

# Towards Automatic Face Recognition Using Discrete Cosine Transforms and Neural Networks

Sanjeev Chundurduth Debipersad BScEng

Submitted in fulfillment of the academic requirements for the degree  
of Master of Science in Engineering in the Department of Electronic  
Engineering, University of Natal.

Durban

June 1998

## Abstract

---

This thesis describes the development of a model that simulates the human recognition of a face by a machine in real time. The need for such research is a result of the increasing fraudulent behaviour in society with the result that there is a renewed interest in the collection of biometric measures to strengthen identity checks. Various attempts have been made to solve the problem of face recognition using neural networks, algebraic moments, elastic template matching, Karhunen-Loeve Transforms, eigenfaces and Hidden Markov Models. Varying degrees of success have been reported. The method proposed in this thesis is simple, and inexpensive to implement.

The system is based upon capturing an image of a subject. Psychophysical observations are used in the image capture process by using  $\frac{3}{4}$  views of the subjects. A general symmetry transform was initially used for the normalisation process whereby only the face of the subject is evaluated. Although this was found to be very accurate it is computationally intensive and manual normalisation was used.

Compression of the facial images is affected using a 2-D discrete cosine transform. Since the variance of each element of the transformed image represents the information content, transform coefficients with larger variances are selected.

Two different neural networks are used. The standard back-propagation neural network is used in a "network per person" implementation, while the counter-propagation and radial basis function network are used in a database implementation. The database was split and tests were performed with the two new databases to determine the generalisation ability of the neural networks. The results of the classification process are discussed and proposals are made for incorporating this model into a face recognition unit for the prevention of fraud.





*The bird with you, the wings with Me;  
The foot with you, the way with Me;  
The eye with you, the form with Me;  
The thing with you, the dream with Me;  
So are we free, so are we bound;  
So we begin and so we end;  
You in Me and I in you.*

**- BABA**

*This thesis is dedicated to Bhagawan Sri Sathya Sai Baba, who is Aartha Traana Parayanaaya (He who comes gladly to the rescue of those in agony), the Prime Mover of the phenomenal Universe and who manifests as Jnaana Swaroopini Saraswathie Ma.*

## **Preface**

---

The work and scientific experiments presented in this thesis was carried out from January 1996 to December 1997 in the Department of Electronic Engineering at the University of Natal, under the supervision of Professor A. D Broadhurst. Unless otherwise stated in the main body of the thesis, all work presented in the thesis is my own and has not been submitted in part, or in whole to any other university.

## Acknowledgements

---

The author would like to thank the following person/people/organisations and institutions:

- My mother for so patiently bearing with me and without whom there would be no me.
- My brother Pramodh and sister Ramola for believing in me and encouraging me every step of the way.
- Professor A.D Broadhurst for his supervision during this study.
- Bashan Naidoo for providing invaluable advice, inspiration and for those much meaningful talks on the philosophy of life and the thesis.
- Rakhee Gobind for so diligently reading through this thesis.
- First National Bank for their interest and grant, without which this work would not have been possible.
- University of Natal for supporting this study academically and financially.
- The authors of SNNS for providing an excellent neural network package.
- All subjects who agreed to the use of their images in this study.

# Table of Contents

---

## Chapter 1

### Introduction

1.1	Human Vision .....	1-1
1.2	The Need for Biometric Measures .....	1-2
1.3	Difficulties Associated with Automatic Face Recognition .....	1-3
1.4	Objectives and Limitations to this thesis .....	1-3
1.5	Structure of this thesis .....	1-5

## Chapter 2

### Review of Face Recognition

2.1	Introduction .....	2-1
2.2	Psychology of Face Recognition .....	2-1
2.2.1	The $\frac{3}{4}$ Face View .....	2-2
2.3	Survey of Relevant Literature .....	2-4
2.3.1	Face Location .....	2-6
2.3.2	Face Recognition .....	2-10
2.3.3	Neural Network Approaches .....	2-18
2.3.4	Basis of this study .....	2-22
2.3.4.1	Grey Scale Projection Model .....	2-23
2.3.4.2	Appraisal of Bouwer's technique .....	2-32

## Chapter 3

### Overview of the System and its Components

3.1	Introduction .....	3-1
3.2	A Two-Dimensional Approach .....	3-1
3.2.1	Video Camera .....	3-2
3.2.2	Frame Grabber .....	3-2
3.2.3	Face Location .....	3-3
3.2.4	2-D Discrete Cosine Transform .....	3-3
3.2.5	Classification Process .....	3-3
3.3	Image Capture .....	3-4
3.3.1	Experimental Setup .....	3-4
3.3.2	Examples of Images Captured.....	3-11

## Chapter 4

### Face Preprocessing

4.1	Introduction .....	4-1
4.2	Face Location .....	4-1
4.2.1	The General Symmetry Transform .....	4-4
4.2.2	Definition of the General Symmetry Transform .....	4-6
4.2.3	Performance of the General Symmetry Transform .....	4-10
4.2.4	Manual Normalisation .....	4-13
4.3	The Discrete Cosine Transform .....	4-15
4.3.1	Motivation for use of the Discrete Cosine Transform.....	4-16
4.3.2	The 2-D Discrete Cosine Transform .....	4-17
4.3.3	2-D DCT Implementation by reduction to a 1-D DCT .....	4-18
4.3.4	Results of the 2-D Discrete Cosine Transform .....	4-20
4.3.5	Choosing the DCT Coefficients .....	4-24

## **Chapter 5**

### **Classification Technique**

5.1	Introduction .....	5-1
5.2	What is a Neural Network .....	5-2
5.2.1	Processing Elements .....	5-3
5.2.2	Network Topologies .....	5-6
5.2.3	Weights .....	5-7
5.3	Training of Artificial Neural Networks .....	5-8
5.4	The Back-propagation Neural Network .....	5-8
5.5	The Counter-propagation Neural Network .....	5-10
5.6	The Radial Basis Function Neural Network .....	5-16
5.6.1	The Dynamic Decay Adjustment Algorithm .....	5-21
5.7	About SNNS .....	5-24

## **Chapter 6**

### **Results and Discussion**

6.1	Introduction .....	6-1
6.2	Some Definitions .....	6-1
6.2.1	Biometric System Performance .....	6-2
6.3	Database Approach .....	6-2
6.3.1	Training and Testing Sets for the Database Approach .....	6-3
6.4	The Radial Basis Function Neural Network .....	6-12
6.4.1	Tests with unnormalised data .....	6-15
6.4.2	Tests with unnormalised data and split databases .....	6-20
6.4.3	Tests with normalised data .....	6-34
6.4.4	Tests with normalised data and split databases .....	6-37
6.5	The Counterpropagation Neural Network .....	6-45
6.6	Network per Person Approach .....	6-48

## Chapter 7

### Conclusion and Recommendations

7.1	Conclusion .....	7-1
7.2	Recommendations .....	7-7

<b>References</b> .....	<b>R-1</b>
-------------------------	------------

<b>Appendix A</b> .....	<b>A-1</b>
-------------------------	------------

<b>Appendix B</b> .....	<b>B-1</b>
-------------------------	------------

<b>Appendix C</b> .....	<b>C-1</b>
-------------------------	------------

<b>Appendix D</b> .....	<b>D-1</b>
-------------------------	------------

<b>Appendix E</b> .....	<b>E-1</b>
-------------------------	------------

<b>Appendix F</b> .....	<b>F-1</b>
-------------------------	------------

## List of Acronyms and Abbreviations

---

1-D	-	One Dimensional
2-D	-	Two Dimensional
AFR	-	Automatic Face Recognition
BP	-	Back-Propagation
CN	-	Convolutional Network
CP	-	Counter-Propagation
DA	-	Database Approach
DCT	-	Discrete Cosine Transform
DFT	-	Discrete Fourier Transform
FAR	-	False Acceptance Rate
FRA	-	Face Recognition Algorithm
FRR	-	False Rejection Rate
FSE	-	Flat Spot Elimination
GST	-	General Symmetry Transform
HMM	-	Hidden Markov Model
KLT	-	Karhunen-Loeve Transform
MDM	-	Mahalanobis Distance Metric
MIT	-	Massachusetts Institute of Technology
MLP	-	Multi-layer Perceptron
MSE	-	Mean Squared Error
MTE	-	Maximum Tolerable Error
MTNRR	-	Mean True Negative Recognition Rate
MTPRR	-	Mean True Positive Recognition Rate
NN	-	Neural Network
NPPA	-	Network per Person Approach
ORL	-	Olivetti Research Laboratory



Pels	-	pixels
RBFN	-	Radial Basis Function Network
RBFN-DDA	-	Radial Basis Function Network with Dynamic Decay Adjustment
SNNS	-	Stuttgart Neural Network Simulator
SOM	-	Self Organising Map
SSE	-	Sum Squared Error

# Chapter 1

## INTRODUCTION

---

It is indeed a great boon to see the myriad animate and inanimate objects of this universe. The ability to see, process what is seen and arrive at relevant decisions has determined to a large extent the social behaviour of humans and animals. Human vision develops over several years of childhood and together with related abilities has played an important role in the course of evolution[3].

### 1.1 Human Vision

One of the most remarkable abilities of human vision is face identification/recognition. According to *Reisfeld*[7], face recognition can be defined as the classification of a face image as belonging to a specific individual or a class (e.g. gender). Classification can be defined as either the positive or negative verification of an individual. Facial processing also consists of understanding of expressions in the context of emotion or communication; social attention - evaluating gaze and face direction; recognising facial qualities such as beauty, age, sex and character; lip reading and other facial movements. Without the skill responsible for the identification of a familiar face in a group of strangers or the ability to discriminate between animate and inanimate objects, humans would be at a functional loss[1].

## 1.2 The need for biometric measures

The increase of fraudulent behaviour in society has caused many professional institutions to turn to biometric<sup>1</sup> measures for positive identification of their clients. There is thus a renewed interest in the collection of biometric measures of people to strengthen identity checks[8]. The collection of biometric measures are used in biometric systems which are (usually) automated devices used for verifying or recognising the identity of a living person on the basis of a physiological characteristic like fingerprinting, iris images or some aspects of behaviour like handwriting or key-stroke patterns[29].

Many institutions, including the FBI have carried out research into the field of fingerprint recognition. The basis for this research is that fingerprints are unique to each human being. The iris of the eye, much like fingerprints exhibits unique characteristics with respect to texture and patterns. Research has shown that these characteristics remain stable for decades. The use of speech is another biometric measure that has been used to identify the speaker. However, all these measures are in some way obtrusive[8].

Perhaps the most passive method of identifying an individual is by face recognition. An automatic face recognition system would allow a subject to be identified by taking an image of that subject without the subject's knowledge of the event. Thus, the need to unobtrusively identify clients via biometric measures warrants further investigation into the problem of automatic face recognition.

Automatic face recognition is defined as the machine recognition of human faces. It was considered in the early stages of computer vision[2]. Another reason for its revival, after nearly two decades, could be attributed to the increase in computational power, which enables effective implementation of algorithms[4]. Automatic face recognition has several applications which range from access control, credit card verification, user authentication, surveillance of video images and monitoring of patients for post operative swelling[5].

---

<sup>1</sup> Biometric measures are based on physiological or behavioural characteristics.

### **1.3 Difficulties Associated with Automatic Face Recognition**

The robustness of human vision is demonstrated in their ability to identify subjects in quite adverse conditions. They are able to account for changes in lighting, pose, expression, occluded detail (a beard or glasses) and even hairstyles.

Though the process of face identification has become quite a natural task for humans, it is one of the most difficult things we do. The difficulty can be attributed to the fact that all faces contain roughly the same geometric arrangement (extreme homogeneity of the faces). Non-rigidity of faces (seen in facial expressions) in addition to imaging conditions such as lighting or pose compounds the problem[6].

Various attempts have been made to solve the problem of automatic face recognition. Techniques as varied as the implementation of neural networks, elastic template matching, Karhunen-Loeve transforms, algebraic moments and isodensity line maps have been proposed[5]. None of the above-mentioned techniques have been totally successful. In just a glance, humans can determine a person's identity, race, sex, age and physical expressions. Unfortunately most computer models of face recognition are not capable of this.

### **1.4 Objectives and Limitations to this Thesis**

Most face recognition systems employ either geometric features or template matching. This study focuses on a technique based on a global or holistic approach. A new database of faces containing  $\frac{3}{4}$  view information is built whereby the whole face is used in the recognition process as opposed to a collection of features (e.g. nose, eyes, distance from nose to eyes etc.). Captured images of individuals contained full frontal views, in-between full frontal views and  $\frac{3}{4}$  views, and  $\frac{3}{4}$  views. After capturing the individual's image, it is normalised.

The 2-D discrete cosine transform (DCT) is then applied to the normalised image. This results in a transformed image in which most of the energy of the coefficients is concentrated in the upper left-hand corner of the transformed matrix. The coefficients with the greatest variance were selected in two phases:

- a) Visual inspection - this was undertaken to determine the area of search for coefficients with high variance.
- b) Neural networks - tests were performed with neural networks to determine which area gave the optimum performance with respect to the classification rate.

Neural networks have been used with success in fields as diverse as prediction and classification. The ability of the neural network to generalise<sup>1</sup> prompted many researchers to use them in image recognition applications with varying degrees of success[6]. They were thus chosen for the classification/verification of the faces. The neural networks are deployed in two specific fashions. A standard back-propagation network with momentum is used to train a network per person while a counter-propagation and a radial basis function network are used in a database approach. The two approaches were used independently of each other, and no hybrid training/network was used.

