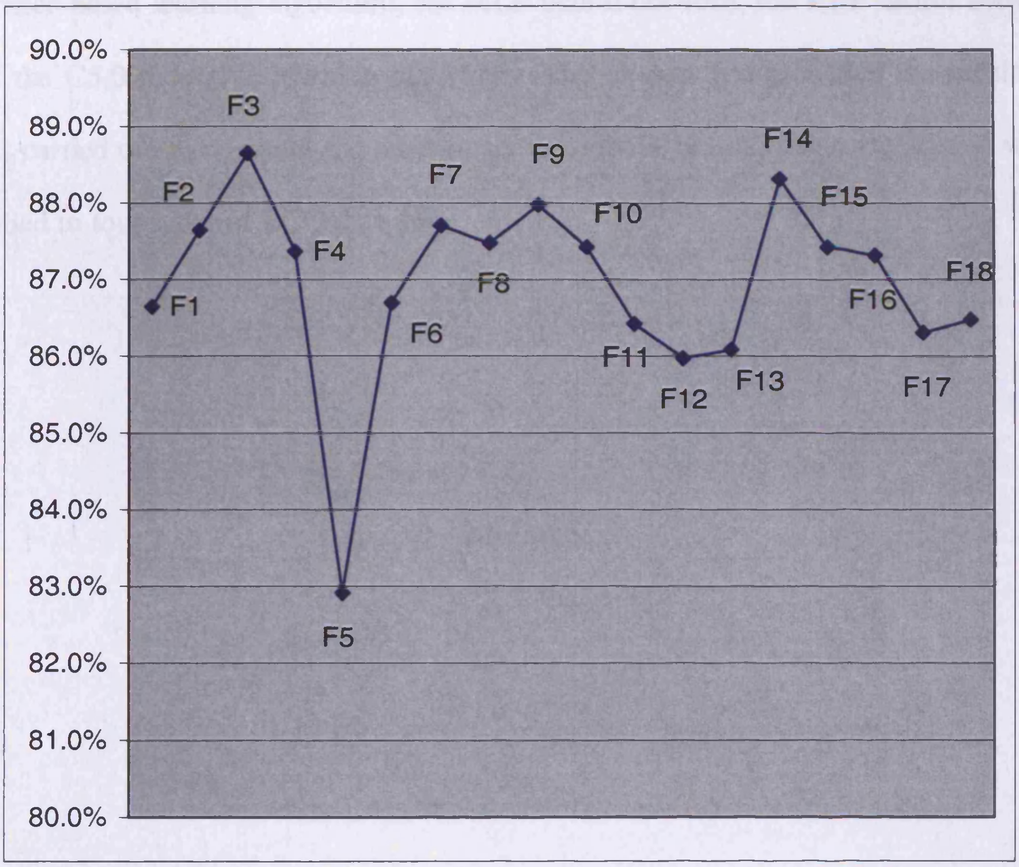


Figure 3.13: Effect of removing different features of the C5.0 on the classification accuracy

3.8 Summary

The chapter has described four different pattern classification techniques, the k-NN instance-based learning algorithm, the MLP neural network, the RBF neural network and the C5.0 inductive learning algorithm. The chapter has provided the results of tests carried out to compare the performances of those classification techniques when applied to four types of ECG data sets.

Chapter 4

Enhanced Learning Vector Quantisation Network using All Weights Updating

4.1 Preliminaries

Despite the good classification performance of C5.0, the best accuracy achieved was still below 90%. In ECG classification, a higher accuracy is desired in order not to cause false alarms or miss dangerous situations. This chapter presents an improved Learning Vector Quantisation (LVQ) neural network. The reason for the focus on LVQ network is their proven strong classification abilities [Baig et al., 2001].

The chapter describes a new modification to the LVQ neural network to yield the all weights updating LVQ (AWU-LVQ). AWU-LVQ employs a new method for updating weights that uses the Gaussian function in order to determine the amount of update for each weight in the network. This algorithm improves the classification accuracy of the standard LVQ and shortens the learning time.

This chapter gives details of the LVQ and the new AWU-LVQ algorithms, and their application to the recognition of ECG patterns from 5 different classes.

4.2 Learning Vector Quantisation (LVQ)

The LVQ neural network was developed by Kohonen [Kohonen, 1989], and has been successfully used for many classification problems. The learning method is supervised and based on "competitive" learning, in which neurons compete to have their weights updated. Figure 4.1 shows the LVQ network architecture, which consists of three layers of neurons: an input buffer layer, a hidden layer and an output layer. The network is fully connected between the input and hidden layers and partially connected between the hidden and output layers, with each output neuron linked to a different cluster of hidden neurons.

The weights of the connections between the hidden and output neurons are fixed at 1. The weights of the input to hidden neuron connections form the components of "reference" vectors, with one reference vector assigned to each hidden neuron. When an input vector is supplied to the network for recognition, the hidden neuron whose reference vector is closest in terms of Euclidean distance to the input vector is said to win the competition against all the other hidden neurons to have its output set to "1". All other hidden neurons are forced to produce a "0". The output neuron connected to the cluster of hidden neurons that contains the winning neuron also emits a "1" and all other output neurons, a "0". The output neuron that produces a "1" gives the class of the input vector, each output neuron being dedicated to a different class.

In the learning stage, the neurons in the hidden layer again compete amongst themselves in order to find the winning neuron whose weight vector is most similar to

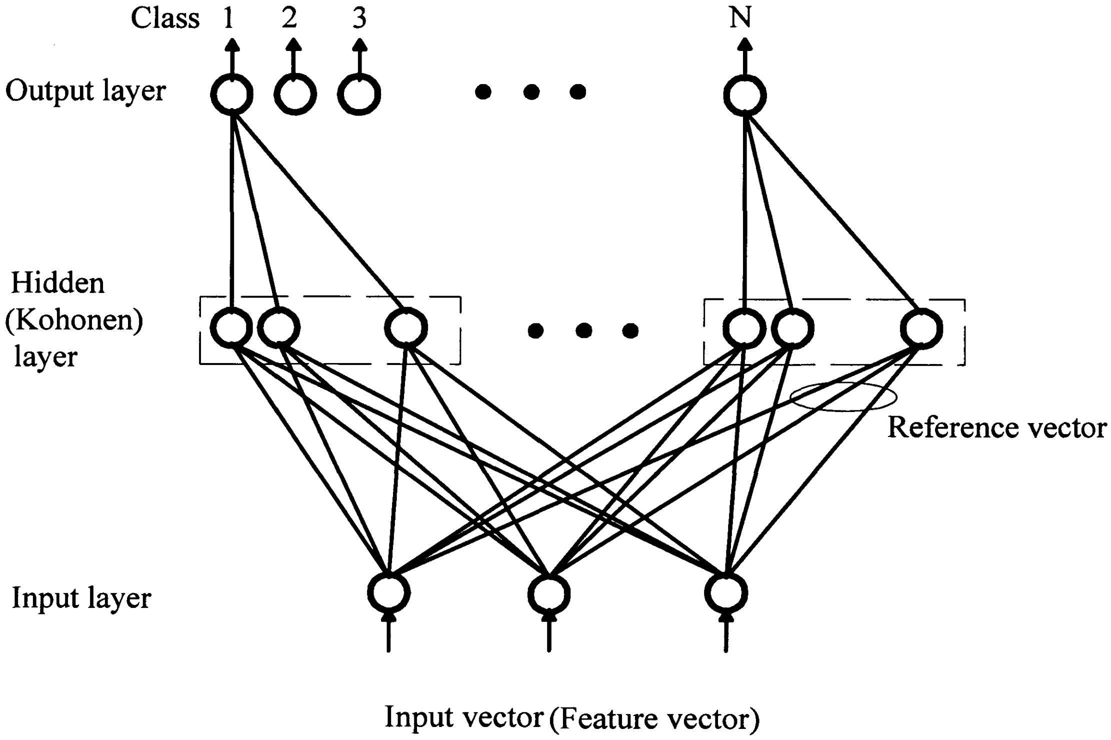


Figure 4.1: Learning vector quantisation network

the input vector, the winning neuron being that one with weight vector closest to the input vector [Kohonen, 1990]. The winning neuron gives the class of the input vector as in the recognition phase. Only the winning neuron will modify its weights using a positive or negative reinforcement learning formula, depending on whether the class indicated by the winning neuron is correct or not. If the winning neuron belongs to the same class as the input vector (the classification is not correct), it will be allowed to increase its weights, moving slightly closer to the input vector (positive reinforcement). On the contrary, if the class of the winning neuron is different from the input vector class (the classification is not correct), it will be made to decrease its weights, moving slightly further from the input vector (negative reinforcement).

Weights are modified during the training of the network. Both the hidden neurons (also known as Kohonen neurons) and the output neurons have binary outputs.

The competition between the Kohonen neurons is based on the Euclidean distance between the weight vectors $w_i(t)$ and the input vector *x (t).* The calculation of the Euclidean distance d_i between *x (t)* and $w_i(t)$ is as follows:

$$d_i = \| w_i(t) - x(t) \| = \sqrt{\left(\sum_j \left(w_{ij}(t) - x_j(t) \right)^2 \right)} \tag{4.1}$$

where w_{ij} and x_j are the j^{th} components of w_i and x. As mentioned before, the neuron that has the minimum distance wins the competition and is allowed to change its connection weights, while the rest of the weights remain unchanged. The new weights are given by:

$$w_i(t+1) = w_i(t) + \alpha \; (x_i(t) - w_i(t)) \tag{4.2}$$

if the winning neuron is in the same category as the output, which means the weights will move closer to the input vector, and by:

$$w_i(t+1) = w_i(t) - \alpha \; (x_i(t) - w_i(t)) \tag{4.3}$$

if the winning neuron is in the wrong category, which means the weights will move further away from the input vector. In Equations 4.2 and 4.3, α is the learning rate, which decreases monotonically with the number of iterations. Usually, $0 < \alpha < 1$.

A selection of pattern recognition applications that unitise LVQ algorithm is given below:

- Object orientation detection [Morris et al., 1990].
- Protein classification [Merelo et al., 1991].
- Classification of liver tissues [Pan and Chen, 1992].
- Classification of seismic events [Jang et al., 1993].
- Classification of electroencephalographic EEG power spectra [Veselis et al., 1993].
- Inspection of coated steel samples using scattering angles as inputs [Olsson and Gruber, 1993].

- Analysis of multivariate biological data [Wilkins et al., 1994].
- Discrimination between various partial discharge pulse shapes [Mazroua et al., 1994].
- Combined image compression and classification [Oehler and Gray, 1995].
- Detection and location of gross errors in plant performance data [Aldrich and Vandeventer, 1995].
- Control chart pattern recognition [Pham and Oztemel, 1994].
- Defect detection in concrete [Shoukry et al., 1996].
- Automatic speech recognition [Cosi et al., 2000].
- Performance analysis and optimisation of shape recognition and classification [Nabhani and Shaw, 2002].
- Classification of heart sounds [Olmez and Dokur, 2003].
- Real-time fault detection and isolation in industrial machines [Marzi, 2004].

4.3 All Weights Updating-LVQ Technique

As describe above, in the original LVQ neural network, only one weight will be modified in each training phase. In later versions, such as LVQ2 [Kohonen, 1989] and

LVQ-X [Pham and Oztemel, 1994], two weights (that of the winning neuron "global winner", and the next weight which is in the correct category and nearest to the input vector in that category "local winner") will be modified.

The AWU-LVQ algorithm is based on updating all the weights at the same time in each iteration during the training stage. However, the amount of updating will vary according to how far a neuron is from the winner in the hidden layer. The weight of the winning neuron will receive the maximum amount of updating, while the other neurons will be updated to reducing degrees as their distances from the winner increase. Figure 4.2 shows the Gaussian function used to determine the amount of updating. The Gaussian function was adopted because it enables a smooth asymptotic reduction in the amount of weight update. If a neuron is in the same class as the input vector, this neuron will move toward the input vector and if it is not, it will move away from the input vector.

Let the input vectors be represented by $x_i\,(t)$ and the weights of the network by $w_i(t)$. If the neuron weight vector closest to the input vector is given the index c, the Gaussian function $h_i\,(t)$ is then used to adjust the amount of updating according to the following equation:

$$h_i(t) = e^{-\frac{\| d_i - d_k \|}{2\sigma^2}} \tag{4.4}$$

where the parameter σ defines the width of the Gaussian function which determines the degree of neuron excitation.