---

<sup>1</sup> A neural network is said to be able to generalise when it can correctly classify data that it has not seen during its training phase[10].

As this study forms an initial investigation for future work, certain assumptions were made:

The thesis assumes that the face presented to the discrete cosine transform is already normalised. Automatic face location was attempted via the general symmetry transform, however it was found that due to the computational intensity required by the transform, it was not a viable solution. Hence manual normalisation was performed on the images.

It must be emphasised that this study concentrates on the investigation into whether 2-D discrete cosine transforms and neural networks are a suitable solution for the problem of automatic face recognition.

The definition of what constitutes acceptable performance in a face recognition system is dependent on the context of the application. However, it should be agreed upon that speed of recognition/rejection as well as recognition rate should be high. The term "high" is relative. For the purposes of this study, high speed of recognition is defined to be within 8s (i.e. the image capture, face location and recognition should take place in 8s). Current technology prevents the achievement of a 100% recognition rate. A true positive recognition rate of above 90% and true negative recognition rate of above 95% is defined as acceptable. (The definitions of true positive and true negative recognition rates are dealt with in Chapter 6).

## **1.5 Structure of this thesis**

**Chapter 2** surveys related work in the field of automatic face recognition. Important psychophysical findings that impact on the current study are described first, followed by reports on automatic face location, automatic face recognition and the application of neural networks to automatic face recognition. The basis of this study, as well as the limitations of the previous work are also described.

**Chapter 3** - describes the system components that make up the face recognition algorithm. An overview of the algorithm is given, and the details of the experimental set-up used to capture the images are described.

**Chapter 4** - presents the general symmetry transform that was investigated for use in automatic face location. The 2-D discrete cosine transform is defined and the motivation for using it is then described. The variance of the coefficients of the 2-D discrete cosine transform is also reported on.

**Chapter 5** - is a summary of neural networks. The back-propagation network is mentioned while the counter-propagation and radial basis function network is explained in detail.

**Chapter 6** - describes experimental results using different sized, as well as different shape regions that were presented to the neural network. This was done to determine the optimum region size and shape to be presented to a neural network. The insights gained from these results were used in further experiments.

**Chapter 7** - concludes the thesis by summarising the results obtained and suggesting further avenues of research for the study, as well as possible implementation scenarios.

**Appendix A** - contains a description of the code used for the frame grabber card.

**Appendix B** - describes the face database that was built.

**Appendix C** - describes the code that was written for the general symmetry transform.

- Appendix D** - details the results of extensive tests performed with the radial basis function network (with the dynamic decay algorithm).
- Appendix E** - reports the results obtained from tests performed with the back-propagation network with momentum.
- Appendix F** - reports the results obtained from tests performed with the counter-propagation network.
- Appendix G** - contains recommendations for further work, as well as plans for a prototype AFR system.



## Chapter 2

# REVIEW OF FACE RECOGNITION

---

### 2.1 Introduction

Automatic face recognition has over the years drawn the interest of researchers in the fields of psychophysics, neural sciences, engineering, image processing and analysis and computer vision[30]. This chapter reviews some of the most current research in automatic face recognition. It must be emphasised that this review is by no means a complete review of the field.

### 2.2 Psychology of face recognition

The human face recognition faculty receives stimuli that are gathered from the senses and which are used for both the storage and retrieval of face images for face recognition. The view held by *Chellapa et al*[30] is that it is impossible (with present technology) to design an automatic face recognition system that would be able to fully mimic the human face recognition faculty. However, the latter[30] maintains that the advantage of an automatic face recognition system is that it would be able to handle large amounts of faces, as opposed to the human brain that can identify a finite number of faces.

Psychologists are interested in the cognitive mechanisms of face recognition. A detailed review of the relevant studies in psychophysics and neuroscience is beyond the scope of this study. Studies done in the fields of psychophysics and neurophysiology show that

both holistic and feature information are crucial for the perception and recognition of faces. The hair, face outline and mouth have been determined as being important in the face recognition process. It has also been determined that the nose plays an insignificant role in the face recognition process. This could be due to the fact that only frontal views were used in the study[30].

Other issues that psychophysicists and neuroscientists have been concerned with are

- Uniqueness of faces;
- Whether face recognition is done holistically or by local feature analysis;
- Analysis and use of facial expressions for recognition;
- How infants perceive faces; organisation of memory for faces;
- Inability to accurately recognise inverted faces;
- The role of the right hemisphere of the brain in face perception and inability to recognise faces due to conditions such as prosopagnosia<sup>1</sup>[30].

Some of the theories that have been put forward by psychophysicists and neuroscientists to explain observed experimental results are contradictory[30]. However, some of their findings have important consequences for engineers who design algorithms and systems for the machine recognition of human faces. One such finding is the importance of the  $\frac{3}{4}$  face view in its relation to face recognition.

### 2.2.1 The $\frac{3}{4}$ Face View

*Perret et al* [25] suggested the presence of individual cells of the macaque superior temporal sulcus of monkey's brains appeared to be exclusively tuned to respond to specific views of a head. Most of the cells were viewer centred and responded to one view (either the frontal, the two profiles or the back views). Few cells were tuned to

---

<sup>1</sup> Prosopagnosia is a variety of visual agnosia characterised by inability to recognise the faces of other people, or even one's own face in a mirror, associated usually with agnosia for colour, objects and places. Agnosia is defined as the loss of the power to recognise the import of sensory stimuli; the varieties correspond with the several senses.

other views in the 360° range. This work provided some neurophysical basis for effects of pose change and a  $\frac{3}{4}$  view<sup>2</sup> advantage[24].

*Bruce et al*[20] performed experiments to test the postulate: three-quarter views promote better recognition memory for previously unfamiliar faces than do full-face views. Tests were performed using  $\frac{3}{4}$  views and full face views. The psychophysical observations showed

- no evidence of the advantage of the  $\frac{3}{4}$  view over the full face view when the faces used were highly familiar to the subjects (psychology students who were asked to identify the faces)
- $\frac{3}{4}$  views led to increased speeds of recognition of the same faces when shown to subjects to whom the faces were unfamiliar.

According to *Bruce et al*[20], the  $\frac{3}{4}$  view advantage appeared when the faces were unfamiliar and the task had to be performed at the level of visual matching. It was reiterated that the  $\frac{3}{4}$  view does not function as a “canonical view” in the representation of familiar faces.

*Schyns and Bulthoff*[24] also investigated the condition of viewpoint dependence of face recognition in humans. Five views were used in the experiment (frontal view, between frontal and  $\frac{3}{4}$  view). In all the tests performed, the highest classifications were obtained for the  $\frac{3}{4}$  views. The data revealed a strong interaction between the learned view of a face and the generalisation to other views of the same face. The experiments show that no single view was canonical. Results did however show that face recognition could be performed from a non-singular view, and that the  $\frac{3}{4}$  view should be preferred over a full-face view because it allows better encoding and recognition. It was reported that among all views, the  $\frac{3}{4}$  view is identified fastest and with greatest accuracy.

Although some of the research done in the field of psychophysics and neuroscience serves as a useful guide, one must be prudent in using only those theories that are applicable or relevant from a practical /implementation point of view[30].

---

<sup>2</sup>  $\frac{3}{4}$  view – the viewpoint between full face and profile views

## Survey of relevant literature

*Chellapa et al*[30] did an extensive survey on the applications, advantages and disadvantages of face recognition. They categorised the different recognition systems as follows:

- Matching - defined as matching one face image to another face image. (see applications 1-3 of Table 2.1)
- Similarity and detection - defined as finding or creating a face image, which is similar to a human recollection of a face. (see applications 4-6 of Table 2.1)
- Transformation - defined as generation of an image of a face from input data that is useful in other applications. The information is used to perform modifications on a face image to arrive at the relevant image. The reconstruction of a face of a mummy from its remains is such an example. (see applications 7-9 of Table 2.1)

**Table 2. 1** : Applications of face recognition technology (reproduced from *Chellapa* [30])

Application	Advantages	Disadvantages
1a. Credit card, drivers license, passport and personal identification. 1b. Mugshot matching	Controlled image Controlled segmentation Good quality	No existing database Large potential database Rare search type
2. Bank/Store security	High value Geographic search limits	Uncontrolled segmentation Low image quality
3. Crowd surveillance	High value Small file size	Uncontrolled segmentation Low image quality Real time
4. Expert Identification	High value Enhancement possible	Low image quality Legal certainty required

5. Witness Face Reconstruction	Genetic optimisation	Unknown similarity
6. Electronic Mugshot Book	Descriptor search limits Genetic optimisation	Unknown similarity
7. Electronic Lineup	Descriptor search limits	Unknown similarity
8. Reconstruction of face from remains	High value	Requires physiological input
9. Computerised Ageing	Missing children	Requires example input

For the purposes of this study, only “matching” is considered.

### 2.3.1 Face location

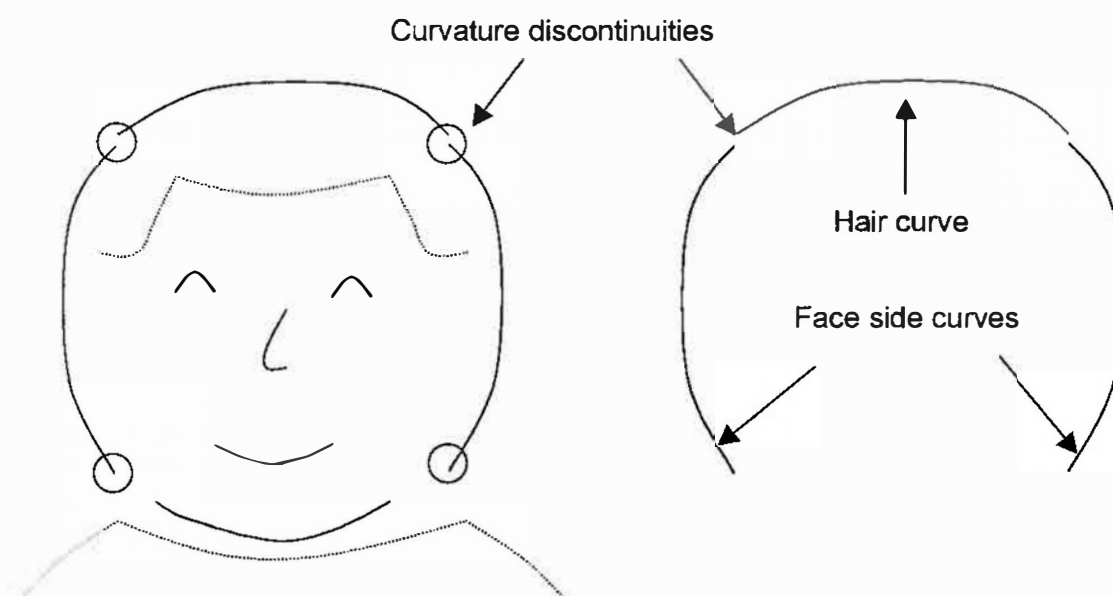
Face location is a difficult, yet important first step in any fully automatic face recognition system. The location of faces is a non-trivial task and deserves a complete study devoted to it. **In this study emphasis is placed on the recognition aspect of the automatic face recognition system, however, face location is attempted by use of the general symmetry transform.**

*Sung and Poggio*[33] define face detection as: given an arbitrary input image that could be digitised or scanned in, determine whether or not there are any human faces in the image, and if there are return an encoding of the location and spatial extent of each human face in the image. An example encoding is defined as probably fitting each face in the image with some bounding box. If the description of such an action is so simple, then one is justified in asking why face detection is so difficult. *Sung and Poggio*[33] give the following reasons for the difficulties experienced in face detection:

- Although most faces are similar in structure and appearance (i.e. the same facial features arranged in roughly the same spatial configuration), it is possible that there can be a large component of non-rigidity and textural differences among faces. The size of a person's nose, colour of a person's skin, and distance between the two eyes are some factors that contribute to the variability of faces. Thus, traditional fixed template matching techniques and geometric model-based object recognition approaches that work well for rigid objects do not perform as well for the detection of faces.
- Glasses or moustaches can be either present or absent from a face, thus increasing the variability that a comprehensive face detection system must handle.
- Unpredictable imaging conditions in an unconstrained environment can enhance the variability among faces, e.g. a change in light source can remove significant shadows from a particular face.

Some automatic face recognition systems employ constraints on the location of the head within the image for easier face location. In the *Wilder*[9] face recognition system, subjects were asked to place their chin on a chin rest and look into a window in a booth containing a camera, light source, mirror and monitor. During training the subjects observed their image in the monitor and aligned their faces roughly within a box appearing on the monitor. The camera and mirror produced a full-face image of the subject. The light source was placed off-axis to create shadows that de-lineated facial features.

A much more sophisticated model based approach is presented by *Govindaraju et al*[35] for the location of human faces in newspaper photographs. The shape of the face was modelled via a series of templates. The templates were composed of three segments that are obtained from the curvature discontinuities of the head outline. These are abstract descriptions of simple geometric features such as lines, arcs and edges. The relationships between templates are represented by springs that are pulled and pushed to align the template with the original image. The constraints placed by *Govindaraju et al*[35] are that the faces must be front view, upright, not occluded, with contrast against the background and their sizes should be greater than an absolute minimum and less than an absolute maximum. Their system was able to find the only face in the image 83% of the time.