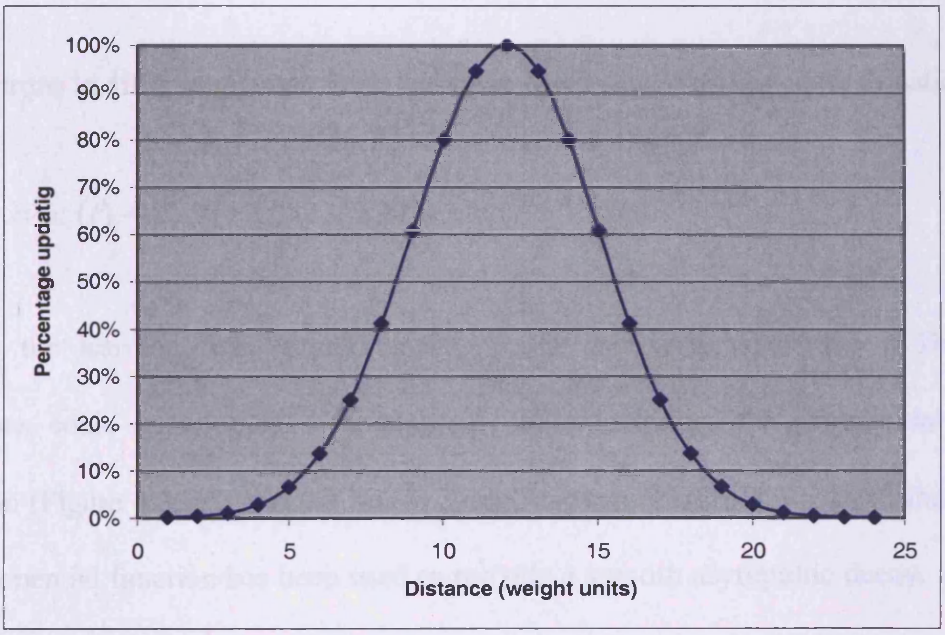


Figure 4.2: Gaussian weight updating function

d_k is Euclidean distance between the next closest neuron of the winning neuron and the input vector, d_i the Euclidean distance between the input vector x and the weight vector w_i

The weight update Equation for neurons in the same class as the input vector is:

$$w_i(t) = w_i(t) + \alpha(t)\ h_i(t)\,[x_i(t) - w_i(t+1)] \tag{4.5}$$

For neurons in different classes from the input vector, the weight update Equation is:

$$w_i(t) = w_i(t) - \beta\ \alpha(t)\ h_i(t)\,[x_i(t) - w_i(t+1)] \tag{4.6}$$

$\alpha(t)$ is the learning rate, which monotonically decreases with time *t*. Different functions could be adopted to implement $\alpha(t)$, including the exponential decay function (Figure 4.3 (a)) and the linear decay function (Figure 4.3 (b)). In this work, an exponential function has been used to provide a smooth asymptotic decay. $h_i(t)$ is the neighbourhood function. The β parameter is used to avoid negative corrections in supervised learning [Yang and Yang, 2002].

The AWU-LVQ algorithm works as follows:

1- Decide the network architecture (number of inputs, number of hidden neurons, and number of outputs). Randomly initialise the values of all the reference weight vectors.

2- Initialise the learning rate α and the width of the Gaussian function σ.

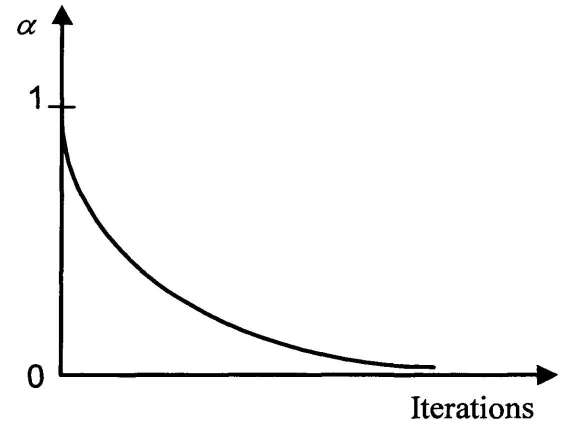


Figure 4.3 (a): Exponential decay function

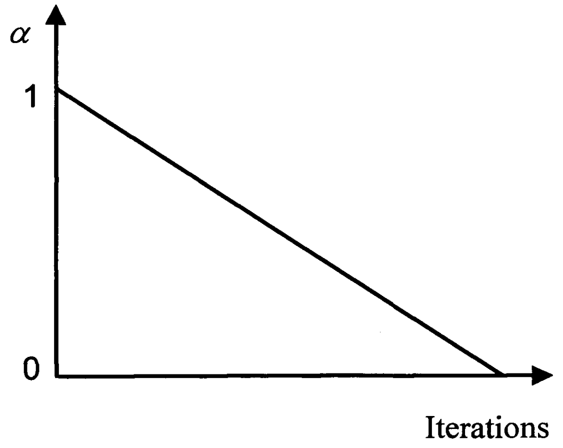


Figure 4.3 (b): Linear decay function

3- Present the input (training) vector to the network.

4- Find the winning neuron i by calculating the Euclidean distance between the input vector and all the network weights using Equation 4.1.

5- Calculate the distance function h_i for all the reference vectors using Equation 4.4.

6- Update all the weights of the neurons in the hidden layer. If a neuron is in the same class as the input, its weight vector will move toward the input vector according to the Equation 4.5, and if not, the weight vector will move away using the Equation 4.6.

7- Reduce the learning rate $\alpha(t)$ and go to step 3 with a new input vector. Repeat the procedure until all input vectors are correctly classified or a stopping criterion is met.

4.4 Experimental Results

The experimental work described in this chapter was performed using two neural networks, LVQ and AWU-LVQ, in order to compare these networks from the point view of classification accuracy and training time.

4.4.1 Network Configuration

The AWU-LVQ and LVQ neural networks were given the same architecture to facilitate their comparison. Data consisting of input vectors with 15 features as shown in Table (4.1) and from 5 classes were utilised. The networks had 15 neurons in the input layer, 25 neurons in the hidden layer and 5 neurons in the output layer. The 15 inputs represent the 15 features extracted from one cycle of the ECG signal. The output layer has 5 neurons, representing the five types of ECG arrhythmia classes (V, L, P, R, and N ECG class) chosen for this experimental work. The hidden layer consisted of 25 neurons, allowing five reference vectors per class. The weights were initialised randomly to values between -1 and 1.

The learning rate was set to 0.1 and was made to decrease exponentially with iteration time as already mentioned. The exponentially decay coefficient was 0.01. The standard deviation of the Gaussian distance function for the AWU-LVQ algorithm was 3. All the values used were empirically chosen.

4.4.2 Training and Test Sets

The training and test data sets employed consisted respectively of 400 and 150 patterns for each of the 5 classes.

Feature	Significance	Description
Signal period	Physiological	Time between successive R-waves.
Q peak height	Physiological	Amplitude of the Q-wave.
R peak height	Physiological	Amplitude of the R-wave.
S peak height	Physiological	Amplitude deflection of the Q-wave.
Q-wave duration	Physiological	Overall duration of the Q-wave.
R-wave duration	Physiological	Overall duration of the R-wave.
S-wave duration	Physiological	Overall duration of the S-wave.
QRS interval	Physiological	Overall duration of the QRS complex – from the onset of the Q-wave to the end of the S-wave. (Time taken for complete ventricular pumping action).
RS height	Physiological	Distance between the peak of the R-wave and depth of the S-wave.
R to S height ratio	Clinical	Ratio of R-wave height to S-wave depth (This indicates the orientation of the electrodes relative to the axis of the heart).
Q to QRS duration ratio	Physiological	Ratio of the durations of the Q-wave and QRS complex.
Mean	Statistical	Mean value of the electrical signal.
Standard deviation	Statistical	Standard deviation of the electrical signal from the baseline.
Skewness	Statistical	Statistical skewness of electrical signal.
Excess	Statistical	Statistical excess of electrical signal.

Table 4.1: Features of ECG signal selected for classification

The classification accuracy was again calculated using the following equation:

$$\text{Accuracy}\,(\%) = \frac{\text{Nunmers of test patterns correctly classified}}{\text{Total numbers of pattern tested}} * 100 \qquad (4.7)$$

4.4.3 AWU-LVQ and LVQ Results

Several experiments were carried out in order to select two AWU-LVQ algorithm parameters, namely, the number of training epochs and the width σ of the kernel of the Gaussian distance function.

Table 4.2 shows the influence of σ on the classification accuracy. It can be noted that the accuracy improves with increasing σ, up to a value of 3.5.

When σ is small only a few neurons near to the winning neuron will be updated by a significant amount, and the rest of the neurons will receive only small amounts of updating or even no updating, which explains why the classification accuracy is poor and highly variable. When σ increases, more neurons will be updated to different degrees, which yields improvement in the classification results (higher accuracy and lower variance).

The effect of epoch number on classification accuracy is presented in Table 4.3. This shows an increase in accuracy up to 2000 epochs and a decrease thereafter. This happened because of overtraining of the neural network which makes it lose its ability

	Gassian function kernel width						
Run	1	1.5	2	2.5	3	3.5	4
1	85.60%	89.60%	85.60%	85.20%	85.20%	85.60%	85.20%
2	20.00%	20.00%	89.60%	85.20%	85.60%	85.60%	85.60%
3	85.60%	85.60%	85.60%	85.20%	85.60%	85.20%	85.20%
4	90.80%	85.60%	89.60%	85.20%	85.20%	85.60%	85.60%
5	85.60%	20.00%	85.60%	85.20%	85.60%	85.60%	85.60%
6	20.00%	90.00%	58.00%	85.20%	85.60%	85.60%	85.20%
7	85.60%	89.60%	85.60%	85.20%	85.20%	85.60%	85.20%
8	90.00%	90.00%	85.60%	85.20%	85.20%	85.20%	85.20%
9	85.60%	85.60%	85.60%	85.60%	85.20%	85.20%	85.60%
10	89.60%	89.60%	85.60%	85.20%	85.20%	85.20%	85.60%
Average	73.84%	74.56%	83.64%	85.28%	85.36%	85.44%	85.40%
Variance	809.572	830.514	83.9271	0.028	0.0427	0.0427	0.0444

Table 4.2: Effect of kernel width on the classification accuracy

(learning rate α = 0.1 and epoch number = 200)

Epoch number	Classification accuracy
5	54.16%
10	73.60%
20	74.48%
50	79.84%
100	80.24%
200	83.64%
500	87.24%
1000	87.56%
2000	88.32%
2500	86.76%

Table 4.3: Effect of epoch number on the classification accuracy

(learning rate $\alpha = 0.1$ and kernel width $\sigma = 2$)

to generalise. In other words, the network simply memorises the training patterns and cannot recognise new test patterns. This phenomenon is similar to overtraining in MLP neural networks [Pham and Liu, 1995]. Table 4.4 presents more results concerning the effect of the number of epochs on the classification accuracy of the AWU-LVQ network. It can be seen that beyond 50 epochs there was a small drop in the classification accuracy and beyond 200 epochs there was no noticeable improvement in the stability of the classification results.

Table 4.5 shows the results for the LVQ algorithm with $\alpha = 0.1$ and different numbers of epochs.

Table 4.6 compares AWU-LVQ and LVQ. From this table the improvement of AWU-LVQ over the standard LVQ in both classification accuracy and training time is evident. The best classification accuracy using LVQ was 64.48% after 1000 epochs, while using AWU-LVQ the best classification accuracy was 85.44% after only 10 epochs.

	Epoch number									
Run	**1**	**2**	**5**	**10**	**20**	**50**	**100**	**200**	**500**	**1000**
1	38.00%	47.20%	82.00%	85.60%	85.60%	85.60%	85.60%	85.60%	85.60%	85.20%
2	20.00%	38.00%	85.20%	85.20%	85.60%	85.20%	85.60%	85.20%	85.20%	85.60%
3	38.00%	71.20%	85.20%	85.20%	85.20%	85.60%	85.60%	85.20%	85.20%	85.20%
4	20.00%	38.00%	85.20%	85.20%	85.20%	85.60%	85.20%	85.20%	85.20%	85.20%
5	20.00%	38.00%	85.20%	85.60%	85.20%	85.20%	85.60%	85.20%	85.20%	85.20%
6	32.80%	38.00%	85.20%	85.20%	85.60%	85.20%	85.20%	85.20%	85.20%	85.60%
7	38.00%	53.60%	85.20%	85.60%	85.60%	85.60%	85.20%	85.60%	85.60%	85.20%
8	38.00%	38.00%	85.20%	85.60%	85.20%	85.60%	85.20%	85.20%	85.20%	85.20%
9	38.00%	38.00%	77.20%	85.60%	85.60%	85.20%	85.20%	85.20%	85.20%	85.60%
10	20.00%	49.60%	85.20%	85.60%	85.60%	85.60%	85.60%	85.60%	85.60%	85.20%
Average	**30.80%**	**44.96%**	**84.08%**	**85.44%**	**85.44%**	**85.44%**	**85.40%**	**85.32%**	**85.32%**	**85.32%**
Variance	80.784	120.0427	6.8551	0.0427	0.0427	0.0427	0.0444	0.0373	0.0373	0.0373

Table 4.4: Effect of number of epochs on AWU-LVQ classification accuracy

(learning rate $\alpha = 0.1$, kernel width $\sigma = 3$)

	Epoch number									
Run	1	2	5	10	20	50	100	200	500	1000
1	20.00%	20.00%	20.00%	36.40%	54.00%	64.00%	65.20%	64.00%	66.40%	66.40%
2	20.00%	20.00%	20.00%	36.40%	63.20%	64.00%	65.20%	67.60%	64.00%	64.00%
3	20.00%	20.00%	20.00%	36.40%	56.00%	64.00%	64.00%	65.20%	65.20%	65.20%
4	20.00%	20.00%	20.00%	36.40%	52.80%	64.40%	64.00%	64.00%	65.60%	65.60%
5	20.00%	20.00%	20.00%	36.40%	63.20%	64.00%	64.00%	64.00%	65.20%	65.20%
6	20.00%	20.00%	20.00%	36.40%	55.20%	64.00%	64.00%	64.00%	64.00%	64.00%
7	20.00%	20.00%	20.00%	36.40%	57.20%	64.80%	64.00%	64.00%	64.80%	64.80%
8	20.00%	20.00%	20.00%	36.40%	55.20%	65.20%	64.00%	67.60%	64.40%	66.40%
9	20.00%	20.00%	20.00%	36.80%	52.80%	65.20%	64.40%	64.00%	64.40%	66.40%
10	20.00%	20.00%	20.00%	36.80%	55.60%	64.00%	65.60%	64.00%	66.00%	66.80%
Average	**20.00%**	**20.00%**	**20.00%**	**36.48%**	**56.52%**	**64.36%**	**64.44%**	**64.84%**	**65.00%**	**65.48%**

Table 4.5: LVQ classification results (learning rate $\alpha = 0.1$)

Epochs	LVQ	AWU-LVQ
1	20.00%	30.80%
2	20.00%	44.96%
5	20.00%	84.08%
10	36.48%	85.44%
20	56.52%	85.44%
50	64.36%	85.44%
100	64.44%	85.40%
200	64.84%	85.32%
500	65.00%	85.32%
1000	65.48%	85.32%

Table 4.6: Comparison between AWU-LVQ and LVQ

4.5 Summary

This chapter has introduced a modification to the standard LVQ algorithm, the All Weights Updating LVQ (AWU-LVQ), which operates by simultaneously updating all network weights during training. Results of applying both AWU-LVQ and LVQ algorithms for ECG classification have been presented. These show marked improvements of AWU-LVQ over the standard LVQ although the classification accuracy of AWU-LVQ is still lower than those of C5.0 and other known classifiers.

Chapter 4

Enhanced Learning Vector Quantisation Network using All Weights Updating

4.1 Preliminaries

Despite the good classification performance of C5.0, the best accuracy achieved was still below 90%. In ECG classification, a higher accuracy is desired in order not to cause false alarms or miss dangerous situations. This chapter presents an improved Learning Vector Quantisation (LVQ) neural network. The reason for the focus on LVQ network is their proven strong classification abilities [Baig et al., 2001].

The chapter describes a new modification to the LVQ neural network to yield the all weights updating LVQ (AWU-LVQ). AWU-LVQ employs a new method for updating weights that uses the Gaussian function in order to determine the amount of update for each weight in the network. This algorithm improves the classification accuracy of the standard LVQ and shortens the learning time.

This chapter gives details of the LVQ and the new AWU-LVQ algorithms, and their application to the recognition of ECG patterns from 5 different classes.

4.2 Learning Vector Quantisation (LVQ)

The LVQ neural network was developed by Kohonen [Kohonen, 1989], and has been successfully used for many classification problems. The learning method is supervised and based on "competitive" learning, in which neurons compete to have their weights updated. Figure 4.1 shows the LVQ network architecture, which consists of three layers of neurons: an input buffer layer, a hidden layer and an output layer. The network is fully connected between the input and hidden layers and partially connected between the hidden and output layers, with each output neuron linked to a different cluster of hidden neurons.

The weights of the connections between the hidden and output neurons are fixed at 1. The weights of the input to hidden neuron connections form the components of "reference" vectors, with one reference vector assigned to each hidden neuron. When an input vector is supplied to the network for recognition, the hidden neuron whose reference vector is closest in terms of Euclidean distance to the input vector is said to win the competition against all the other hidden neurons to have its output set to "1". All other hidden neurons are forced to produce a "0". The output neuron connected to the cluster of hidden neurons that contains the winning neuron also emits a "1" and all other output neurons, a "0". The output neuron that produces a "1" gives the class of the input vector, each output neuron being dedicated to a different class.

In the learning stage, the neurons in the hidden layer again compete amongst themselves in order to find the winning neuron whose weight vector is most similar to

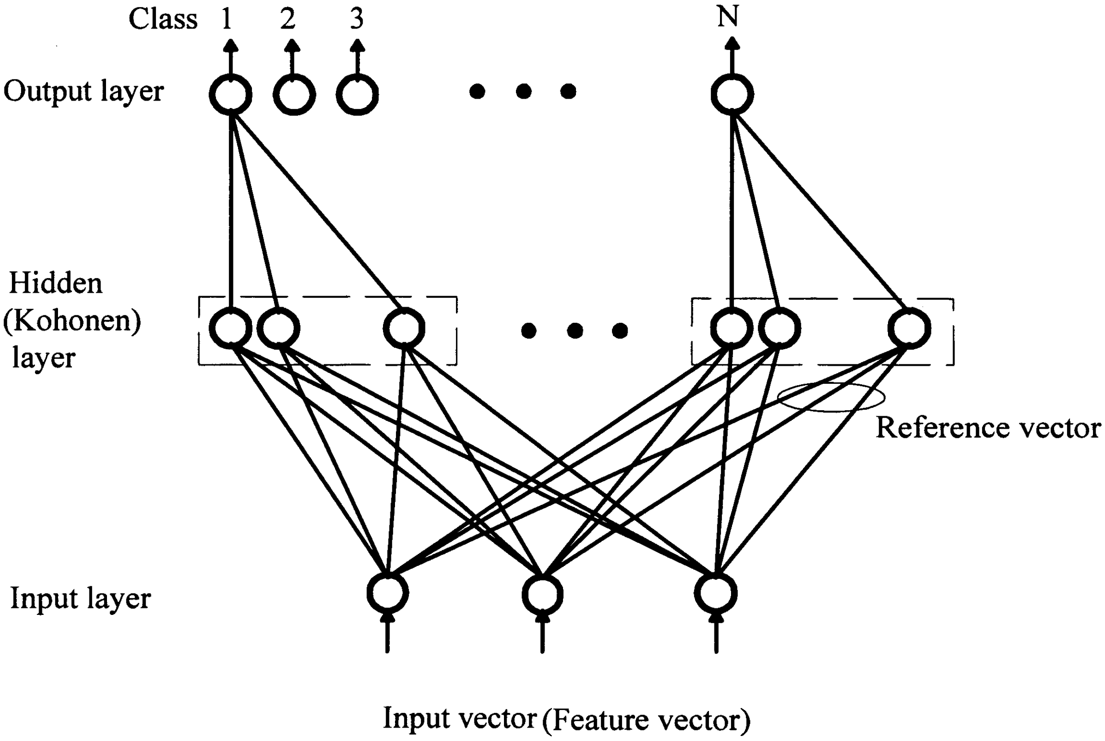


Figure 4.1: Learning vector quantisation network

the input vector, the winning neuron being that one with weight vector closest to the input vector [Kohonen, 1990]. The winning neuron gives the class of the input vector as in the recognition phase. Only the winning neuron will modify its weights using a positive or negative reinforcement learning formula, depending on whether the class indicated by the winning neuron is correct or not. If the winning neuron belongs to the same class as the input vector (the classification is not correct), it will be allowed to increase its weights, moving slightly closer to the input vector (positive reinforcement). On the contrary, if the class of the winning neuron is different from the input vector class (the classification is not correct), it will be made to decrease its weights, moving slightly further from the input vector (negative reinforcement).

Weights are modified during the training of the network. Both the hidden neurons (also known as Kohonen neurons) and the output neurons have binary outputs.

The competition between the Kohonen neurons is based on the Euclidean distance between the weight vectors $w_i(t)$ and the input vector *x (t)*. The calculation of the Euclidean distance d_i between *x (t)* and $w_i(t)$ is as follows:

$$d_i = \| w_i(t) - x(t) \| = \sqrt{\left(\sum_j \left(w_{ij}(t) - x_j(t) \right)^2 \right)} \tag{4.1}$$

where w_{ij} and x_j are the j^{th} components of w_i and x. As mentioned before, the neuron that has the minimum distance wins the competition and is allowed to change its connection weights, while the rest of the weights remain unchanged. The new weights are given by:

$$w_i(t+1) = w_i(t) + \alpha\ (x_i(t) - w_i(t)) \tag{4.2}$$

if the winning neuron is in the same category as the output, which means the weights will move closer to the input vector, and by:

$$w_i(t+1) = w_i(t) - \alpha\ (x_i(t) - w_i(t)) \tag{4.3}$$

if the winning neuron is in the wrong category, which means the weights will move further away from the input vector. In Equations 4.2 and 4.3, α is the learning rate, which decreases monotonically with the number of iterations. Usually, $0 < \alpha < 1$.

A selection of pattern recognition applications that unitise LVQ algorithm is given below:

- Object orientation detection [Morris et al., 1990].
- Protein classification [Merelo et al., 1991].
- Classification of liver tissues [Pan and Chen, 1992].
- Classification of seismic events [Jang et al., 1993].
- Classification of electroencephalographic EEG power spectra [Veselis et al., 1993].
- Inspection of coated steel samples using scattering angles as inputs [Olsson and Gruber, 1993].

- Analysis of multivariate biological data [Wilkins et al., 1994].
- Discrimination between various partial discharge pulse shapes [Mazroua et al., 1994].
- Combined image compression and classification [Oehler and Gray, 1995].
- Detection and location of gross errors in plant performance data [Aldrich and Vandeventer, 1995].
- Control chart pattern recognition [Pham and Oztemel, 1994].
- Defect detection in concrete [Shoukry et al., 1996].
- Automatic speech recognition [Cosi et al., 2000].
- Performance analysis and optimisation of shape recognition and classification [Nabhani and Shaw, 2002].
- Classification of heart sounds [Olmez and Dokur, 2003].
- Real-time fault detection and isolation in industrial machines [Marzi, 2004].

4.3 All Weights Updating-LVQ Technique

As describe above, in the original LVQ neural network, only one weight will be modified in each training phase. In later versions, such as LVQ2 [Kohonen, 1989] and

LVQ-X [Pham and Oztemel, 1994], two weights (that of the winning neuron "global winner", and the next weight which is in the correct category and nearest to the input vector in that category "local winner") will be modified.

The AWU-LVQ algorithm is based on updating all the weights at the same time in each iteration during the training stage. However, the amount of updating will vary according to how far a neuron is from the winner in the hidden layer. The weight of the winning neuron will receive the maximum amount of updating, while the other neurons will be updated to reducing degrees as their distances from the winner increase. Figure 4.2 shows the Gaussian function used to determine the amount of updating. The Gaussian function was adopted because it enables a smooth asymptotic reduction in the amount of weight update. If a neuron is in the same class as the input vector, this neuron will move toward the input vector and if it is not, it will move away from the input vector.

Let the input vectors be represented by $x_i\,(t)$ and the weights of the network by $w_i(t)$. If the neuron weight vector closest to the input vector is given the index c, the Gaussian function $h_i\,(t)$ is then used to adjust the amount of updating according to the following equation:

$$h_i(t) = e^{-\frac{\| d_i - d_c \|}{2\sigma^2}} \tag{4.4}$$

where the parameter σ defines the width of the Gaussian function which determines the degree of neuron excitation.

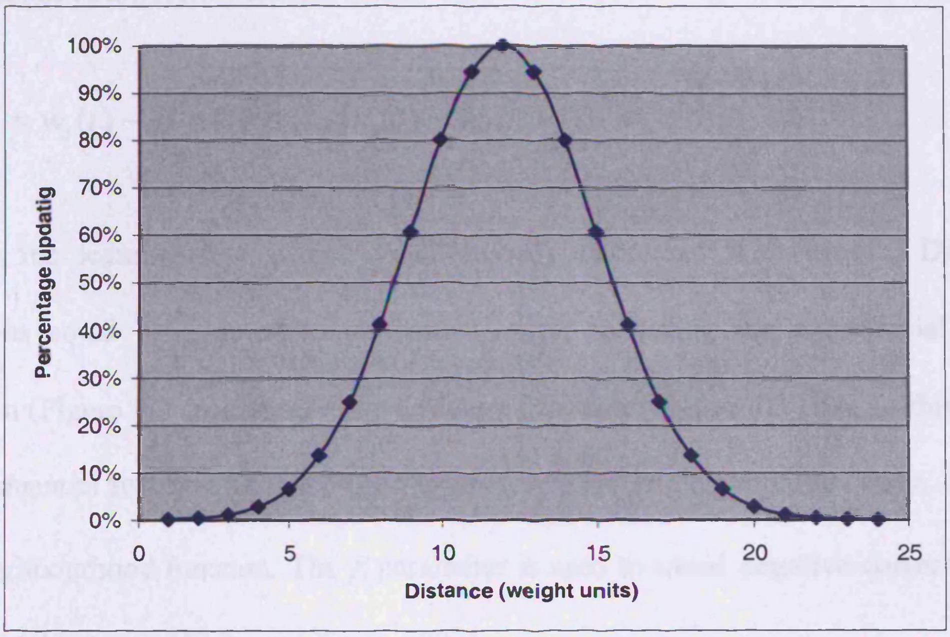


Figure 4.2: Gaussian weight updating function

d_c is Euclidean distance between the next closest neuron of the winning neuron and the input vector

The weight update Equation for neurons in the same class as the input vector is:

$$w_i(t) = w_i(t) + \alpha(t)\ h_i(t)\,[x_i(t) - w_i(t+1)] \tag{4.5}$$

For neurons in different classes from the input vector, the weight update Equation is:

$$w_i(t) = w_i(t) - \beta\ \alpha(t)\ h_i(t)\,[x_i(t) - w_i(t+1)] \tag{4.6}$$

α(t) is the learning rate, which monotonically decreases with time *t*. Different functions could be adopted to implement $\alpha(t)$, including the exponential decay function (Figure 4.3 (a)) and the linear decay function (Figure 4.3 (b)). In this work, an exponential function has been used to provide a smooth asymptotic decay. $h_i(t)$ is the neighbourhood function. The β parameter is used to avoid negative corrections in supervised learning [Yang and Yang, 2002].