**Figure 2. 1 :** Model of the shape of a human face used by *Govindaraju et al*[35].  
[Reproduced from [37]]

*Craw et al*[34] used a multi-resolution approach that employed templates of 8x8, 16x16, 32x32, 64x64 and 128x128 (full size image). Their initial processing starts at the lowest resolution and averages the full size image to 8x8 pixels. A model head was used at different resolutions iteratively to handle scaling. Their algorithm calculated the edge and magnitude direction after applying a Sobel mask to the original grey-scale image. The head outline was obtained from the lowest resolution image by joining the locations of the maximum gradients. They used the results obtained at lower resolutions as guidelines for the next higher resolution. After the head outline was located, a search for features (eyes, ears, nose, and lips) was done. Their algorithm was tested on 20 faces, and the features that were identified are reflected in Table 2.2.

**Table 2. 2 :** Table of features detected

Feature	Number of times detected
Outlines	12
Lips	19
Eyebrows	27
Eyes	10



Neural networks have also been employed in the task of face location. *Baluja et al*[36] presented a neural network based algorithm to detect frontal views of faces in grey-scale images. They used a neural network based filter that received as input a small square region and generated an output that ranged from 1 to -1, signifying the presence or absence of a face respectively. For the filter to detect faces anywhere in the input, it must be applied at every location in the image. If the faces are larger than the window size, the input image is reduced in size and the filter is applied at each size. Before the window is passed to the neural network, pre-processing is performed on it. The pre-processing stage consists of histogram equalisation, which non-linearly maps the intensity values to expand the range of intensities in the window. The pre-processed window is passed through a neural network whose single valued output indicates whether the window contains a face or not. The network itself has “retinal connections” to its input layer. *Baluja et al*[36] used two and three sets of hidden units. The detector is enhanced by merging overlapping detections from a single network and arbitrating among multiple networks. The algorithm detected up to 92.9% of the faces in a set of test images. Their results are impressive, however the algorithm has problems in detecting faces with glasses. This could possibly be due to the fact that the glare from the lens of the glasses occludes the details of the eyes (i.e. the eye region is not dark enough) [36].

Faces can also be located by first locating important features such as the eyes, nose or mouth. One can then exploit the geometric information of the face to locate the face. According to *Kokuer*[37], two types of features are commonly used:

1. Features that are derived from the profile of the image, e.g. the notch between the brow and the nose, the nose and the upper lip, and the tip of the nose and the tip of the chin.
2. Features that are derived from intensity images, such as the size of the eyes, inter-ocular distance and the distance between the eyes and lips, hair or cheek intensity.

*Kokuer*[37] applied a valley edge detector to full face images of 360x288 pixels. The valley edge detector was used as opposed to others since it yields edges that tend to match with those that humans find important. A series of horizontal slits were applied to the images and vertical profiles from these slits were obtained. *Kokuer*[37] was able to locate the eyes from the vertical projections of the horizontal slits. It was not clear which horizontal slits should be chosen over others, where these slits should be placed, as well as the methodology used for the detection of the peaks from the vertical profiles.

### 2.3.2 Face recognition

Faces represent complex, multidimensional, meaningful visual stimuli and developing a computational model for face recognition is difficult [41].

*Brunelli and Poggio*[2] identified two general strategies that have evolved over the years as possible solutions to the problem of face recognition:

1. **Geometric Feature Based Matching:** In this approach a face can be recognised even when the details of individual features are no longer resolved. Relative positions as well as parameters such as the distinctiveness of the eyes, mouth, nose and chin are extracted from the remaining data, which is essentially geometrical in nature. They also state that the very fact that face recognition is possible at coarse recognition implies that the overall geometrical configuration of the face is sufficient for face recognition[2].
2. **Template Matching:** In a simple version of template matching, an image is stored as a two dimensional array of numbers and is compared using some metric (e.g. the Euclidean distance) with a single template[2].

*Brunelli and Poggio*[2] performed a series of experiments to determine the superior of the two approaches (i.e. geometric feature based approach and template based approach). They used a database of 188 images, four for each of the 47 subjects. Two

of the four pictures were taken in the same session while the other two were taken over an interval of two weeks.

### Geometric Feature Based Matching:

The work of *Brunelli and Poggio*[2] is loosely based on that of *Kanade*[38]. *Kanade*[38] developed a face recognition system that used a database of 20 people (2 views per person) and achieved a 75% recognition rate. His approach was also based on geometrical feature based approach. *Brunelli and Poggio*[2] first normalised their images by means of a normalised cross-correlation coefficient defined by :

$$C_N(y) = \frac{\langle I_T \cdot T \rangle - \langle I_T \rangle \langle T \rangle}{\sigma(I_T) \sigma(T)} \quad 2-1$$

where

$I_T$  is the patch of image I that must be matched to T.

$\langle \rangle$  is the average operator.

$I_T \cdot T$  is the pixel by pixel product

$\sigma$  is the standard deviation over the area being matched

This was done so that the images were independent of position, scale and rotation of the face in the image plane. They employed techniques of hierarchical correlation to overcome the computational intensity demanded by standard correlation[2].

Feature extraction was then performed on the normalised image using integral projections. These projections are simply defined as: if  $I(x,y)$  is an image, then the vertical projection in the  $(x_1, x_2)$   $(y_1, y_2)$  rectangle is defined as[2] :

$$V(x) = \sum_{y=y_1}^{y_2} I(x, y) \quad 2-2$$

and the horizontal direction is defined as

$$H(y) = \sum_{x=x_1}^{x_2} I(x, y) \quad 2-3$$

*Brunelli and Poggio*[2] claim that integral projections can be extremely effective in determining the position of the features, provided that the window on which they act is

suitably located. Before any of the features are located, the vertical and horizontal edge maps are calculated. The window used to locate the mouth and nose is guessed using anthropometric standards. As a first estimate, peaks are searched for in the horizontal projection of the vertical gradient map. The peaks are rated with respect to the distance from the expected location, as well as the prominence of the peaks. The highest rating peaks are the vertical position of the mouth. Once the vertical position is found, the search is refined. The eyebrows are found using a similar method on the vertical gradient map. Their eyebrow detector searches for pairs of peaks of gradient intensity with opposite direction. No hairline information was considered. Dynamic programming was used to determine the face outline. This was done by exploiting the fact that the face outline is essentially elliptical. Hence dynamic programming was used to follow the outline on a gradient intensity map of an elliptical projection of a face image[2].

A 35-D vector that contained geometric feature information was composed as follows:

- eyebrow thickness and vertical position at eye centre position
- a coarse description (11 data elements) of the left eyebrow arches
- nose vertical position and width
- mouth vertical position, width(upper and lower lips) and height
- 11 radii describing the chin shape
- bigonial breadth (face width at nose position)
- zygomatic breadth ( face width halfway between nose tip and eyes)

A Bayes classifier was used for the recognition.

### **Template Based Matching :**

Correlation lies at the core of the template approach. The image is normalised as described in equation 2-1. Each person is represented in the database by 4 masks representing the eyes, nose, mouth and full face template of that person. The unclassified image is compared to the rest of the database using normalised cross correlation. A score is returned for each correlation. The unclassified image is then

placed in the same class as the vector with the highest cumulative score. *Brunelli and Poggio*[2] point out that correlation is sensitive to illumination gradients and hence different normalisations were used :

- no preprocessing, a plain intensity image was used.
- intensity normalisation using the ratio of the local value over the average brightness in a suitable neighbourhood.
- gradient magnitude: the intensity of the gradient computed with an  $L_1$  norm on a Gaussian regularised image
- the Laplacian of the intensity image was computed

It was reported that the best recognition rates were obtained when gradient information was used. *Brunelli and Poggio*[2] reported that the experimental analysis showed that features could be rated as follows (in accordance of decreasing performance):

1. eyes
2. nose
3. mouth
4. whole face template

The difficulty associated with perfectly normalising each full face template is attributed to the low performance of the whole face template by the authors. They also suggest that since the whole face is the template, it is sensitive to slight deformation due to deviations from frontal views. *Brunelli and Poggio*[2] reported that their template based strategy was superior in recognition performance even though the feature based strategy allowed for high recognition speeds and smaller memory requirements. They admit that their template based approach contains some elements of the feature based approach, but argue that successful object recognition architectures need to combine aspects of feature based approaches with template matching techniques.

*Samaria*[8] proposed a new approach to the problem of face recognition that used hidden markov models. The faces are treated as 2-D objects and the model is able to extract statistical facial features. *Samaria*[8] states that this model is also able to use

structural information, thus yielding a hybrid model. Although hidden markov models are conventionally used in speech recognition (because of the 1-D nature of the speech signals), the 2-D images can be converted to 1-D spatial sequences. *Samaria*[8] achieved this by sampling an image using a 2-D sliding window.

The following results were obtained :

**Table 2. 3** : Error rates obtained at various model resolutions by *Samaria*[8]

Full image resolution	Error Rate
92x112	5.5%
46x56	6.5%
23x28	6%
12x14	12%

The low error rate is at the expense of high computational cost. A single classification takes 4 minutes on a Sparc II for the system to sustain an error rate of 5.5%. It was also noted by *Samaria*[8] that although an increased recognition rate was achieved, the segmentation obtained with the pseudo 2-D hidden markov models was quite erratic. *Samaria*[8] used the ORL<sup>3</sup> database.

*Intrator et al*[16] use a GST (General Symmetry Transform – described in Chapter 4) for the detection of certain facial features (the transform can detect the eye and mouth regions quite accurately). The GST output is projected onto the horizon, edge linked and local maxima are used for the detection of certain anchor points. After locating the eyes and mouth, affine transformations were applied to the image. Feed-forward neural networks were implemented for classification due to their ability to cope with very high dimensional data. *Intrator et al*[16] state that they are thus excellent candidates to perform recognition from pixel values. A hybrid training method was also employed, that used unsupervised methods for extracting features and supervised methods for minimising the classification error. The hybrid training in the feed-forward network was

<sup>3</sup> ORL - Olivetti Research Laboratory

achieved by modifying the learning rule of the hidden units to reflect additional constraints.

An MIT media lab database, which consisted of 37 instances of 16 distinct people, was used. *Intrator et al*[14] don't report on the number of faces used in the test and training sets. The results obtained are as follows:

**Table 2. 4 :** Classification errors obtained by *Intrator et al*[14].

Method	% Error
Back Propagation	3.28±0.31
Hybrid BCM/BP	3.96±0.96
Averaged Back-Propagation	1.25
Averaged Hybrid BCM/BP	0.62

*Konen and Schulze-Kruger*[39] presented a novel technique that was an extension of the elastic graph matching algorithm. This system is able to simultaneously perform localisation, separation, standardisation and recognition. The core principle of this approach lies in the fact that faces are stored as flexible graphs with characteristic visual features attached to the node of the graphs. The features are extracted by a convolution with Gabor wavelets and are computed at the location of the graph node[39].

The author states the following computational advantages in using labelled graphs for data representation:

1. Robustness - Gabor features are more tolerant to head posture, size and facial expression than raw gray level images.
2. Data Compression - They achieve data compression of about a factor of 10.
3. Scaling - Changes in geometry can be accounted for in sparse graphs.

4. Distribution - If information at a single node is missing, recognition is still possible as the graph contains sufficient information distribution.

Variations like facial expression and wearing glasses are handled very well by the algorithm. The system achieved a recognition rate of 96% and a false rejection rate (FRR) of 4% on a database of 87 persons (1 image per person). The actual approach is not clearly described by the authors, rather a high level explanation is provided.

*Zhang et al*[40] performed a comparative study of three recently proposed algorithms for face recognition:

- eigenfaces
- auto-association and classification neural networks
- elastic graph matching

These techniques were analysed under a statistical decision framework and evaluated experimentally on four different databases of moderate subject size. The following databases were used:

**Table 2. 5:** Databases used by *Zhang et al*[40].

Database	Subject	Variation	Total
MIT	16	3	48
Olivetti	40	2	80
Weizmann	28	3	84
Bern	30	2	60
Combined	114	2,3	272



The following results were obtained:

**Table 2. 6:** Results obtained by *Zhang et al*[40].

Database	Eigenface	Elastic Matching	Auto-Association and Classification Networks
MIT	97%	97%	72%
Olivetti	80%	80%	20%
Weizmann	84%	100%	41%
Bern	87%	93%	43%

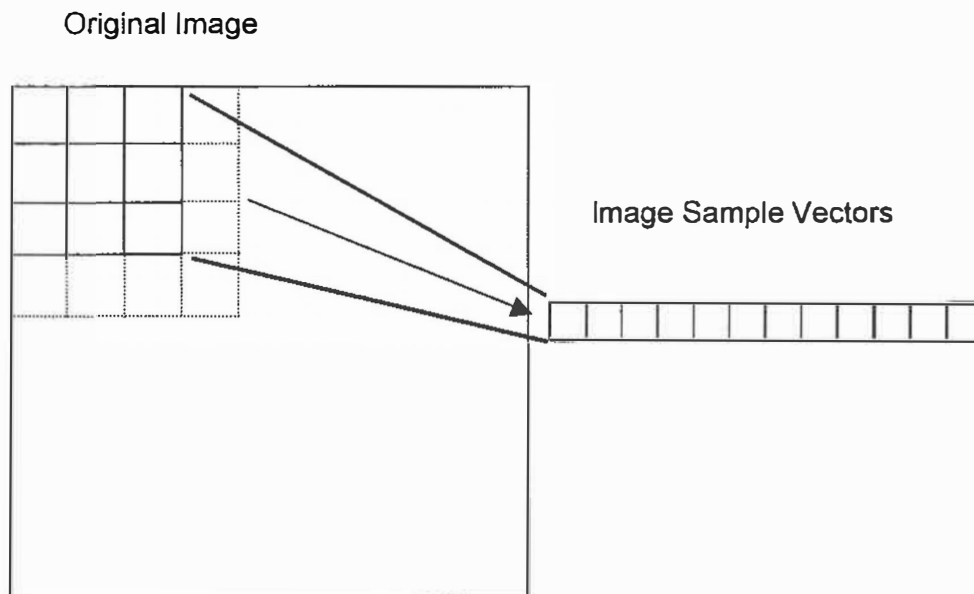
*Zhang et al*[40] reported that the eigenface algorithm worked well only when the lighting variations were small. This is due to the fact that the eigenface algorithm is essentially a minimum distance classifier, and hence its performance decreases as lighting variations increase. *Zhang et al*[40] state that lighting variations introduce biases in distance calculations. Hence, when the biases are large, the image distance is no longer a reliable measure of face difference.