The AWU-LVQ algorithm works as follows:

1- Decide the network architecture (number of inputs, number of hidden neurons, and number of outputs). Randomly initialise the values of all the reference weight vectors.

2- Initialise the learning rate α and the width of the Gaussian function σ.

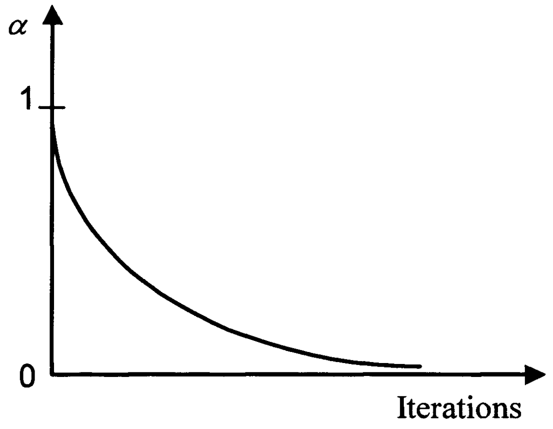


Figure 4.3 (a): Exponential decay function

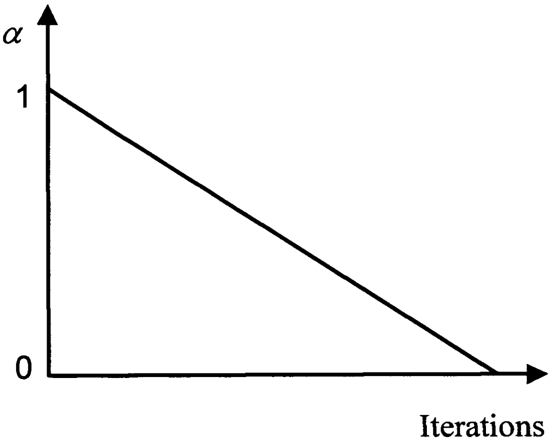


Figure 4.3 (b): Linear decay function

3- Present the input (training) vector to the network.

4- Find the winning neuron i by calculating the Euclidean distance between the input vector and all the network weights using Equation 4.1

5- Calculate the distance function h_i for all the reference vectors using Equation 4.4.

6- Update all the weights of the neurons in the hidden layer. If a neuron is in the same class as the input, its weight vector will move toward the input vector according to the Equation 4.5, and if not, the weight vector will move away using the Equation 4.6.

7- Reduce the learning rate $\alpha(t)$ and go to step 3 with a new input vector. Repeat the procedure until all input vectors are correctly classified or a stopping criterion is met.

4.4 Experimental Results

The experimental work described in this chapter was performed using two neural networks, LVQ and AWU-LVQ, in order to compare these networks from the point view of classification accuracy and training time.

4.4.1 Network Configuration

The AWU-LVQ and LVQ neural networks were given the same architecture to facilitate their comparison. Data consisting of input vectors with 15 features as shown in Table (4.1) and from 5 classes were utilised. The networks had 15 neurons in the input layer, 25 neurons in the hidden layer and 5 neurons in the output layer. The 15 inputs represent the 15 features extracted from one cycle of the ECG signal. The output layer has 5 neurons, representing the five types of ECG arrhythmia classes (V, L, P, R, and N ECG class) chosen for this experimental work. The hidden layer consisted of 25 neurons, allowing five reference vectors per class. The weights were initialised randomly to values between -1 and 1.

The learning rate was set to 0.1 and was made to decrease exponentially with iteration time as already mentioned. The exponentially decay coefficient was 0.01. The standard deviation of the Gaussian distance function for the AWU-LVQ algorithm was 3. All the values used were empirically chosen.

4.4.2 Training and Test Sets

The training and test data sets employed consisted respectively of 400 and 150 patterns for each of the 5 classes.

Feature	Significance	Description
Signal period	Physiological	Time between successive R-waves.
Q peak height	Physiological	Amplitude of the Q-wave.
R peak height	Physiological	Amplitude of the R-wave.
S peak height	Physiological	Amplitude deflection of the Q-wave.
Q-wave duration	Physiological	Overall duration of the Q-wave.
R-wave duration	Physiological	Overall duration of the R-wave.
S-wave duration	Physiological	Overall duration of the S-wave.
QRS interval	Physiological	Overall duration of the QRS complex – from the onset of the Q-wave to the end of the S-wave. (Time taken for complete ventricular pumping action).
RS height	Physiological	Distance between the peak of the R-wave and depth of the S-wave.
R to S height ratio	Clinical	Ratio of R-wave height to S-wave depth (This indicates the orientation of the electrodes relative to the axis of the heart).
Q to QRS duration ratio	Physiological	Ratio of the durations of the Q-wave and QRS complex.
Mean	Statistical	Mean value of the electrical signal.
Standard deviation	Statistical	Standard deviation of the electrical signal from the baseline.
Skewness	Statistical	Statistical skewness of electrical signal.
Excess	Statistical	Statistical excess of electrical signal.

Table 4.1: Features of ECG signal selected for classification

The classification accuracy was again calculated using the following equation:

$$\text{Accuracy}(\%) = \frac{\text{Nunmers of test patterns correctly classified}}{\text{Total numbers of pattern tested}} * 100 \qquad (4.7)$$

4.4.3 AWU-LVQ and LVQ Results

Several experiments were carried out in order to select two AWU-LVQ algorithm parameters, namely, the number of training epochs and the width σ of the kernel of the Gaussian distance function.

Table 4.2 shows the influence of σ on the classification accuracy. It can be noted that the accuracy improves with increasing σ, up to a value of 3.5.

When σ is small only a few neurons near to the winning neuron will be updated by a significant amount, and the rest of the neurons will receive only small amounts of updating or even no updating, which explains why the classification accuracy is poor and highly variable. When σ increases, more neurons will be updated to different degrees, which yields improvement in the classification results (higher accuracy and lower variance).

The effect of epoch number on classification accuracy is presented in Table 4.3. This shows an increase in accuracy up to 2000 epochs and a decrease thereafter. This happened because of overtraining of the neural network which makes it lose its ability

	Gassian function kernel width						
Run	1	1.5	2	2.5	3	3.5	4
1	85.60%	89.60%	85.60%	85.20%	85.20%	85.60%	85.20%
2	20.00%	20.00%	89.60%	85.20%	85.60%	85.60%	85.60%
3	85.60%	85.60%	85.60%	85.20%	85.60%	85.20%	85.20%
4	90.80%	85.60%	89.60%	85.20%	85.20%	85.60%	85.60%
5	85.60%	20.00%	85.60%	85.20%	85.60%	85.60%	85.60%
6	20.00%	90.00%	58.00%	85.20%	85.60%	85.60%	85.20%
7	85.60%	89.60%	85.60%	85.20%	85.20%	85.60%	85.20%
8	90.00%	90.00%	85.60%	85.20%	85.20%	85.20%	85.20%
9	85.60%	85.60%	85.60%	85.60%	85.20%	85.20%	85.60%
10	89.60%	89.60%	85.60%	85.20%	85.20%	85.20%	85.60%
Average	73.84%	74.56%	83.64%	85.28%	85.36%	85.44%	85.40%
Variance	809.572	830.514	83.9271	0.028	0.0427	0.0427	0.0444

Table 4.2: Effect of kernel width on the classification accuracy

(learning rate α = 0.1 and epoch number = 200)

Epoch number	Classification accuracy
5	54.16%
10	73.60%
20	74.48%
50	79.84%
100	80.24%
200	83.64%
500	87.24%
1000	87.56%
2000	88.32%
2500	86.76%

Table 4.3: Effect of epoch number on the classification accuracy

(learning rate $\alpha = 0.1$ and kernel width $\sigma = 2$)

to generalise. In other words, the network simply memorises the training patterns and cannot recognise new test patterns. This phenomenon is similar to overtraining in MLP neural networks [Pham and Liu, 1995] Table 4.4 presents more results concerning the effect of the number of epochs on the classification accuracy of the AWU-LVQ network. It can be seen that beyond 50 epochs there was a small drop in the classification accuracy and beyond 200 epochs there was no noticeable improvement in the stability of the classification results.

Table 4.5 shows the results for the LVQ algorithm with $\alpha = 0.1$ and different numbers of epochs.

Table 4.6 compares AWU-LVQ and LVQ. From this table the improvement of AWU-LVQ over the standard LVQ in both classification accuracy and training time is evident. The best classification accuracy using LVQ was 64.48% after 1000 epochs, while using AWU-LVQ the best classification accuracy was 85.44% after only 10 epochs.

	Epoch number									
Run	**1**	**2**	**5**	**10**	**20**	**50**	**100**	**200**	**500**	**1000**
1	38.00%	47.20%	82.00%	85.60%	85.60%	85.60%	85.60%	85.60%	85.60%	85.20%
2	20.00%	38.00%	85.20%	85.20%	85.60%	85.20%	85.60%	85.20%	85.20%	85.60%
3	38.00%	71.20%	85.20%	85.20%	85.20%	85.60%	85.60%	85.20%	85.20%	85.20%
4	20.00%	38.00%	85.20%	85.20%	85.20%	85.60%	85.20%	85.20%	85.20%	85.20%
5	20.00%	38.00%	85.20%	85.60%	85.20%	85.20%	85.60%	85.20%	85.20%	85.20%
6	32.80%	38.00%	85.20%	85.20%	85.60%	85.20%	85.20%	85.20%	85.20%	85.60%
7	38.00%	53.60%	85.20%	85.60%	85.60%	85.60%	85.20%	85.60%	85.60%	85.20%
8	38.00%	38.00%	85.20%	85.60%	85.20%	85.60%	85.20%	85.20%	85.20%	85.20%
9	38.00%	38.00%	77.20%	85.60%	85.60%	85.20%	85.20%	85.20%	85.20%	85.60%
10	20.00%	49.60%	85.20%	85.60%	85.60%	85.60%	85.60%	85.60%	85.60%	85.20%
Average	**30.80%**	**44.96%**	**84.08%**	**85.44%**	**85.44%**	**85.44%**	**85.40%**	**85.32%**	**85.32%**	**85.32%**
Variance	80.784	120.0427	6.8551	0.0427	0.0427	0.0427	0.0444	0.0373	0.0373	0.0373

Table 4.4: Effect of number of epochs on AWU-LVQ classification accuracy

(learning rate α = 0.1, kernel width σ = 3)

	Epoch number									
Run	1	2	5	10	20	50	100	200	500	1000
1	20.00%	20.00%	20.00%	36.40%	54.00%	64.00%	65.20%	64.00%	66.40%	66.40%
2	20.00%	20.00%	20.00%	36.40%	63.20%	64.00%	65.20%	67.60%	64.00%	64.00%
3	20.00%	20.00%	20.00%	36.40%	56.00%	64.00%	64.00%	65.20%	65.20%	65.20%
4	20.00%	20.00%	20.00%	36.40%	52.80%	64.40%	64.00%	64.00%	65.60%	65.60%
5	20.00%	20.00%	20.00%	36.40%	63.20%	64.00%	64.00%	64.00%	65.20%	65.20%
6	20.00%	20.00%	20.00%	36.40%	55.20%	64.00%	64.00%	64.00%	64.00%	64.00%
7	20.00%	20.00%	20.00%	36.40%	57.20%	64.80%	64.00%	64.00%	64.80%	64.80%
8	20.00%	20.00%	20.00%	36.40%	55.20%	65.20%	64.00%	67.60%	64.40%	66.40%
9	20.00%	20.00%	20.00%	36.80%	52.80%	65.20%	64.40%	64.00%	64.40%	66.40%
10	20.00%	20.00%	20.00%	36.80%	55.60%	64.00%	65.60%	64.00%	66.00%	66.80%
Average	**20.00%**	**20.00%**	**20.00%**	**36.48%**	**56.52%**	**64.36%**	**64.44%**	**64.84%**	**65.00%**	**65.48%**

Table 4.5: LVQ classification results (learning rate $\alpha = 0.1$)

Epochs	LVQ	AWU-LVQ
1	20.00%	30.80%
2	20.00%	44.96%
5	20.00%	84.08%
10	36.48%	85.44%
20	56.52%	85.44%
50	64.36%	85.44%
100	64.44%	85.40%
200	64.84%	85.32%
500	65.00%	85.32%
1000	65.48%	85.32%

Table 4.6: Comparison between AWU-LVQ and LVQ

4.5 Summary

This chapter has introduced a modification to the standard LVQ algorithm, the All Weights Updating LVQ (AWU-LVQ), which operates by simultaneously updating all network weights during training. Results of applying both AWU-LVQ and LVQ algorithms for ECG classification have been presented. These show marked improvements of AWU-LVQ over the standard LVQ although the classification accuracy of AWU-LVQ is still lower than those of C5.0 and other known classifiers.

Chapter 5

Enhanced Learning Vector Quantisation Network using Neighbouring Weights Updating

5.1 Preliminaries

This chapter presents another modification to the LVQ neural network, the Neighbouring Weights Updating LVQ (NWU-LVQ) algorithm. This employs a new method of weights updating that modifies the weights of the winning neuron and its neighbours but in a supervised way. The aim was to reduce training time and increase classification accuracy.

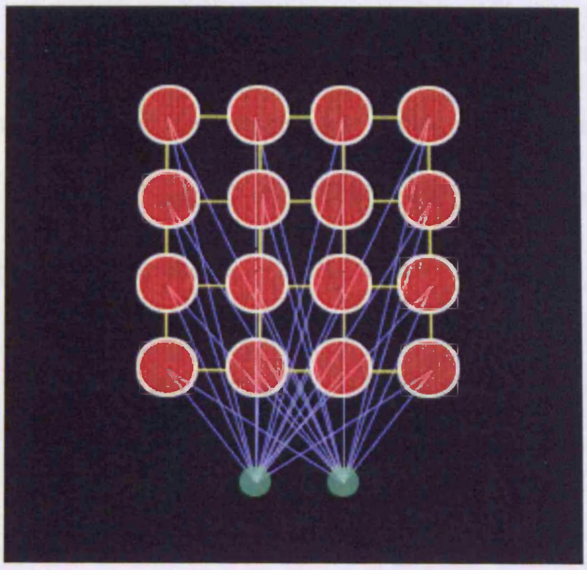
The performance of the NWU-LVQ algorithm has been compared with that of the recent Fuzzy Soft Learning Vector Quantisation FS-LVQ algorithm by Yang and Yang [Yang and Yang, 2002] which also adapts the weights of the neighbours of the winning neuron.

This chapter first reviews the Self Organising Map (SOM) on which the FS-LVQ is based. The FS-LVQ algorithm is then detailed, followed by the NWU-LVQ algorithm. A comparison of the results obtained by FS-LVQ and NWU-LVQ in classifying ECG patterns belonging to five different types is given at the end of the chapter.

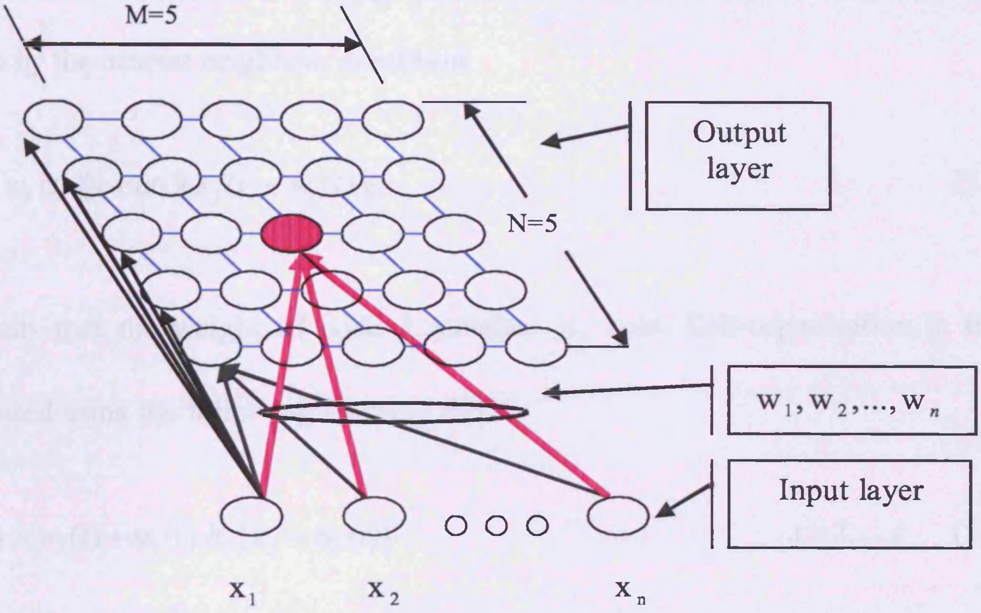
5.2 Self Organising Map

The Self Organising Map (SOM) was introduced by Kohonen in 1981 [Kohonen, 1982]. The SOM (also known as the Kohonen feature map or Self Organising Feature Map (SOFM)) algorithm is one of the best-known artificial neural network algorithms. The SOM is an unsupervised feed forward neural network. It does not require an external teacher, but is a competitive topology-preserving map which can be adjusted to reflect the probability of distribution of the inputs [Kohonen, 1984]. In this topology, neurons located next to one another on the map correspond to inputs that are close to one another. By using this map, a high–dimensional input vector can project onto a low–dimensional map, in a way that maintains the natural order of the input vectors. Thus, the map is able to convert complex non-linear statistical relationships between high-dimensional data items into simple geometric relationships on a low-dimensional display.

A SOM consists of two-layers, an input and an output layer (the Kohonen layer) trained using competitive learning. Figure 5.1 shows the architecture of a SOM with a two-dimensional rectangular array of output nodes. In some cases, the output neurons could form a one-dimensional array rather than a rectangular grid as illustrated in Figure 5.1.



(a)



(b)

Figure 5.1: A typical two-dimensional Kohonen self organising map

(a) (Adapted from http://www.ai-junkie.com)

In a SOM, each node in the output layer has a weight vector with dimensions that are the same as the input vector dimension. The weight vectors are initialised randomly, so that the outputs of all nodes will be different and so only one will respond maximally to a particular input vector. This "winning" node and its neighbours will be trained so that they can respond to that particular input vector even more strongly. After a period of time, different areas of the SOM will respond more strongly to a particular input vector than other areas.

By identifying the most strongly-responding areas of the SOM, different input vectors can be recognised.

Suppose $w_k(t)$ in R^n is the weight of node k and the feature vector $x_j(t)$ in R^n is presented at time t, then the winning neuron k among all competitors (neurons) is produced by the nearest neighbour condition:

$$\| x_j(t) - w_k(t) \| = \min_i \| x_j(t) - w_i(t) \| \tag{5.1}$$

This means that the weight of node k matches x_j best. Self-organisation is then implemented using the following learning rule:

$$w_i(t+1) = w_i(t) + \alpha_i(t)\, h_i\, [x_j - w_i(t)] \qquad i = 1, \ldots, c \tag{5.2}$$

where $\alpha_i(t)$ is the learning rate factor of node i and is confined to decrease monotonically with time t. The neighbourhood function h_i denotes the lateral neural interaction phenomenon and the degree of excitation of the neuron. A simpler definition of h_i is:

$$h_i = \begin{cases} 1 & \textit{if the node i belongs to } N_k(t) \\ 0 & \textit{otherwise} \end{cases} \tag{5.3}$$

$N_k\ (t)$ is called the neighbourhood set of the winner k and is made to decrease with time to accomplish the "winner-take-all" principle:

$$h_i = \begin{cases} 1 & \textit{if } i = k, \\ 0 & \textit{if } i \neq k \end{cases} \tag{5.4}$$

h_i is a unimodal function with lateral distance d, such that it satisfies two distinct requirements:

- The topology of h_i is symmetric about the maximum point defined by $d = 0$; in other words, it attains its maximum value at the winning neuron k for which the distance d is zero.

- The amplitude of the topological neighbourhood about the maximum value decreases monotonically with increasing lateral distance d, decaying to zero for ($d \rightarrow 0$); this is a necessary condition for convergence.

After a feature vector input $x_j\,(t)$ is presented to the SOM, the weight of the winning node and the weights of the neighbours will be updated towards the input vector with a step size $\alpha\,(t)*\,h_i\,(t)$ with corresponding definitions of $h_i(t)$ in Equation (5.4).

A typical choice of h_i that satisfies these requirements is the Gaussian function [Haykin, 1999]:

$$h_i = \exp - \left(\frac{\| r_j - r_k \|^2}{2\sigma^2(t)} \right) \tag{5.5}$$

where r_j and r_k in R^2 are the vectorial locations of the display (output) grid in Figure (5.1). $\sigma(t)$ is the effective width of the function, corresponding to the kernel width of the nodes, which determines the degree of neuron excitation. $\sigma(t)$ also decreases monotonically as time t increases. In a qualitative sense, the Gaussian topological neighbourhood is more appropriate than a rectangular one. Its use also makes the SOM converge more quickly than a rectangular topological neighbourhood would [Lo et al., 1991; 1993; Erwin et al., 1992].

5.3 Fuzzy Soft Learning Vector Quantisation (FS-LVQ)

Wu and Yang [Wu and Yang, 2002] applied fuzzy soft techniques to SOM and proposed an unsupervised competitive learning network called fuzzy-soft unsupervised learning vector quantisation (FS-U-LVQ) network.

This algorithm uses a fuzzy relaxation technique to update simultaneously all neurons according to the difference between the reference vector of each neuron and the input vector during the learning phase.

The FS-U-LVQ network is created by setting the learning rate $\alpha_i(t)$ to decrease according to the degree of excitation for individual neurons as described in the following equation:

$$\alpha_i(t) = \frac{\alpha_0}{\dfrac{\alpha_0}{\alpha_i(0)} + \sum_{j=1}^{t} h_i(j)} \tag{5.6}$$

where $\alpha_0(0)$ controls the starting value of the learning rate for separated neurons and α_0 will adjust the rate of decrease, and the neighbourhood function *h_i (t)* to

$$h_i(t) = \left(\frac{\mu_i(x(t))}{\max\limits_{1 \le j \le c} \mu_j(x(t))} \right)^{(1+f(t)/c)} \tag{5.7}$$

with

$$\mu_i(x(t)) = \left(\sum_{k=1}^{c} \frac{\| x(t) - w_i(t) \|^{2/(m-1)}}{\| x(t) - w_k(t) \|^{2/(m-1)}} \right)^{-1} \tag{5.8}$$

Note that $(\mu_i(x(t)), i = 1,\ldots,c)$ are the Fuzzy C-Mean (FCM) membership functions. In general, *m* = 2 is chosen. The function *f(t)* is a monotonically increasing function of *t* that controls the excitation of neurons. In general, *f(t)* can be chosen as $\sqrt{t}$, *t or* t^2 etc.

Using their FS-U-LVQ, Yang and Yang [Yang and Yang, 2002] recently proposed a supervised version called fuzzy-soft LVQ (FS-LVQ). They employed the following weight update equation:

$$w_i(t+1) = w_i(t) + \alpha_i(t)\, h_i(t)\, (x_j(t) - w_i(t)) \tag{5.9}$$

If neuron i is in the correct category.

and

$$w_i(t+1) = w_i(t) - \beta\, \alpha_i(t)\, h_i(t)\, (x_j(t) - w_i(t)) \tag{5.10}$$

If neuron i is in the wrong category.

Parameter $\beta = 0.05$ is used to suppress negative corrections.

5.4 Neighbouring Weights Updating LVQ (NWU-LVQ)

There are, at most, two weights to be modified with the existing LVQ and LVQ-X [Pham and Oztemel, 1993] methods. Like the FS-LVQ technique, the proposed Neighbouring Weights Updating LVQ (NWU-LVQ) technique modifies the weight of the winning neuron and all neighbouring weights simultaneously according to the differences between the input vector and the reference vectors of individual neurons.

NWU-LVQ follows the same updating criteria as the basic LVQ algorithm. If a reference vector is in the same class as the input vector, it will move closer to the input vector, and if it is not, it will move away.

The steps in the NWU-LVQ algorithm are as follows:

1- Decide the network architecture, such as the numbers of inputs, hidden neurons, and outputs.

2- Initialise the learning rate α_i *(t)*, the width σ *(t)* of the Gaussian function. The latter defines the radius of the neighbourhood of the winning neurons. Only the weights of neurons in that neighbourhood will be updated in each epoch.

3- Present an input vector to the network.

4- Find the winning neuron *k* by calculating the Euclidean distance between the input vector and all the reference vectors and selects the neuron with the shortest distance using Equation (5.1).

5- Evaluate the Gaussian function for the neighbours at the winning neuron using Equation (5.5).

6- Examine each neighbour. If it is in the same class as the input vector, its weight will move toward the input vector according to Equation (5.9), and if it is not, it will move away according to Equation (5.10).

7- Go to 3 with a new input vector and repeat the procedure until all input vectors are correctly classified or a stopping criterion is met.

5.5 Experimental Results

The FS-LVQ and NWU-LVQ networks were compared from the point of view of accuracy and training time.

5.5.1 Network Configuration

The same 15 features as used in the work repeated in Chapter 4 (see Table (4.1)) were employed. The features combined statistical (standard deviation, mean), clinical (R to S height ratio), and physiological measures (QRS interval, Q-wave duration). The input layer therefore had 15 input neurons. These 15 inputs were used to represent each cycle of an ECG. As for the hidden layer there were 25 neurons, 5 neurons representing each class. This means there were five reference vectors per category. The output layer consists of 5 neurons for the five types of ECG arrhythmia, namely N, V, L, P, and R.

5.5.2 Training and Test Sets

The total number of examples in the training set is 2000, 400 for each class, and the total number of examples in the test set are 750, 150 for each class.