As stated in *Konen*[39], the elastic matching algorithm is insensitive to lighting variation, face position and expression variations and therefore more versatile. *Zhang et al*[40] state that this can be attributed to Gabor features and the fact that features at key points in the image, rather than the entire image are used. The only disadvantage presented by the elastic template matching is that it requires more computational effort than the eigenface approach. However, the authors feel that the superior performance seems to justify the computational intensity. Due to implementation difficulties, the eigenface algorithm is preferred over the classification networks.

### 2.3.3 Neural Network approaches

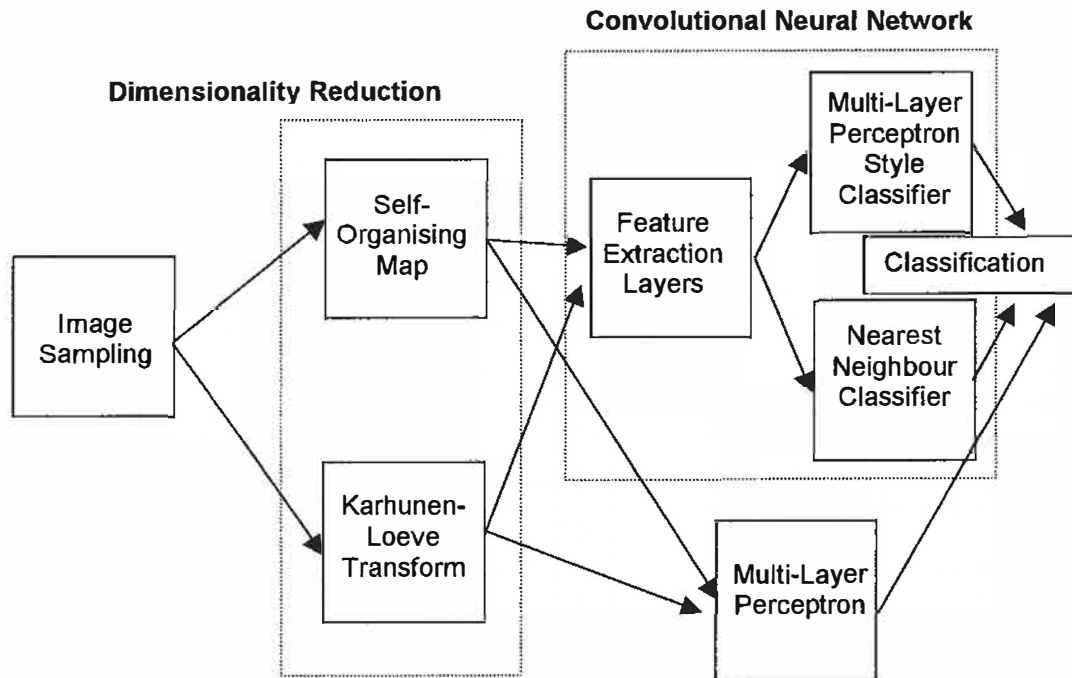
*Lawrence et al*[41] used a hybrid neural network approach in developing a computational model for face recognition. Their system combined local image sampling, a self-organising map (SOM) neural network, and a convolutional neural network. The image sampling was implemented in two ways:

- A local window was stepped over the image, and a vector was created from simple grey level intensity values at each point in the window.
- The second method created the vector by taking the grey level intensity of the centre pixel of the window, and the difference of this pixel to the rest of the pixels within the square window. This representation is partially invariant to variations in intensity of the complete sample.



**Figure 2. 2** : Depiction of local sampling process. Reproduced from *Lawrence*[41]

*Lawrence et al*[41] describe their system as follows :



**Figure 2. 3 :** System used for face recognition by *Lawrence et al*[41] showing the alternative methods that were considered and reported on.

*Lawrence et al*[41] used the ORL database (40 subjects with 10 distinct images each). They varied the number of faces in the testing and training sets and found that best results were obtained with 5 faces of a subject in a training set and 5 in the testing set, with no overlap between testing and training sets. Results are presented using the Karhunen-Loeve transform (KLT) in place of the self-organising map and a multi-layer perceptron in place of the convolutional network. It is reported that the KLT performs almost as well as the SOM and the multi-layer perceptron performs very badly. *Lawrence et al*[41] report the following results :

**Table 2. 7:** Table of results obtained by *Lawrence et al*[41].

System	Error rate	Classification time
Top-Down HMM	13%	n/a
Eigenfaces	10.5%	n/a
Pseudo 2-D HMM	5%	240s
SOM+CN	3.8%	<0.5s

**Table 2. 8:** Table showing different dimensionality reduction techniques and respective error rates.

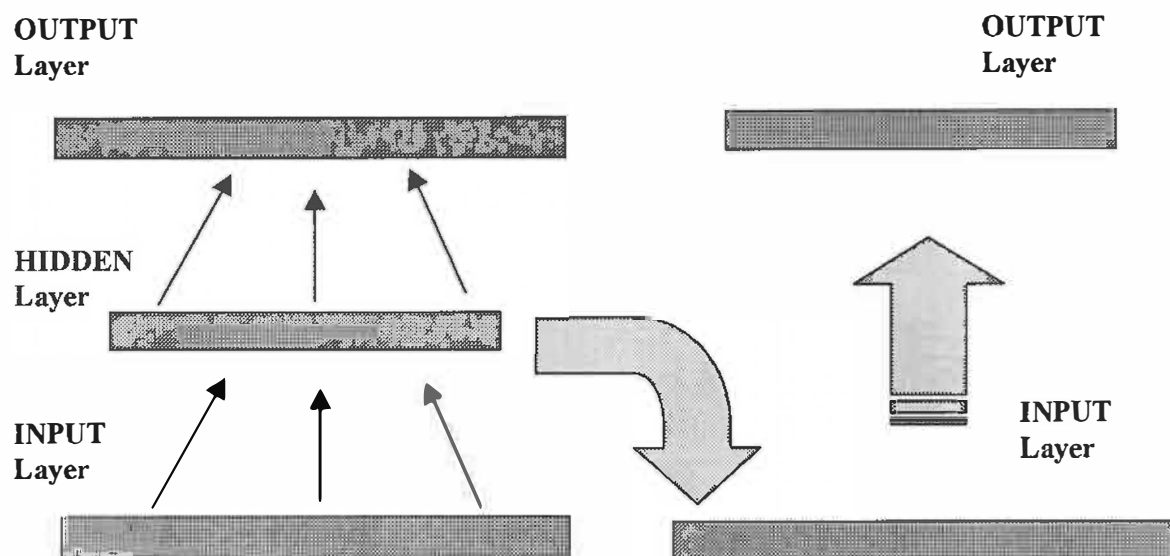
Dimensionality Reduction	KLT	SOM
Error rate	5.33%	3.83%

**Table 2. 9:** Table of results for different combinations used by *Lawrence et al*[41]

	KLT	SOM
MLP	41.2%	39.6%
CN	5.33%	3.83

*Fleming and Cottrell*[1] used a dataset comprising 204 face and 27 non-face images. All images were digitised from a live video signal of 504x488 8-bit pixels. The lighting, position of the camera, location and orientation was kept constant. The training set comprised 64 face images (29 male and 37 female faces). Two testing sets were used by the authors : a 'familiar' set (28 male and 34 female) that consisted of 'unchosen' images from the individuals that were used in the training set and a 'novel' set (8 male and 70 female) that consisted of the images of the individuals that the network had not been trained upon. This was done to allow for the testing of the model's robustness. Two neural networks were used: a 63x61 image is presented to a three layer back-propagation network to facilitate image compression. Upon successful training of this network, the network's weights are fixed and the hidden unit encoding of each face is passed as a feature to the second network. The compressed hidden unit representations

are used by the second network for training. It is interesting to note that the second network doesn't contain any hidden layers (see Figure 2.4).



**Figure 2. 4:** Model of networks used by *Fleming and Cottrell*[1]. The network on the left is the image compression network (implemented via a back-propagation network). The network on the right is used for classification purposes and receives as input the compressed hidden unit activations for each image in the training set.

*Fleming and Cottrell*[1] report that the model was highly accurate over a wide range of stimuli and produced a 70% identity recognition rate. The reported work doesn't accurately describe the system. The methodology and criterion for building the testing and training sets is also not concisely explained.

*Samaria*[8] reported that the comparison and description of the various studies in the field of face recognition is indeed a difficult task since the results reported by the various authors are obtained using

- different image sets
- different databases
- no common terminology to describe the methods

The author concurs with this point of view.

### 2.3.4 Basis of this study

Most of the recognition systems that have been described are complex (see e.g. [40], [41] and [8]) and require large processing power. Some of the others (see e.g. [8], [39] and [16]) take quite a long time (e.g. *Samaria*[8] takes approximately 4 minutes) for recognition to take place. The approach adopted in this study is to extend the work of *Bouwer and Broadhurst*[6], whose one-dimensional discrete cosine transform approach was based upon the method described by *Wilder*[9]. The reason for choosing such a technique is that it is easily implemented on current computer hardware. Face recognition on this system is also quite rapid, and a high true positive recognition rate was reported (see *Bouwer*[6]). Bouwer's investigation was based on a small database. This section investigates the approach adopted by Bouwer and the possible reasons for the limitations reported by him. A new recognition model based upon a two-dimensional discrete cosine transform is proposed in Chapter 3 and described in *Debipersad and Broadhurst*[43].

Bouwer's implementation of a face recognition system used a small database of 22 subjects which was captured with 12 images per subject over a two day period. No attempt was made to control the lighting, and the subject was captured against a black background. A manual technique was used to segment the face from the black background. The grey scale values of the images were summed along vertical as well as horizontal directions, thus creating two one-dimensional signatures from the two-dimensional image data. Five different windows of interest (i.e. full face, eye region, eye and nose region, nose and mouth region, and mouth and chin region) corresponding to various combinations of facial features were used on the normalised image, resulting in ten one-dimensional signatures[6]. The 1-D DCT was performed on these signatures and the first 25 components of each 1-D DCT were retained.

### 2.3.4.1 Grey-scale Projection Model

Unfortunately no detailed results were published by *Bouwer and Broadhurst*[6] of his model and its limitations. These are best illustrated using the grey-scale images of three different subjects and used to develop the model that was improved upon in this study. Bouwer used normalised images of dimension 132x232 pixels. For the purposes of illustrating the model implemented by Bouwer, images that are 81x89 pixels (digitised to 8-bit accuracy) are used.

The average grey-scale values for pixels summed in the vertical direction is:

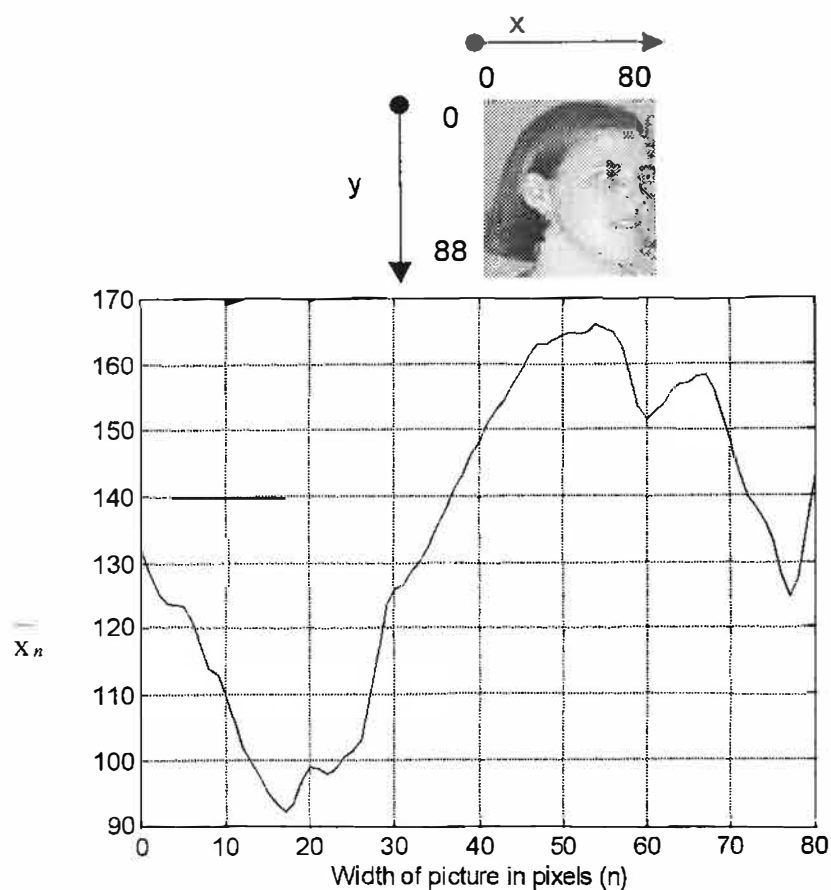
$$\bar{x}_n = \frac{1}{81} (y_{n0} + y_{n1} + y_{n2} + \dots + y_{n88}) \quad 2-4$$

where n is the column number ranging from 0 to 80. The average grey-scale values for pixels summed in the horizontal direction is:

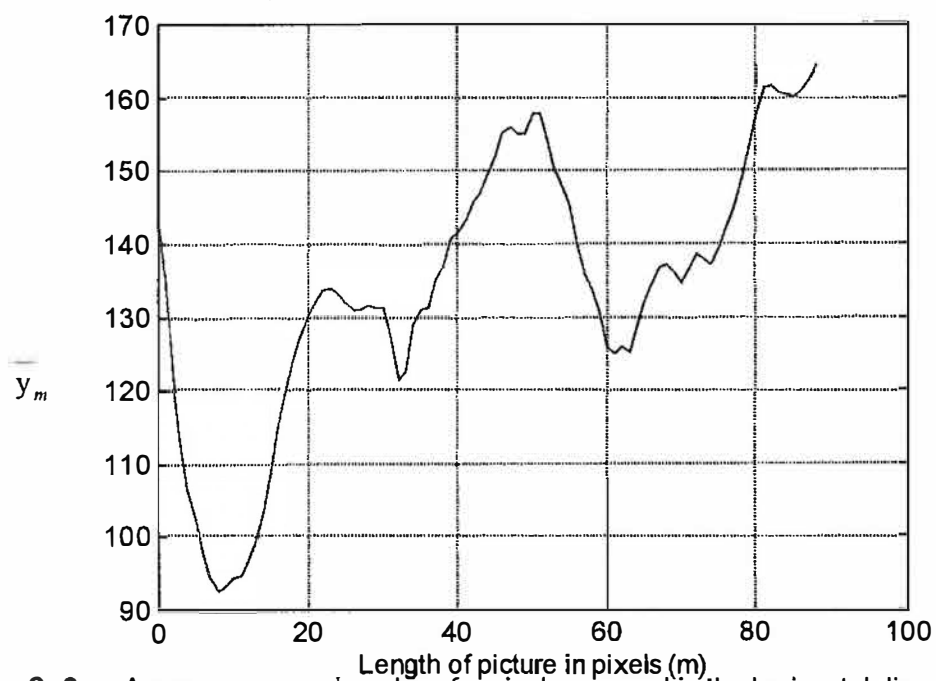
$$\bar{y}_m = \frac{1}{89} (x_{m0} + x_{m1} + x_{m2} + \dots + x_{m80}) \quad 2-5$$

where m is the row number ranging from 0 to 88.

Figures 2.5 to 2.10 show  $\bar{x}_n$  and  $\bar{y}_n$  for the three different images.

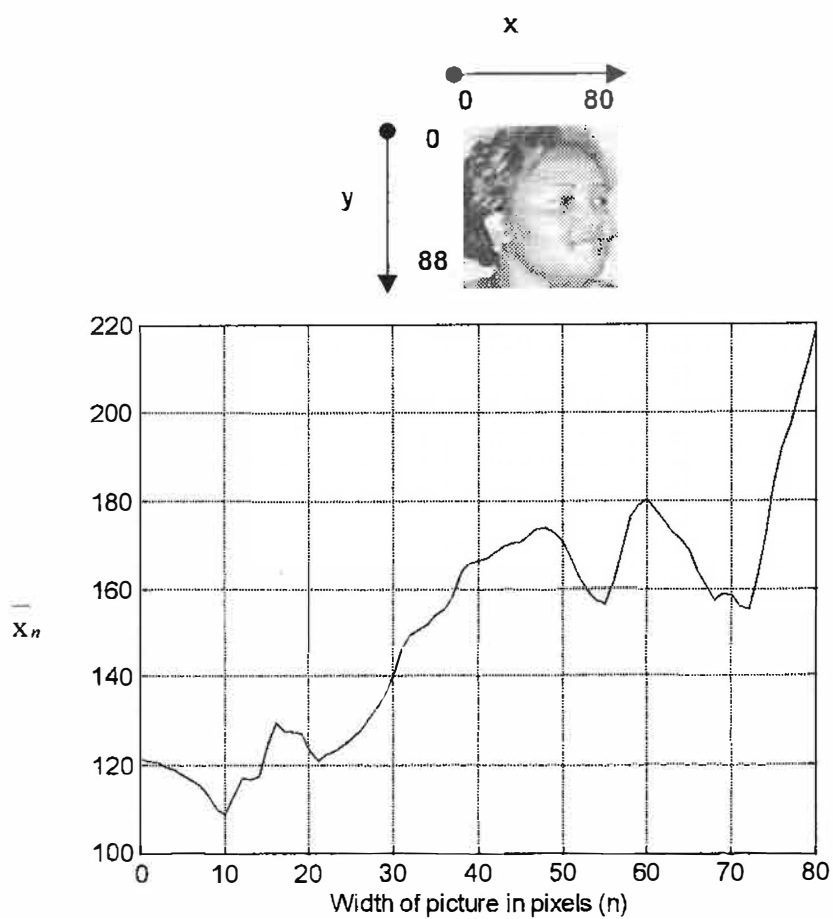


**Figure 2. 5:** Average grey-scale values for pixels summed in the vertical direction for image 1.

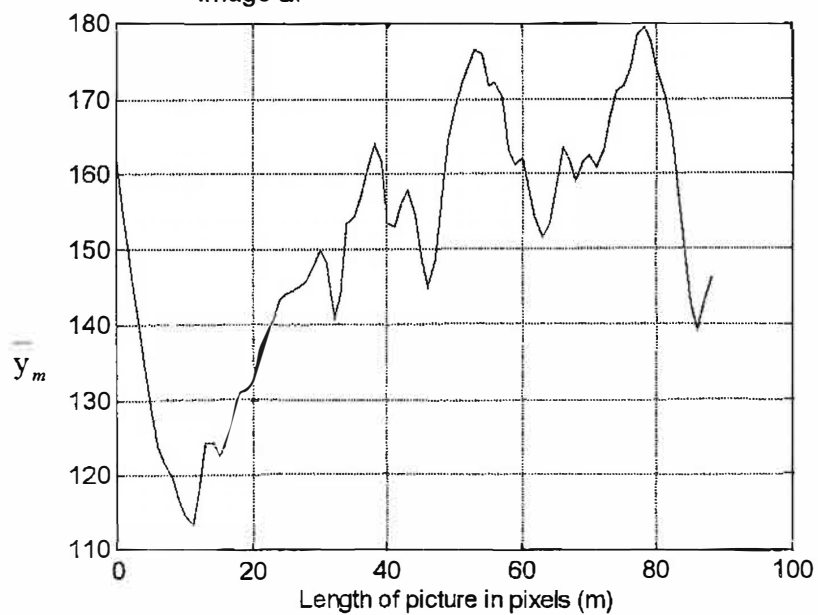


**Figure 2. 6:** Average grey-scale values for pixels summed in the horizontal direction for image 1

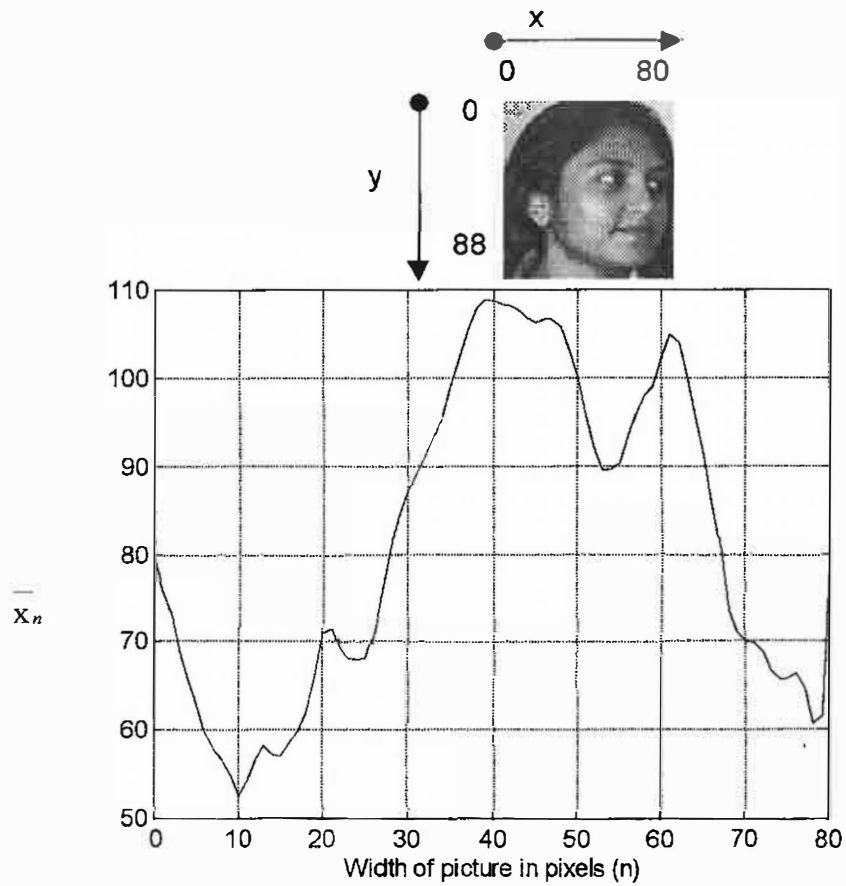




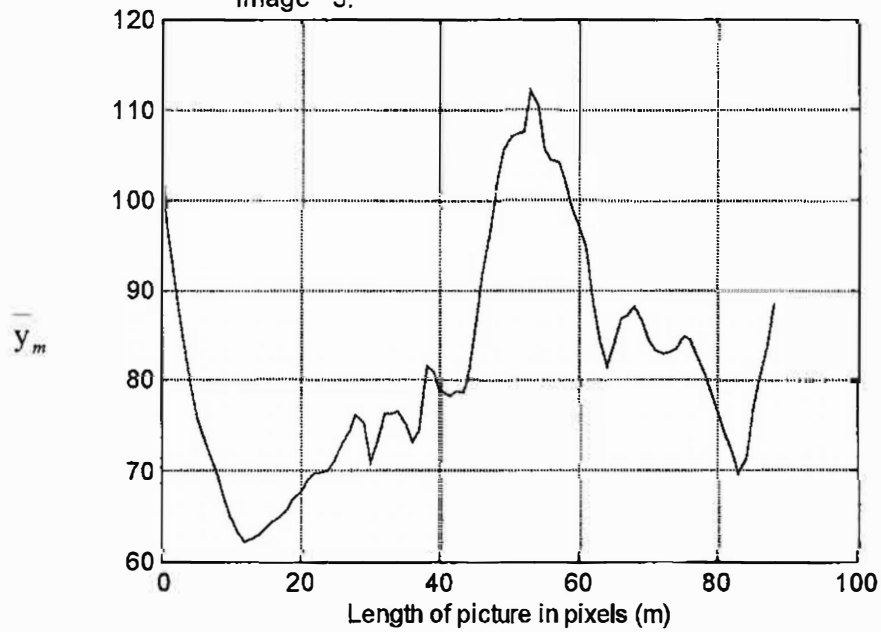
**Figure 2. 7:** Average grey-scale values for pixels summed in the vertical direction for image 2.



**Figure 2. 8:** Average grey-scale values for pixels summed in the horizontal direction for image 2.



**Figure 2. 9:** Average grey-scale values for pixels summed in the vertical direction for image 3.



**Figure 2. 10:** Average grey-scale values for pixels summed in the horizontal direction for image 3.

Figures 2.5 to 2.10 serve to illustrate the initial data reduction technique that was used by Bouwer. The discrete cosine transform (DCT) was used to further reduce the dimensionality of the data presented to the classifier. The DCT is given by

$$\bar{X}_k = c_n \cdot \sum_{k=0}^{80} \bar{x}_n \cdot \cos\left(\frac{\pi \cdot n(2k+1)}{162}\right) \quad 2-6$$

for the average grey-scale values summed in the vertical direction. The following equation defines the DCT for the average grey-scale values summed in the horizontal direction.

$$\bar{Y}_k = c_n \cdot \sum_{k=0}^{88} \bar{y}_m \cdot \cos\left(\frac{\pi \cdot n(2k+1)}{176}\right) \quad 2-7$$

$$\text{where } c_0 = \frac{1}{\sqrt{N}}, c_n = \sqrt{\frac{2}{N}} \\ \text{and } 0 \leq n \leq N-1$$

Bouwer set the dc term (i.e.  $\bar{X}_0$  and  $\bar{Y}_0$ ) to zero. It was judged by Bouwer that the next 23 components of each DCT transformed signature be used as the feature extracted input to a neural network[6].

Figures 2.11 to 2.16 give the DCT values for  $\bar{X}_k$  and  $\bar{Y}_k$  for all three images shown in Figures 2.5 to 2.10.

*Wilder*[9] performed an additional data reduction step in which the 1D signatures are integrated over bands narrow enough to resolve the main features of the face, but yet wide enough to neglect the fine features of the face that may vary over time. Bouwer did not consider this beneficial and was thus not implemented.