	Epoch number										
Run	1	2	5	10	20	50	100	200	500	1000	2000
1	20	40	40	85.6	89.6	85.6	90.8	90.8	85.6	85.6	85.6
2	20	20	74	58.8	85.6	90.8	90.8	90.8	85.6	85.6	90.8
3	38.8	20	40	78.4	85.6	90.8	90.8	85.6	85.6	85.6	90.8
4	20	38.8	40	79.2	85.6	90.8	90.8	85.6	85.6	85.6	85.6
5	26.8	38	20	40	90.8	85.6	90.8	90.8	85.6	85.6	90.8
6	34	40	20	74	90.8	90.8	85.6	85.6	90.8	90.8	85.6
7	34	38	60	85.6	90.8	85.6	85.6	85.6	90.8	90.8	85.6
8	20	38	54	54	85.6	85.6	85.6	85.6	85.6	85.6	85.6
9	36.8	40	40	91.6	85.6	90.8	90.8	90.8	90.8	90.8	85.6
10	20	20	38.8	85.6	90.8	90.8	90.8	85.6	85.6	85.6	85.6
Average	**27.04%**	**33.28%**	**42.68%**	**73.28%**	**88.08%**	**88.72%**	**89.24%**	**87.68%**	**87.16%**	**87.16%**	**87.16%**

Table 5.1: NWU-LVQ classification accuracy with different numbers of training epochs

(learning rate $\alpha = 0.1$, Kernel width $\sigma = 3$)

5.5.3 NWU-LVQ and FS-LVQ Results

Several experiments were conducted with the NWU-LVQ algorithm in order to study the effect of the number of training epochs and the kernel width of the Gaussian function.

Table 5.1 shows how classification accuracy varies with the number of training epochs. As mentioned in the pervious chapter exponential function was adopted to the learning rate. The initial learning rate (α) was 0.1, and the kernel width (σ) 3. It can be noted that the classification accuracy increased with the number of training epochs, up to 100 epochs, after which the classification accuracy reduced.

Table 5.2 shows the effect of the kernel width on classification accuracy. The epoch number was fixed at 100 and the initial learning rate at 0.1. Note that the classification accuracy was poor at small kernel widths, improving as the kernel width increased, and then reducing again slightly with kernel widths over 3.

This could be explained as follows. When the kernel width is small, in the range [1-1.5], the shape of the Gaussian function is very narrow and the difference in the weight changes between the winner and the remainder of the neurons is very large. As the kernel width increases, the weight update difference between the winner and the remainder of the neurons reduces. This means that, in addition to more neurons being updated, they are also updated in a more uniform way, improving the classification accuracy and the stability of the results.

	Kernal value							
Run	1	1.5	2	2.5	3	3.5	4	4.5
1	85.6	85.6	90.8	85.6	90.8	85.6	85.6	89.6
2	72.4	85.6	85.6	90.8	90.8	89.6	85.6	85.6
3	85.6	90	85.6	85.6	90.8	85.6	89.6	85.6
4	58.8	58.4	90.8	85.6	90.8	85.6	85.6	85.6
5	90.8	85.6	85.6	85.6	90.8	89.6	85.6	85.6
6	90.8	90	90.8	90.8	85.6	89.6	85.6	89.6
7	72.4	85.6	90.8	90.8	85.6	85.6	89.6	89.6
8	54	90.8	90.8	90.8	85.6	89.6	89.6	85.6
9	20	90	90.8	90.8	85.6	89.6	89.6	89.6
10	90.8	85.6	90.8	90.8	90.8	89.6	85.6	85.6
Average	72.12%	84.72%	89.24%	88.72%	88.72%	88.00%	87.20%	87.20%

Table 5.2: NWU-LVQ classification accuracy with different kernel values

(learning rate $\alpha = 0.1$ and epoch number = 100)

However, when too wide a kernel is employed, the possibility of interference between the different clusters increases and this causes a deterioration on the classification accuracy.

Table 5.3 shows the results of applying the NWU-LVQ algorithm with $\sigma = 3$ and $\alpha = 0.1$ for different epoch numbers.

Table 5.4 summarises the results for the FS-LVQ algorithm.

Table 5.5 summarises the results for the FS-LVQ algorithm and the NWU-LVQ algorithm with $\alpha = 0.1$.

NWU-LVQ clearly gives a better classification accuracy in a shorter training time compared to FS-LVQ. For ease of comparison, the results for the original LVQ and the AWU-LVQ networks presented previously are also shown in Table 5.5. It can be seen that NWU-LVQ is the best performing neural network and the original LVQ network is the poorest classifier.

The NWU-LVQ network learned faster than the FS-LVQ network although the classification accuracy of the latter is marginally higher. The performance of the FS-LVQ network was also more stable in that there were no noticeable variations between the different runs.

	Epoch number						
Run	1	2	5	10	20	50	100
1	20	40	40	85.6	89.6	85.6	90.8
2	20	20	74	58.8	85.6	90.8	90.8
3	38.8	20	40	78.4	85.6	90.8	90.8
4	20	38.8	40	79.2	85.6	90.8	90.8
5	26.8	38	20	40	90.8	85.6	90.8
6	34	40	20	74	90.8	90.8	85.6
7	34	38	60	85.6	90.8	85.6	85.6
8	20	38	54	54	85.6	85.6	85.6
9	36.8	40	40	91.6	85.6	90.8	90.8
10	20	20	38.8	85.6	90.8	90.8	90.8
Average	27.04%	33.28%	42.68%	73.28%	88.08%	88.72%	89.24%

Table 5.3: Classification accuracy of NWU-LVQ for different epoch numbers for

(learning rate α = 0.1 and kernel width σ = 3)

	Epoch number						
Run	1	2	5	10	20	50	100
1	23.6	17.2	26	62	85.6	85.6	85.6
2	7.6	30.8	20	82	85.6	85.6	85.6
3	20	14.4	20	72.8	85.6	85.6	85.6
4	20	15.6	20	74	85.6	85.6	85.6
5	32.4	20	19.2	41.2	85.6	85.6	85.6
6	10.4	20	20	51.2	85.6	85.6	85.6
7	18	10.4	23.2	89.6	85.6	85.6	85.6
8	20	20	20	41.2	85.6	85.6	85.6
9	7.6	20	38.8	79.6	85.6	85.6	85.6
10	22.8	20	33.2	51.6	85.6	85.6	85.6
Average	18.24%	18.84%	24.04%	64.52%	85.60%	85.60%	85.60%

Table 5.4: Classification accuracy of FS-LVQ for different epoch numbers

(learning rate α = 0.1 and kernel width σ = 3)

Epochs	LVQ	FS-LVQ	AWU-LVQ	NWU-LVQ
1	20.00%	18.24%	30.80%	27.04%
2	20.00%	18.84%	44.96%	33.28%
5	20.00%	24.04%	84.08%	42.68%
10	36.48%	64.52%	85.44%	73.28%
20	56.52%	85.60%	85.44%	88.08%
50	64.36%	85.60%	85.44%	88.72%
100	64.44%	85.60%	85.40%	89.24%

Table 5.5: Summary of classification results

5.4 Summary

After reviewing two neural network algorithms, SOM and FS-LVQ, this chapter has introduced a new modification to the standard LVQ algorithm called NWU-LVQ. Both the FS-LVQ algorithm and the NUW-LVQ algorithm involve updating the weights of the neurons in the neighbourhood of the winning neuron.

The results of applying this technique and the FS-LVQ technique to ECG classification have been presented. These show the superior classification accuracy and learning speed of the proposed algorithm.

Chapter 6

Conclusions and Future Work

6.1 Preliminaries

This chapter presents the conclusions of this work, outlines the main contributions of the research and makes recommendations for further studies.

6.2 Conclusions

The aim of the research reported in this thesis was to investigate the use of intelligent techniques for automated ECG classification.

One of the objectives of the work was to design new features which effectively and concisely describe ECG signals. Several features were defined and compared using different classifiers to obtain an optimum set of derived features. It was found that the set of 18 features including physiological, clinical and statistical descriptors yielded the best accuracy, although a reduced set with 11 features could also be adopted for ECG classification purposes.

An attempt was made to reduce the amount of ECG data to be handled without having to extract features by re-sampling the exiting data. Sets of re-sampled ECG signals with 33 and 64 data points were compared. It was found that the former gave higher classification accuracies although they were not as good as the results obtained with the extracted features.

Another objective was to analyse and compare different types of classifiers including the k-NN classifier, the MLP and RBF neural networks and the C5.0 inductive learning algorithm. C5.0 gave the best classification results, although, rather surprisingly, the MLP proved a better classifier than the RBF neural network.

The final objective was to develop neural classifiers with shorter training times and higher accuracies than the MLP. Although classifiers obtained using C5.0 were the most accurate, it was decided to focus on neural classifiers because they would be more easily implemented in hardware to yield compact real-time arrhythmia diagnosis systems. The new classifiers were based on the LVQ NN which is well known for being a fast learning neural network.

Two new learning algorithms were developed to improve the performance of the standard LVQ NN in terms of classification accuracy and length of training time. These were called the All Weights Updating LVQ (AWU-LVQ) and Neighbouring Weights Updating LVQ (NWU-LVQ).

NWU-LVQ gave the best ECG classification accuracy, as well as the shortest training time, compared to standard LVQ, AWU-LVQ and FS-LVQ. The latter is an existing LVQ network employing fuzzy logic techniques for adapting the weights of neurons in the neighbourhood of the winning neuron.

6.3 Contributions

In summary, this research has contributed to the area of ECG classification and artificial neural networks.

The first contribution was in identifying the best set of features to be extracted from an ECG to facilitate the classification process.

The second contribution was in highlighting the most accurate classifier for the ECG classification problem from a range of possible classification techniques.

The third contribution was in proposing two improvements to the LVQ neural network to increase its classification accuracy and reduce its training time.

6.4 Future work

The following are possible topics for further study:

New ECG feature extraction techniques could be developed. It is uncertain, for example, whether the selection of ECG features within the physiological QRS complex, P- and T-waves, PR and ST segments actually helps or confuses the classifier. Also new, more statistically effective features like kurtosis, dispersion and entropy might be employed.

The effect of increasing the number of training examples on classifier performance, potentially enabling it to recognise and differentiate various classes more accurately, could be investigated.

Newly developed classification methods such as Support Vector Machines (SVMs) [Schlkopf et al., 2002] which are based on recent advances in statistical learning theory could be employed.

Further development of the LVQ network employing spiking neuron techniques [Maass and Bishop, 2001] could be explored to improve the classification accuracy.

Chapter 6

Conclusions and Future Work

6.1 Preliminaries

This chapter presents the conclusions of this work, outlines the main contributions of the research and makes recommendations for further studies.

6.2 Conclusions

The aim of the research reported in this thesis was to investigate the use of intelligent techniques for automated ECG classification.

One of the objectives of the work was to design new features which effectively and concisely describe ECG signals. Several features were defined and compared using different classifiers to obtain an optimum set of derived features. It was found that the set of 18 features including physiological, clinical and statistical descriptors yielded the best accuracy, although a reduced set with 11 features could also be adopted for ECG classification purposes.

An attempt was made to reduce the amount of ECG data to be handled without having to extract features by re-sampling the exiting data. Sets of re-sampled ECG signals with 33 and 64 data points were compared. It was found that the former gave higher classification accuracies although they were not as good as the results obtained with the extracted features.

Another objective was to analyse and compare different types of classifiers including the k-NN classifier, the MLP and RBF neural networks and the C5.0 inductive learning algorithm. C5.0 gave the best classification results, although, rather surprisingly, the MLP proved a better classifier than the RBF neural network.

The final objective was to develop neural classifiers with shorter training times and higher accuracies than the MLP. Although classifiers obtained using C5.0 were the most accurate, it was decided to focus on neural classifiers because they would be more easily implemented in hardware to yield compact real-time arrhythmia diagnosis systems. The new classifiers were based on the LVQ NN which is well known for being a fast learning neural network.

Two new learning algorithms were developed to improve the performance of the standard LVQ NN in terms of classification accuracy and length of training time. These were called the All Weights Updating LVQ (AWU-LVQ) and Neighbouring Weights Updating LVQ (NWU-LVQ).

NWU-LVQ gave the best ECG classification accuracy, as well as the shortest training time, compared to standard LVQ, AWU-LVQ and FS-LVQ. The latter is an existing LVQ network employing fuzzy logic techniques for adapting the weights of neurons in the neighbourhood of the winning neuron.

6.3 Contributions

In summary, this research has contributed to the area of ECG classification and artificial neural networks.

The first contribution was in identifying the best set of features to be extracted from an ECG to facilitate the classification process.

The second contribution was in highlighting the most accurate classifier for the ECG classification problem from a range of possible classification techniques.

The third contribution was in proposing two improvements to the LVQ neural network to increase its classification accuracy and reduce its training time.

6.4 Future work

The following are possible topics for further study:

New ECG feature extraction techniques could be developed. It is uncertain, for example, whether the selection of ECG features within the physiological QRS complex, P- and T-waves, PR and ST segments actually helps or confuses the classifier. Also new, more statistically effective features like kurtosis, dispersion and entropy might be employed.

The effect of increasing the number of training examples on classifier performance, potentially enabling it to recognise and differentiate various classes more accurately, could be investigated.

Newly developed classification methods such as Support Vector Machines (SVMs) [Schlkopf et al., 2002] which are based on recent advances in statistical learning theory could be employed.

Further development of the LVQ network employing spiking neuron techniques [Maass and Bishop, 2001] could be explored to improve the classification accuracy.

Appendix A Training and Test Data Extracted from the MIT-BIH Database

The training and test examples were chosen to include data from different records in the MIT-BIH database. A breakdown of the number of examples of each class taken from the different records in the database is given in Tables A and B.