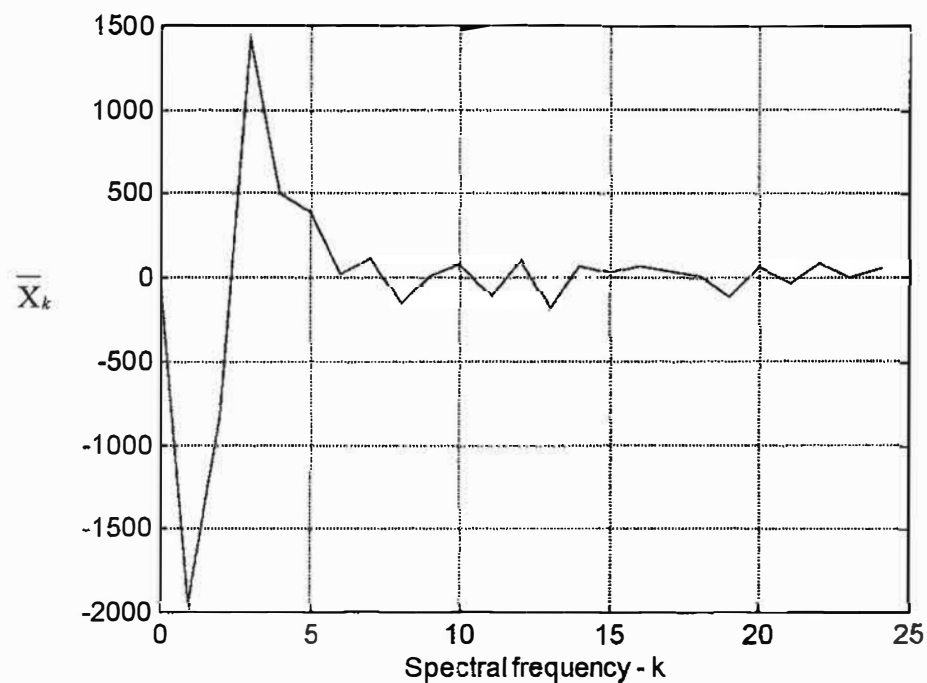


Figure 2. 11 : DCT for image 1 summed in the vertical direction.

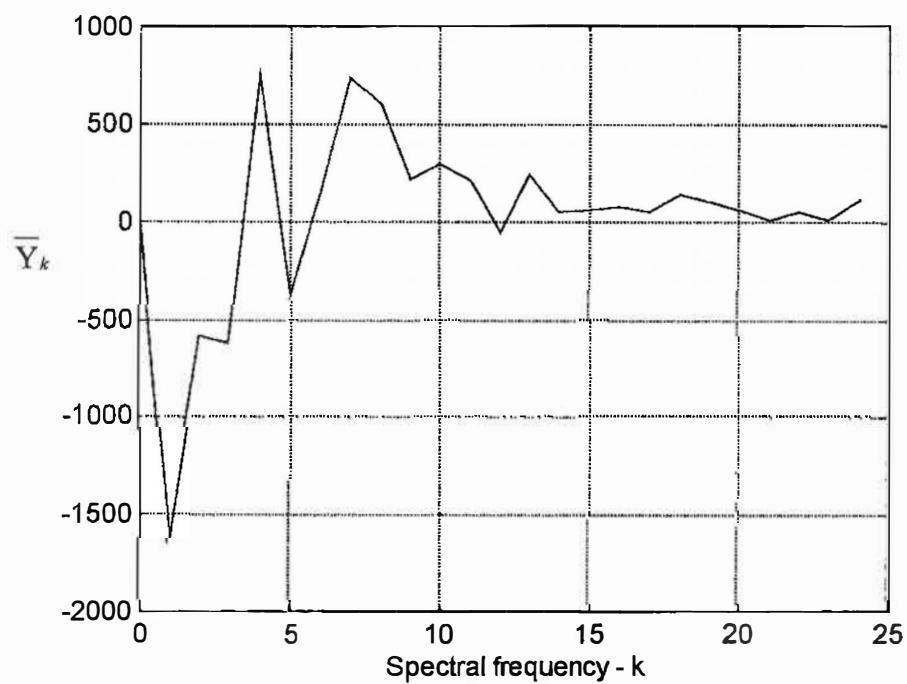


Figure 2. 12 : DCT for image 1 summed in the horizontal direction.

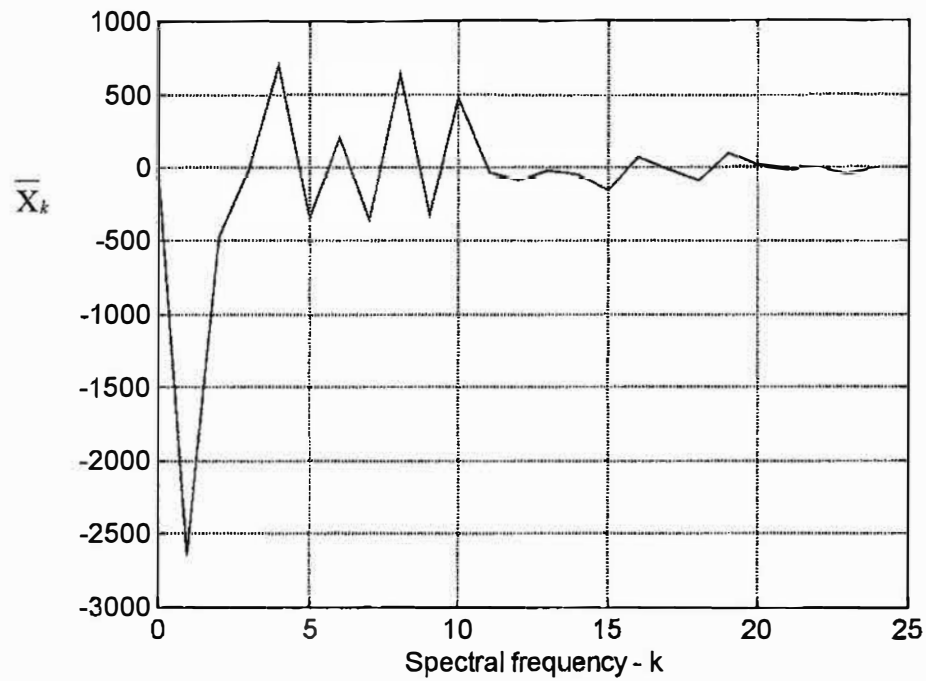


Figure 2. 13 : DCT for image 2 summed in the vertical direction.

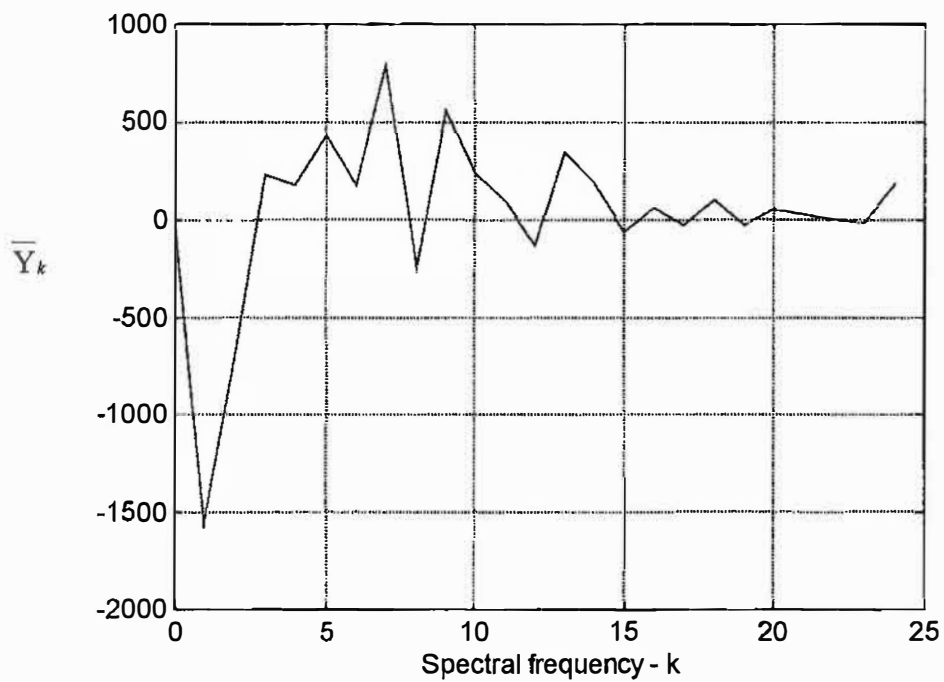


Figure 2. 14 : DCT for image 2 summed in the horizontal direction.

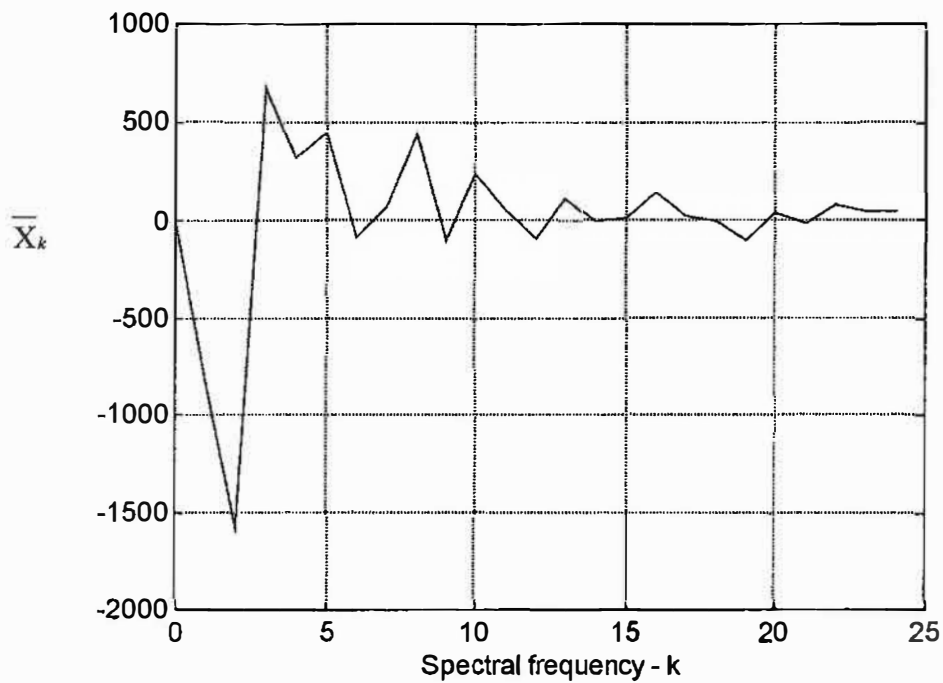


Figure 2. 15 : DCT for image 3 summed in the vertical direction.

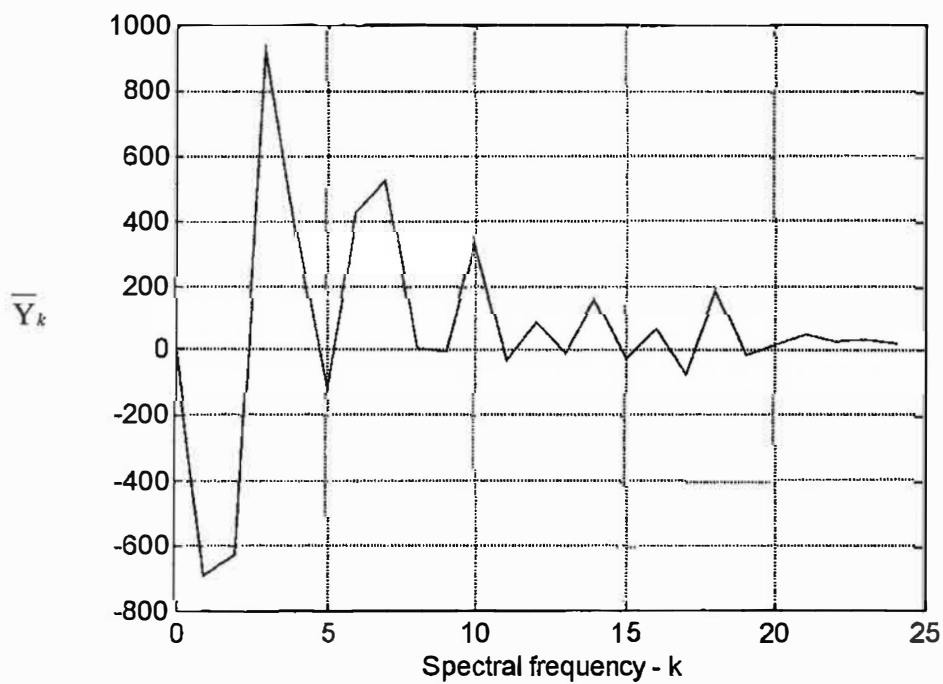


Figure 2. 16 : DCT for image 3 summed in the horizontal direction.

Figures 2.11 to 2.16 show a higher variance in the lower frequencies. Bouwer's findings showed that the DCT data obtained from the grey-levels summed in the vertical direction produced better classification results, than those summed in the horizontal direction. Thus the vertical direction DCT data ( $\bar{X}_k$ ) was used as inputs to a neural network.