Where the letters N, L, R, P, V, A, a, j, V and f stands for:

N : Normal Sinus Rhythm .

L : Left Bundle Branch Block Beat .

R : Right Bundle Branch Block Beat.

P : Paced Beat.

V : Premature Ventricular Contraction.

A :Atrial Premature Beat.

a : Aberrated Atrial Premature Beat.

j : Nodal (junctional) Escape Beat.

V : Ventricular Escape Beat.

f :Fusion of paced and normal beats.

Table A-Training Data

Record	N	V	P	L	R	A	f	a	E	j
100	16	1				1				
101	16					1				
102	16		600				13			
103	16								68	
104	16						1			
105	16	27								
106		54								
108	16	18								1
109		25		150						
111				150						
112										
113	16							4		
114	16									
115	16									
116	16									
117	16									
118		1			193					
119	17									
121	17									
122	17									

123	17	2								
124		1			193					14
200	17	54								
201	17							42		
202	17	12						4		
203	17	54						1		
205	17	15								
207				150						
208	17	55								
209	17	1								
210	17	55						10		
212	17				194					
213	17							14		
214				150						
215	17	55								
217	17									
219	17									
220	17									
222	17									49
223	17	55				29				
228	17	55								
230	17	1								

231		**1**								
332					**20**					
233	**17**	**55**								
234	**17**	**2**								
Total	**600**	**600**	**600**	**600**	**600**	**31**	**14**	**75**	**68**	**64**

Table B- Test data

Record	N	V	P	L	R	A	f	a	E	j
100	5									
101	5					1				
102	5		200				6			
103	5								35	
104	5									
105	5	14								
106		15								
108	5	10								
109		13		50						
111				50						
112										
113	5							2		
114	5									
115	5									
116	5									
117	5									
118		1			63					
119	5									
121	5									

122	5									
123	5	1								
124		1			63					7
200	5	15								
201	5							21		
202	5	6						2		
203	5	15						1		
205	5	8								
207				50						
208	5	15								
209	5									
210	5	15						5		
212	5				63					
213	5	14						7		
214				50						
215	5	14								
217										
219	5									
220	5									
222	5									25
223	5	14				15				
228	5	14								

230	**5**									
231										
332					**11**					
233	**5**	**14**								
234	**5**	**1**								
Total	**200**	**200**	**200**	**200**	**200**	**16**	**6**	**38**	**35**	**32**

Appendix B Statistical Formulae used in Feature Selection

Mean, $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$

where n = number of elements

x_i = value of element i

Standard deviation, $s = \left(\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \right)^{1/2}$

where n = number of elements

x_i = value of element i

$\bar{x}$ = mean value

Skewness, $y = \frac{\bar{x}(x_i - \bar{x})^3}{s^3}$

where x_i = value of element i

$\bar{x}$ = mean value

s = standard deviation

Excess, $e = \frac{\bar{x}(x_i - \bar{x})^4}{s^4 - 3}$

where x_i = value of element i

$\bar{x}$ = mean value

s = standard deviation

References

Abramson M. and Wechsler, H., (2001). Competitive reinforcement learning for combinatorial problems, *Proceedings of the Int. Joint Conference on Neural Networks*, Vol. 4, pp.2333-2338.

Acharya, U. R., Bhat, P. S., Iyengar, S. S., Rao, A and Dua, S., (2003). Classification of heart rate data using artificial neural network and fuzzy equivalence relation. *Journal of the Pattern Recognition Society*, Issue 1, Vol. 36, pp. 61-68.

Aha, D. W., (1989). Incremental instance-based learning of independent and graded concept descriptions. *Proceedings of the 6th International Workshop on Machine Learning*, Ithaca, NY, Morgan Kaufmann, pp. 387-391.

Aldrich, C. and Vandeventer, J. S. J., (1995). Comparison of different artificial neural nets for the detection and location of gross errors in process systems. *Industrial & Engineering Chemistry Research*, Vol. 34, No. 1, pp. 216-224.

Baig, M. H., Rasool, A., Bhatti, M. I., (2001). Classification of Electrocardiogram Using SOM, LVQ and Beat Detection Methods in Localization of Cardiac Arrhythmias. *report on Medicine and Medical research, report no. A749904.*

Biel, L., Pettersson, O., Philipson, L., Wide, P., (2001). ECG Analysis: A new approach in human identification. *IEEE Transactions on Instrumentation and Measurement*, Vol. 50, No. 3, pp. 808-812.

Bezdek, J. C. and Pal, N. R., (1981). Two soft relatives of learning vector quantisation. *Neural Networks*, Vol. 8, No. 5, pp. 729-743.

Bortolan, G., Degani, R. and Willems, J. L., (1991). Neural networks for ECG classification and cluster analysis. *IEEE Computer Soc. Press, Proceedings Computers in Cardiology*, pp. 177-180.

Carpenter, G. A., Grossberg, S. and Reynolds, J. H., (1991). ARTMAP: A self-organising neural network architecture for supervised learning and pattern recognition. *In. Proceeding International Joint Conference on Neural Networks, Seattle,* Vol. 1, pp. 863-868.

Chen K., Wang L., and Chi H., (1997). Methods of Combining Multiple Classifiers with Different Features and Their Applications to Text-Independent Speaker Identification. *Int. Journal of Pattern Recognition and Artificial Intelligence*, Vol. 11, No. 3, pp. 417-445.

Chen D., Odobez J.M. and Bourlard H., (2004). Text Detection and Recognition in Images and Video frames. *Pattern Recognition,* Vol. 37, Issue 3, pp. 595-608.

Conde T., (1994). Automatic neural detection of anomalies in electrocardiogram (ECG) signals. *In Proc. ICNN'94, Int. Conf. on Neural Networks*, Piscataway, N.J., pp. 3552-3558.

Cosi P., Frasconi P., Gori M., Lastrucci Land Soda G., (2000). Competitive radial basis functions training for phone classification. *Neurocomputing,* Vol. 34, Issues 1-4, pp. 117-129.

Costa, M., Dufey, J. P., Letheren, M., Marchioro, A., McLaren, R., Manabe, A., Calvert, D., Djidi, K., Ledu, P., Mandjavidze, I., Gustafsson, L., Lazrak, T., Lindblad Th., and

Tenhunen, H., (1996). The IBM zero instruction set computer ZISC036 A hardware implemented radial basis function neural network. *CRC Industrial Engineering Handbook*, Sect 7.23.

Degani R. and Bortolan G., (1990). Methodology of ECG interpretation in the Padova program. *Methods of Information in Medicine,* Vol 29, pp. 386-392.

DeMarre, D. A. and Michaels, D., (1983). Bioelectronic measurements. *Prentice-Hall, Inc,* New Jersey.

Dokur, Z., Ölmez, T., Yazgan, E. and Ersoy, O. K., (1997). Detection of ECG waveforms by neural networks. *Journal of Medical Engineering and Physics,* Vol. 19, pp. 738–741.

Dorffner, G., Leitgeb, E. and Koller, H., (1994). Towards improving exercise ECG for detecting ischemic heart disease with recurrent and feedforward neural nets. *Neural Networks for Signal Processing IV, Vlontzos, J. et al (eds.), IEEE,* New York, pp. 499-508.

Duda O. R., Hart, P.E., Stork, D.G., (2001). Pattern classification. *Wiley*, New York, ISBN 0471056693.

Edenbrandt, L., Devine, B. and Macfarlane, P. W., (1992). Neural networks for classification of ECG ST-T segments. *Journal of Electrocardiology,* Vol. 25, No. 3, pp. 167-173.

Engelbrecht, A. P., (2002). Computational Intelligence: An Introduction. *Wiley,* London, UK, ISBN 0-470-84870-7.

Errington, P. A. and Graham, J., (1993). Application of artificial neural networks to chromosomes classification. *Cytometry,* Vol. 14, No. 6, pp. 627-639.

Erwin, E., Obermayer, K., and Schulten, K., (1992). II Self-organising maps: Ordering, convergence properties and energy functions. *Biological Cybernetics,* Vol. 67, pp. 47-55.

Foody G.M., (2004). Supervised image classification by MLP and RBF neural networks with and without an exhaustively defined set of classes. *International Journal of Remote Sensing,* Vol. 25, No. 15, pp. 3091-3104

Frenster, J. H., (1990). Neural networks for pattern recognition in medical diagnosis. *Proceedings 12th Ann International Conference IEEE Eng. Medical Biological,* Soc. 12, pp. 1423-1424.

Gallant, S. I., (1988). Connectionist expert systems. *Communications of the ACM,* Vol. 31, No. 2, pp. 152-169.

Gamlyn, L., Needham, P., Sopher, S. M., and Harris, T. J., (1999). The Development of a neural network based ambulatory ECG monitor. *Neural Computing and Applications,* London, Springer-Verlag, Vol. 8, pp. 273-278.

Guyon, I., (1991). Applications of neural networks to character recognition. *International Journal of pattern Recognition and Artificial Intelligence 5,* pp. 353-382.

Hamilton, P. S. and Tompkins, W. J., (1986). Quantitative investigation of QRS detection rules using the MIT/BIH Arrhythmia Database. *IEEE Transactions on Biomedical Engineering,* Vol. 33, No. 12, pp. 1157-1164.

Hampton, J. R., (1998). The ECG Made Easy 5th edition. *Churchill Livingstone,* London.

Hara, Y., Atkins, R. G., Yueh, S. H., Shin, R. T., Kong, J. A., (1994). Application of neural networks to radar image classification. *IEEE Transactions on Geoscience and Remote Sensing,* Vol. 32, Issue 1, pp. 100 –109.

Harrison, R., Marshall, S. and Kennedy, R., (1991).The early diagnosis of heart attacks: A neurocomputational approach. *International Joint Conference on Artificial Neural Networks,* Seattle, WA, Vol. 1, pp. 1–5.

Haykin, S., (1999). Neural Networks: A comprehensive foundation. *Macmillan,* New York.

Haykin, S. and Deng, C., (1991). Classification of the rarer clustering neural networks. *IEEE Transactions on Neural Networks 2,* pp. 589-600.

Hosseini, H. and Nazeran, H., (1999). Efficient features for ANN-based ECG classifiers. *Proceedings of the Inaugural Conference of the Victorian Chapter of the IEEE Engineering in Medicine and Biology Society,* Lithgow B and Cosic I (eds) Monash University:Australia, 0-646-36946-6.

Hu, Y. H., Palreddy, S. and Tompkins, W. J., (1997). A patient adaptable ECG beat classifier using a mixture of experts approach. *IEEE Trans. On Biomedical Engineering Design,* Vol. 44, No. 9, pp. 891-900.

Huang, Y. W., Mithani, R., Takahashi, K., Fan, L. T. and Seib, P. A., (1993). Modular Neural Networks for Identification for starches in manufacturing food-products. *Biotechnology Progress,* Vol. 9, No. 4, pp. 401-410.

Hush, D. R. and Horne, B. G., (1993). Progress in supervised neural networks: What's new since Lippmann. *IEEE Signal Processing magazine,* Vol. 10, No. 1, pp. 8-39.

Inoue K. and Urahama K., (2000). Learning of view-invariant pattern recognizer with temporal context. *Pattern Recognition,* Vol. 33, Issue 10, October 2000, pp. 1665-1674.

ISL. (1998). *Clementine Data Mining Package.* SPSS UK Ltd., 1st Floor, St. Andrew's House, West Street, Woking, Surrey GU21 1EB, United Kingdom.

Izeboudjen, N. and Farah, A., (1998). A neural network system for arrhythmia's classification. *International ICSC/IFAC Symposium on Neural Computation,* Vienna, September 23-25, pp. 208-212.

Jain, A. K., Duin R., and Mao J., (2000). Statistical pattern recognition: A review. *IEEE Trans.*, PAMI, Vol: 22, pp. 4-37.

Jang, G. S., Dowla, F. U. and Vemuri, V., (1993). A comparison of neural network performance for seismic phase identification. *Journal of Franklin Institute,* Vol. 330, No. 3, pp. 505-524.

Jenkins, D., (2001). ECG library [WWW] <www.ecglibrary.com/ecghist.html> [accessed 15/07/02].

Kidwai, M. K., (2001). Modular Neural Networks and Fault Level Classifiers. *PhD Thesis,* University of Wales, Cardiff.

Klingeman J.and Pipberger H.V., (1967). Computer classification of electrocardiogram. *Computers and Biomedical Research,* Vol. 1, pp. 1-17.

Kohonen, T., (1998). The self-organising map. *Neurocomputing,* Vol. 21, Issues 1-3, pp. 1–6.

Kohonen, T., (1990). The Self-Organizing Map. *Procs. IEEE,* Vol. 78, No. 9, pp. 1464-1480.

Kohonen, T., (1989). Self-Organising and Associative Memory (3rd ed.). *Springer-Verlag* , Berlin.

Kohonen, T., (1984). Self-organization and associative memory. *Springer Verlag.*

Kohonen, T., (1982). Self-organised formation of topologically correct feature maps. *Biological Cybernetics,* Vol. 43, pp. 59-69.

Kulkarni, S.R., Lugosi, G., Venkatesh, S.S., (1998). Learning Pattern Classification - A Survey. *IEEE Transactions on Information Theory,* Vol. 44, No. 6, pp. 2178-2206.