A Mahalanobis distance metric (MDM) was used to obtain an indication of how the test pattern ranked relative to the overall location of other known patterns in the pattern class. The Mahalanobis distance between the test vector and the prototype centroid vector is defined as

$$M(\bar{x}, \bar{\mu}) = (\bar{x} - \bar{\mu})^T \Sigma^{-1} (\bar{x} - \bar{\mu}) \quad 2-8$$

This was used to measure the distance from an input face to a statistical prototype of each person in the database. Eight of the twelve images captured were used for training, while the other four were used for testing. The eight images were formed into statistical prototypes by calculating the mean and variance of each of the components of the 25-D vectors. The Mahalanobis distance between the eight 25-D vectors of every subject and the prototype were calculated. If one of the 25-D vectors were of the same subject as the prototype, this was classified as a positive example, otherwise it was a negative example.

A neural network was used for classification and the fundamental aim of Bouwer was to construct a classifier that did not need to be retrained every time a person was added or deleted from the database.

### 2.3.4.2 Appraisal of Bouwer's[6] technique

Bouwer used a small database of 22 subjects. A vector was presented to the one dimensional discrete cosine transform. The first term of the transform is responsible for overall skin tone[9], and in general can be ignored to nullify the effects of lighting. If subjects were captured at different times during the day, i.e. all of Subject A's images in the morning, all of Subject B's images at mid-day and all of Subject C's images in the evening, each image should be slightly different from the next as far as the light levels are concerned. The average value of the pixels obtained from the "averaging process" is dependent on the light falling on the subject. Hence, this could interfere with recognition rates since it introduces variance in the data. The model was to a degree dependent on the amount of light illuminating the subject. Bouwer did not analyse the variance of the inputs to the neural network. It is possible that features with not enough variance, or features with redundant aspects were used as inputs to the network.

The process of averaging the vertical face data could possibly lead to loss of important feature information that could form important pattern boundaries. Bouwer used 8 images of the 12 that were captured for each subject for training and 4 for testing. He claimed results of 99.6%. This recognition technique would work well in a controlled environment, as evidenced by *Wilder*[9].

Bouwer reported that an increase in database size led to an overall decrease of the classifier performance. This was reported by *Wilder* as well. Bouwer has to date not yet published the masters research.



## Chapter 3

# OVERVIEW OF THE SYSTEM AND ITS COMPONENTS

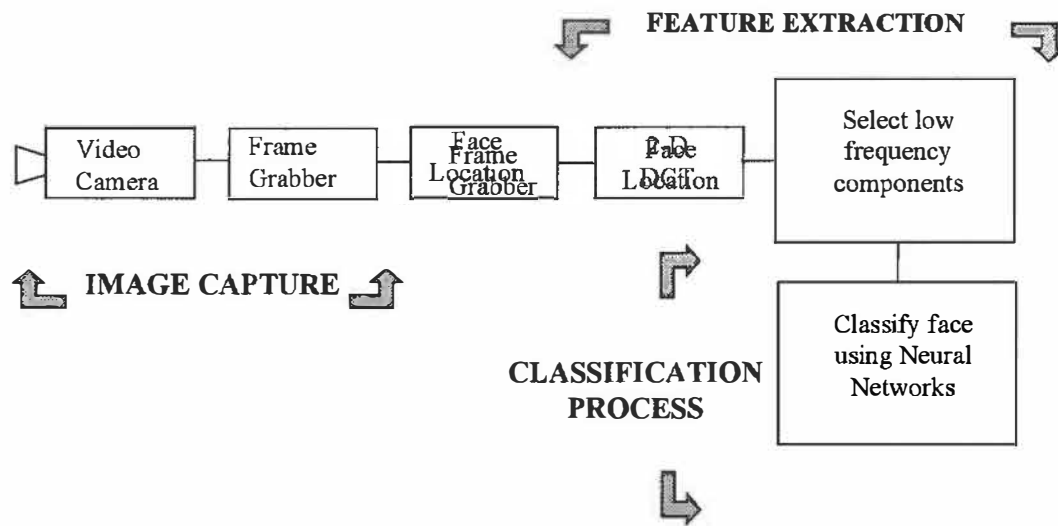
---

### 3.1 Introduction

System components make up the face recognition algorithm. A new recognition model based upon a two-dimensional discrete cosine transform is proposed. Details of the image capture process as well as the criterion that was used to build up the database are discussed.

### 3.2 A Two-dimensional Approach

This section provides an overview of the model that was implemented in this study. A detailed account is given of image capture, while brief introductions are provided for feature extraction and classification. The general approach (i.e. image capture, feature extraction and classification) adopted in this study is similar to that adopted by *Lawrence et al*[41] in their research on hybrid neural networks for face recognition. Figure 3.1 is a graphical representation of the system implemented.



**Figure 3.1:** Graphical model of approach adopted

### 3.2.1 Video Camera

A Sony™ Handycam Video 8 (Model Number CCD TR380E PAL) was used. The camera is equipped with a digital zoom and can work down to 0.1 lux. *Program AE* provides a facility to specify the type of environment in which the camera is operated. The *Outdoors* mode was chosen to compensate for the two light sources that were used. (See § 3.3.1) The digital zoom on the camera was fixed.

### 3.2.2 Frame Grabber

The image was digitised to 8-bit accuracy (256 levels of grey) using a DT-2867 Frame Grabber card. The DT-2867 is a dedicated image processing board with numerous image processing capabilities (real time A/D window, erosion, dilation and histogram processing). The board has two memory buffers that allow for high speed image processing. The image is digitised and stored in memory buffer 1 which allows it to be easily written to disk in a raw file format.

### 3.2.3 Face Location

Face location is performed to select the face from the background. This is done to reduce the redundancy of the data that is presented to the classifier. A generalised symmetry transform[16] was implemented to locate the face and is described in more detail in Chapter 4, §4.1. Although the transform is accurate, the inherent problem with the generalised symmetry transform is that it is computationally intensive and therefore not practical. Instead, manual normalisation was used. A detailed description of face location is given in Chapter 4, §4.1.

### 3.2.4 2-D Discrete Cosine Transform

To overcome the problems encountered by *Bouwer*[6] with large databases, it was decided to increase the inputs to the neural network. However, one must be careful to use data that is de-correlated (to reduce the redundancy presented to the network). A two-dimensional discrete cosine transform was used to de-correlate the data. After the image is captured and normalised a 2-D DCT is applied to the image. This step facilitates data reduction as well as feature extraction. Certain coefficients are chosen in accordance with a variance distribution of the transformed image, while others are discarded. These coefficients form the features that are used as inputs to the neural networks. The 2-D DCT, as well as the features that were selected as inputs to the neural network is discussed extensively in Chapter 4, §4.2.

### 3.2.5 Classification Process

The training of the classifier can be performed in two ways:

- A database approach, in which all (or a fixed number) of the subjects exist and are assigned a unique pattern number.
- A network per person approach, in which a separate network is trained for each valid subject.

Both approaches were investigated. A back-propagation neural network with momentum was used for determining identity. In this case a network is trained for every person i.e. if there are  $N$  individuals, then there are  $N$  separate neural networks. This approach was not investigated in detail and was performed to compare the results against the database approach. The counter-propagation network and the radial basis function network were investigated in the database approach.

### 3.3 Image Capture

*Samaria*[8] did intensive research on face recognition systems proposed by other researchers. It was found that descriptions of the face database that was used by the researchers was incomplete, and left certain question unanswered with respect to the spread of subjects in the database. *Samaria*[8] posed the following questions regarding face databases:

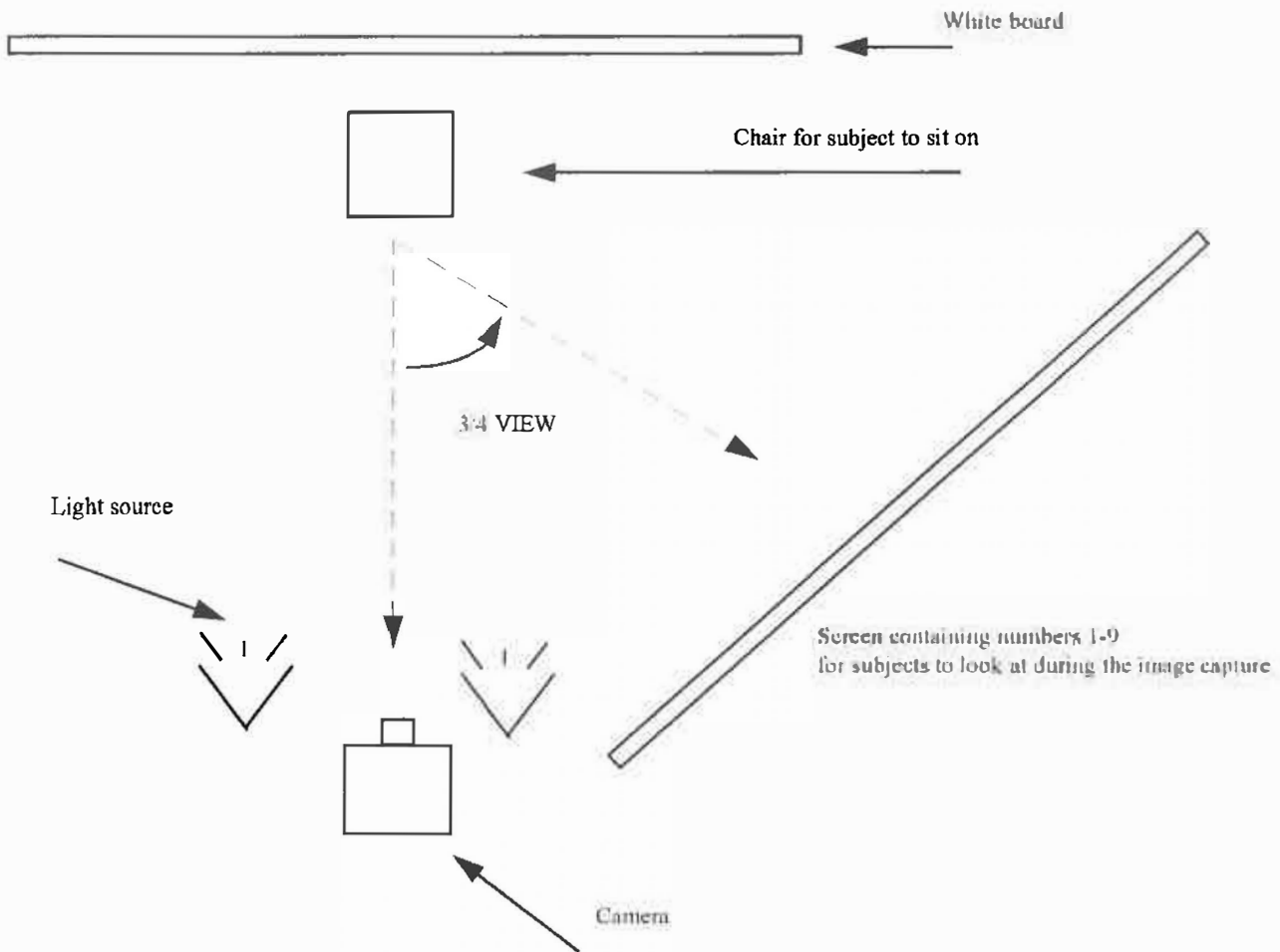
- Are expression, head orientation and lighting conditions controlled?
- Are subjects allowed to wear glasses and have beards or other facial marks?
- Is the subject sample balanced? Is gender, age and ethnic origin spanned evenly?
- How many subjects are there in the database? How many images will be used for testing and how many for training?
- Will scaling be controlled?

These important considerations were taken into account when building the database for they allow the robustness of the recognition model to be tested. Before describing the image capture process, it is necessary to understand the experimental set-up.

#### 3.3.1 Experimental Set-up

Images of students were captured during the registration week of February 1997 in the Postgraduate Laboratory of the Department of Electronic Engineering. The laboratory has blinds on the windows, as well as fluorescent light sources. Two incandescent light

sources were used to illuminate the subject. No attempt was made to control the light that entered through the blinds. The fluorescent lights did not affect the captured images. The laboratory was set up as follows:



**Figure 3. 2 :** Top View of Image Capturing Set-up used in the Postgraduate Laboratory

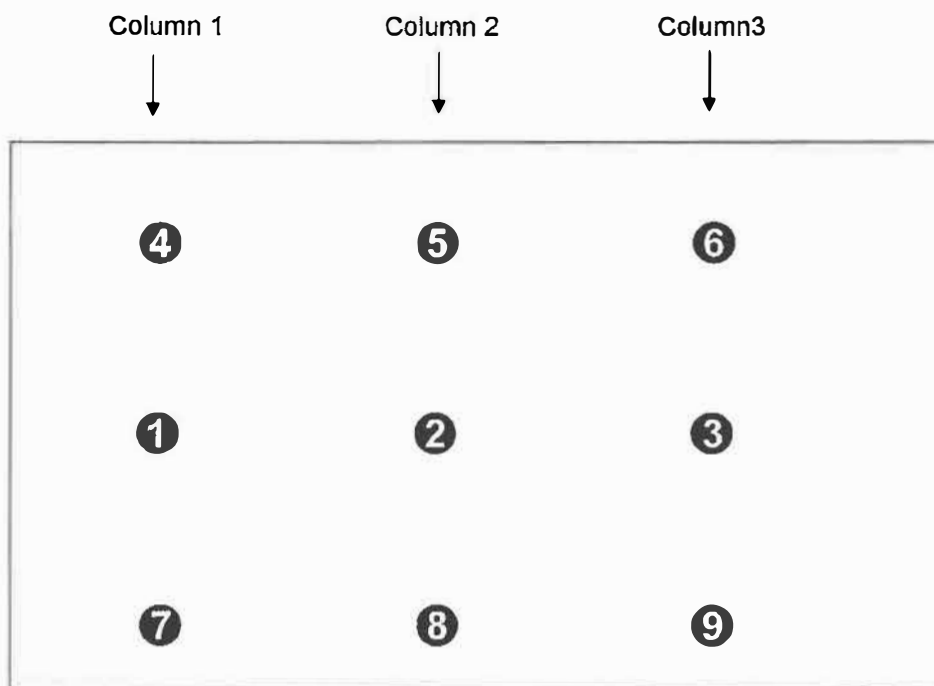
Images that made up the database were captured in accordance with psychophysical observations of viewpoint dependence of face recognition in humans[20]. *Schyns and Bulthoff*[24] defines the  $\frac{3}{4}$  view to be  $\pm 36^\circ$  from the frontal view. (See Chapter 2, § 2.1).