Lippmann, R. P., (1987). An introduction to computing with Neural Nets. *IEEE ASSP magazine,* Vol. 4, pp. 4-22.

Lippmann, R. P., (1989). Pattern Classification using Neural Networks. *IEEE Communications Magazine,* Vol. 27, No 11, pp. 47-64.

Liu J. and Gader P., (2002). Neural networks with enhanced outlier rejection ability for off-line handwritten word recognition. *Pattern Recognition,* Vol.35, Issue 10, pp. 2061-2071.

Maass, W., Bishop C. M., (2001). Plused Neural Networks. *The MIT Press,* London, ISBN: 0-262-63221-7.

Maglaveras, N., Stamkopoulos, T., Diamantaras, K., Pappas , C., and Strintzis ,M., (1998). ECG pattern Recognition and classification using non-linear transformations and neural networks: A review. *International Journal of Medical Informatics,* Vol. 52, Issues 1-3, pp. 191-208.

Marzi, H., (2004) .Real-time fault detection and isolation in industrial machines using learning vector quantization. *Proceedings of the I MECH E Part B Journal of Engineering*. Vol. 18, pp. 949-959.

Mazroua, A. A., Bartnikas, R., Salama, M. M. A., (1994). Discrimination between PD pulse shapes using different neural network paradigms. *IEEE Transactions on Dielectrics and Electrical Insulation,* Vol. 1, No. 6, pp. 1119-1131.

McCulloch, K. and Bastion, K., (1999). *Curious Heart.* [WWW] <www.curiousheart.com> [accessed 26/06/02].

McLoone, S. and Irwin, G., (1998). Non-linear optimisation of RBF networks. *International Journal of Systems Science,* Vol. 29, pp. 179-189.

Merelo, J. J., Andrade, M. A., Urena, C., Prieto, A. and Moran, F., (1991). Application of vector quantisation algorithm to protein Classification and Secondary Structure Computation. *Lecture Notes in Computer Science,* Vol. 540, pp. 415-421.

Minsky, M. and Papert, S., (1969). Perceptrons An Introduction to Computational Geometry,. *MIT Press,* Cambridge.

Mitchell, T. M., (1997). Machine Learning. *McGraw-Hill,* New York, ISBN 0-07-042807-7.

Moody, J. and Darken, C., (1989). Fast learning in networks of locally tuned processing units. *Neural Computation,* 1, pp. 281-294.

Morris, R. J. T., Rubin, L. D. and Tirri, H., (1990). Neural network techniques for object orientation detection: solution by optimal feedforward network and learning vector quantisation approaches. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 12, No. 11, pp. 1107-1115.

Nabhani F. and Shaw T., (2002). Performance analysis and optimisation of shape recognition and classification using ANN. *robotics and computer-integrated manufacturing,* Vol. 18, Issues 3-4, pp. 177-185.

Negnevitsky, M., (2002). Artificial intelligence A guide to intelligent systems. *Addison Wesley,* Essex, England, ISBN 0-201-71159-1.

Nie, J. and Linkens, D. A., (1995). Fuzzy-neural control: principles, algorithms and applications. *Prentic Hall International Ltd.,* UK, Hertfordshire, ISBN 0-13-337916-7.

Nugent, C. D., Webb, J. A. C., McIntyre, M., Black N. D. and Wright, G. T. H., (1998). Computerised Electrocardiology employing bi-group neural networks. *Artificial Intelligence in medicine,* vol. 13, 167-180.

Oehler, K., Gray, R., (1995). Combining image compression and classification using vector quantisation. *IEEE Trans. Pattern Analysis and Machine Learning,* Vol. 17, pp. 461- 473.

Olmez T. and Dokur Z., (2003). Classification of heart sounds using an artificial neural network. *Pattern Recognition Letters,* Vol. 24, Issues 1-3, pp. 617-629.

Olsson, J. and Gruber, S., (1993). Web process inspection using neural classification of scattering light. *IEEE Transactions on Industrial Electronics,* Vol. 40, No.1, pp. 228-234.

Osowski S. and Nghia D. D., (2002). Fourier and wavelet descriptors for shape recognition using neural networks—a comparative study. *Pattern Recognition,* Vol.35, Issue 9, pp. 1949-1957.

Otas J. C., Bernstein L., Bailey B. P., Freedman, S.B., (1988). Real time of ischemic ECG changes using quasi-orthogonal and artificial intelligence. *Proceedings of the real-time ECG rhythm analysis. IEEE transaction on Biomedical Engineering BME-15,* pp. 128-129.

Pal, S. K. and Mitra, S., (1996). Noisy fingerprint classification using multilayer perceptron with fuzzy geometrical and texture features. *Fuzzy Sets and Systems,* Vol. 80, pp. 121-132.

Pan, H. L. and Chen, Y. C., (1992). Liver tissues classification by artificial neural networks. *Pattern Recognition Letters,* Vol. 13, No. 5, pp. 355-388.

Pandya, A. S. and Macy, R. B., (1996). Pattern Recognition with Neural Networks in C++. *IEEE press,* Florida, USA, ISBN 0-8493-9462-7.

Pedrycz W., Bortolan G., Degani R., (1991).Classification of Electrocardiographic signals: a fuzzy pattern matching approach. *Artificial Intelligence in Medicine,* Vol. 46, pp. 3-31.

Peng, Y. and Reggia, J. A., (1989). A connectionist model for diagnostic problem solving. *IEEE Trans. on System, Man and Cybernetics,* Vol. 19, pp. 285-298.

Pham, D. T. and Afify, A. A. (2002). Machine learning: Techniques and trends. *Proc. of the 9th International Workshop on Systems, Signals and Image Processing (IWSSIP – 02),* Manchester Town Hall, UK, World Scientific, pp. 12-36.

Pham, D. T. and Sagiroglu, S., (2001). Training multilayered perceptrons for pattern recognition: a comparative study of four training algorithms. *International Journal of Machine Tools and Manufacture,* Vol. 41, Issue 3, pp. 419-430.

Pham, D. T. and Liu X., (1995). Neural networks for identification, prediction and control. *Springer-Verlag,* London.

Pham, D. T. and Oztemel, E., (1994). Control chart pattern recognition using learning vector quantisation networks. *International Journal of Production Research,* Vol. 32, No. 3, pp. 721-729.

Pham, D. T. and Oztemel, E., (1993). Combing Multi-layer perecptrons with heuristics for reliable control chart pattern recognition. *Eighth International Conference on Applications of Artificial Intelligence, Computational Mechanics*, Southampton, pp. 801-810.

Pipberger, H. V. and Stallman F. W., (1962). Use of computers in electrocardiogram interpretation. *American Heart Journal 64,* pp. 285-286.

Pordy L., Jaffe H., and Chesky, K., (1968). Computer diagnosis of the electrocardiograms IV a computer program for contour analysis with the clinical results of the rhythm and contour interpretation. *Computers and Biomedical Research*, Vol.1, pp. 408-433.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, Vol. 1, pp. 81-106.

Quinlan, J. R. (1989). Unknown attribute values in induction. *Proc. of the 6th International Workshop on Machine Learning,* Ithaca, New York, Morgan Kaufmann, pp. 164-173.

Quinlan, J. R., (1993). C4.5: Programs for machine learning. *Morgan Kauffman,* San Mateo, California.

Rasiah, A. I. and Attikiouzel, Y., (1994). A syntactic approach to the recognition of common cardiac arrhythmia within a single ambulatory ECG trace. *Australian Computer Journal,* Vol. 26, No. 3, pp. 102-112.

Renals, S., Morgan, N., Cohen, M., Franco, H. and Bourlard, H., (1992). Improving statistical speech recognition. *International Joint Conference on Neural Networks,* Vol. 2, pp. 301-307, Baltimore MD.

Rosenblatt, F., (1958). The perceptron: A probabilistic model for information storage and organisation in the brain. *Psychological Review,* Vol. 65, pp. 386-408.

Rumelhart, D. E., McClelland, J. L., and the PDP Research Group, (1987). Parallel Distributed Processing: Explorations in the Microstructure of cognition. *The MIT Press,* Cambridge, Massachusetts, Vol. 1, pp. 151-193.

Schlkopf, B., Smola, A. J., (2002). Learning with kernels: support vector machines, regularization, optimization, and beyond. *MIT Press,* Cambridge, ISBN:0-262-19475-9.

Shoukry, S., Martinelli, D., Srinivas, T. and Halabe, U., (1996). Radar signal interpretation using neural networks for defect detection in concrete. *Materials Evaluation,* Vol. 54, No. 3, pp. 393-397.

Silipo, R. and Bortolan, G., (1997). Neural and Traditional Techniques in Diagnostic ECG Classification. *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing,* pp. 123-126.

Silipo, R., Bortolan, G. and Marchesi, C., (1999). Design of hybrid architectures based on neural classifier and RBF pre-processing for ECG analysis. *International Journal of Approximate Reasoning,* Vol. 21, pp. 177-196.

Sejnowski, T. J. and Rosenberg, C.R., (1987). Parallel network that learn to pronounce English text. *Complex Systems,* Vol. 1, No. 1, pp. 145-168.

Sultan, A. F., Swift, G. W., and Fedirchuk, D. J., (1992). Detection of High Impedance Arcing Faults Using a Multi-layer Perceptron. *IEEE Trans, on Power Delivery,* Vol. 7, No. 4, pp. 1871-1877.

Suzuki, Y., (1995). Self-organising QRS-wave recognition in ECG using neural networks. *IEEE Transactions on Neural Networks,* Vol. 6, No. 6, pp. 1469-1477.

Svardstrom, A., (1993). Neural networks features vectors for sonar targets classification. *Journal of the Acoustical Society of America,* Vol. 93, No. 5, pp. 2656-2665.

Texas Heart Institute, 2002. [WWW] <www.tmc.edu/thi/anatomy.html> [accessed 2/5/02]

Therrien, C.W., (1989). Decision estimation and classification: An introduction to pattern recognition and related topics. *Wiley & Sons Inc,* Toronto, Canada, ISBN 0-471-83102-6.

Titomir, L. I., (2000). The remote past and near future of electro cardiology: Viewpoint of a biomedical engineer. *Bratislava Medical Journal,* Bratisl Lek Listy, Vol. 101 issue 5, pp. 272-279.

Totton, K. A. E. and Limb, P. R., (1991). Electronic diagnosis using multilayer perceptron. *BT Technology Journal,* Vol. 10, No. 3, pp. 97 –102.

Tsujii, O., Freedman M.T.and Mun S. K., (1999). Classification of microcalcifications in digital mammograms using trend-oriented radial basis function neural network. *Pattern Recognition,* Vol. 32, Issue 5, pp. 891-903.

Utgoff, P., Berkman, N., and Clouse J., (1997). Decision tree induction based on efficient tree restructuring. *Machine Learning*, Vol. 29, pp. 5-44.

Veselis, R. A., Reinsel, R. and Wronski, M., (1993). Analytical methods to differentiate similar Electroencephalographic spectra: neural network and discriminant analysis. *Journal of Clinical Monitoring,* Vol .9, No. 4, pp. 257-267.

Venkatesh, Y. V., and Raja, S. K., (2003). On the classification of multispectral satellite images using the multilayer perceptron. *Pattern Recognition,* Vol. 36, Issue 9, pp. 2161-2175.

Wahl, M., (1999). The heart is a muscle, too frequently asked questions about cardiac problems in neuromuscular disease. *Quest,* Vol. 6, No. 3, Available online [WWW] at <www.mdausa.org/publications/Quest/q63cardiac.html> [accessed 12/08/02].

Webb, A., (1999). Statistical pattern recognition. *Arnold,* London, UK, ISBN 0-340-7464-3.

Weisner, S., Tompkins, W. J. and Tompkins, B., (1982). A campact, microprocessor-based ECG ST-segment analyser for the operating room. *IEEE Trans. Biomed. Eng.,* Vol. 29, pp. 642-649.

Wilkins, M. F., Boddy, L. and Morris, C. W., (1994). Kohonen maps and learning vector quantisation neural networks for analysis of multivariate biological data. *Binary Computing in Microbiology,* Vol. 6, pp. 64-72.

Wu, K. L. and Yang, M. S., (2002). A fuzzy-soft learning vector quantisation. *International Journal of Production Research,* Vol. 40, No 12, pp. 2721-2731.

Wuxing L., Tse P. W., Guicai Z, Tielin S., (2004). Classification of gear faults using cumulants and the radial basis function network. *Mechanical Systems and Signal Processing,* Vol. 18, No. 2, pp. 381-389.

Yang B. S., Han T., An J. L., (2004). ART-KOHONEN neural network for fault diagnosis of rotating machinery. *Mechanical Systems and Signal Processing,* Vol. 18, No. 3, pp. 645-657.

Yang, M. S. and Yang J. H. (2002). A fuzzy-soft learning vector quantization for control chart pattern recognition. *International Journal of Production Research,* Vol. 40, No.12, pp. 2721-2731.

Yeap, T. H., Johnson, F. and Rachniowski, M., (1990). ECG beat classification by a neural network. *IEEE Engineering in Medicine and Biology Society Proceedings of the 12th Annual International Conference,* Philadelphia, Pennsylvania, USA, Vol. 3, pp. 1457-1458.

Zurada, J. M., (1992). Introduction to neural systems. *West Publishing Company,* New York, ISBN 0-314-93391-3.