Frontal views as well as  $\frac{3}{4}$  views were taken of each subject. Subjects were captured against a white background (to facilitate easier face location). The subjects were asked

to sit on the chair provided, and look at a board containing numbers. The lights were then switched on to illuminate their face. The camera was attached to a tripod, whose height was adjusted so that the subject's head and shoulders were always captured. The distance from the camera to the subject was fixed, as well as the zoom on the camera.

The person capturing the images called out a view number ranging from 1 to 10. Positions 1 to 9 are indicated in Figure 3.3. The subject then moved his/her head to look at that number. Position 10 required the subject to look straight ahead at the camera. The image was then captured. The grey values ranged from 0 to 255, with 255 being white and 0 black. The window size used was 215x255 pixels.

Ninety-three distinct subjects were captured, with 10 views per subject. Thus the database consists of 930 facial images.



**Figure 3. 3 :** Screen containing the number for subjects to look at. (referred to as View Number)

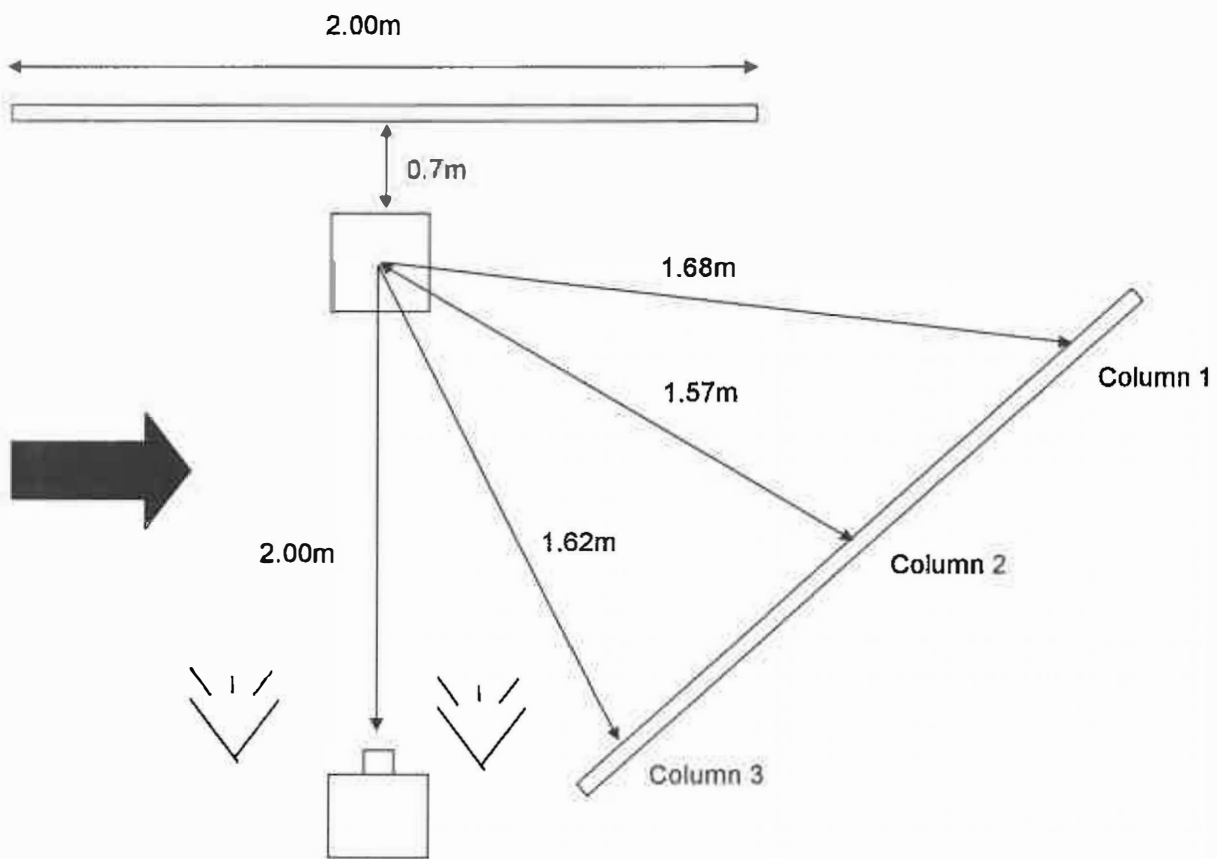


Figure 3. 4 : Dimensions of experimental set-up used in the laboratory

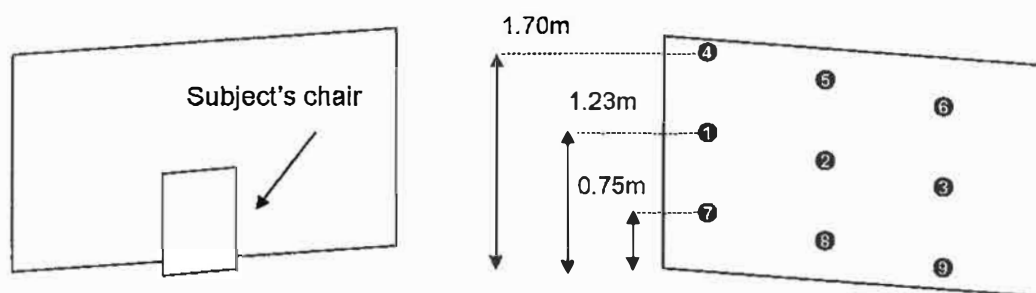
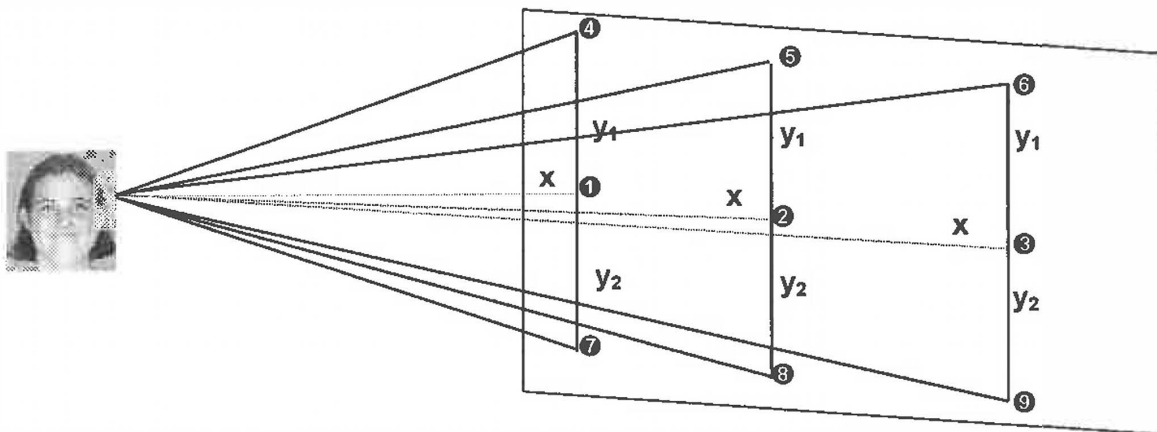


Figure 3. 5 : View of Figure 3.4 from the arrow. (Left hand side of Figure 3.4)

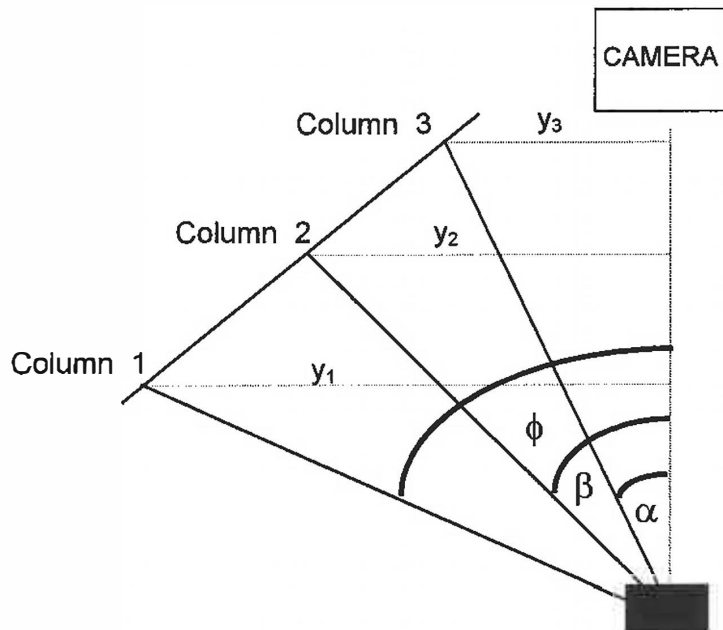


**Figure 3. 6 :** Variations in head movement in the vertical plane

**Table 3.1 :** Table of inclination and depression angles for vertical plane movement

	Column 1	Column 2	Column 3
	(m)	(m)	(m)
$y_1$	0.47	0.47	0.47
$y_2$	0.48	0.48	0.48
X	1.68	1.57	1.62
Inclination angle°	15.6	16.6	16.1
Depression angle°	15.9	17.0	16.5








**Figure 3. 7 :** Top view of experimental set-up showing head variations in the horizontal plane



**Table 3.2 :** Table of angles for horizontal plane head movement

	Column 1	Column 2	Column 3
$y_1$ (m)	1.45	N/A	N/A
$y_2$ (m)	N/A	0.96	N/A
$y_3$ (m)	N/A	N/A	0.50
Angle °	$\phi = 63.5$	$\beta = 37.6$	$\alpha = 17.3$

The angles represented in Tables 3.1 and 3.2 respectively are approximations of the position of the subject's head for the particular view being captured. These angles would be an exact representation of the head position if and only if the subjects were looking at the exact centre of the numbers on the board (see Figure 3.7). Hence, actual capturing results could vary by about 3° to 5°.

**Table 3.3 :** Typical views(normalised) captured per subject, with corresponding vertical and horizontal plane angles.

Typical views captured				
	View Number	:	1	
	Inclination angle	:	None	
	Depression angle	:	None	
	Horizontal plane	:	$\phi=63.5^\circ$	
	View Number	:	2	
	Inclination angle	:	None	
	Depression angle	:	None	
	Horizontal plane	:	$\beta=37.6^\circ$	$\frac{3}{4}$ view
	View Number	:	3	
	Inclination angle	:	None	
	Depression angle	:	None	
	Horizontal plane	:	$\alpha=17.3^\circ$	
	View Number	:	4	
	Inclination angle	:	$15.6^\circ$	
	Depression angle	:	None	
	Horizontal plane	:	$\phi=63.5^\circ$	
	View Number	:	5	
	Inclination angle	:	$16.6^\circ$	
	Depression angle	:	None	
	Horizontal plane	:	$\beta=37.6^\circ$	$\frac{3}{4}$ view
	View Number	:	6	
	Inclination angle	:	$16.1^\circ$	
	Depression angle	:	None	
	Horizontal plane	:	$\alpha=17.3^\circ$	
	View Number	:	7	
	Inclination angle	:	None	
	Depression angle	:	$15.9^\circ$	
	Horizontal plane	:	$\phi=63.5^\circ$	
	View Number	:	8	
	Inclination angle	:	None	
	Depression angle	:	$17.0^\circ$	
	Horizontal plane	:	$\beta=37.6^\circ$	$\frac{3}{4}$ view

	View Number	:	9
	Inclination angle	:	None
	Depression angle	:	16.5°
	Horizontal plane	:	$\alpha=17.3^\circ$
	View Number	:	10
	Inclination angle	:	None
	Depression angle	:	None
	Horizontal plane	:	None

### 3.3.2 Examples of images captured

The following images illustrate some of the aspects that were taken into account when building the database:

- No restraint was placed upon the subject's expression and they were allowed to tilt their heads as if that was their natural position.



**Figure 3. 8 :** Views 1-10 of subject showing various degrees of head tilt

- Subjects who wore glasses were captured with and without the glasses on. Also, some of the subjects wore hats or caps, and were captured with and without the hats or caps on.



**Figure 3. 9 :** Views 1,2,3 and 10 showing subject captured with and without glasses



**Figure 3. 10 :** Views 1,2,3 and 10 showing subject captured with and without hat on

- Subjects with long hair were allowed to have part of their face occluded.



**Figure 3. 11 :** Views 1,2,3 and 10 showing non-occluded and occluded views of subject

- Subjects were allowed to have beards



**Figure 3. 12 :** Views 1,2,3 and 10 showing subject with beard

- In order to balance the database, the subjects spanned gender and ethnic origin as evenly as possible. Subject availability played an important factor in this as well as obtaining subjects of varying ages.
- The database contained the same number of images for each person.

**A further description of the database can be found in Appendix B.**

## Chapter 4

# FACE PREPROCESSING

---

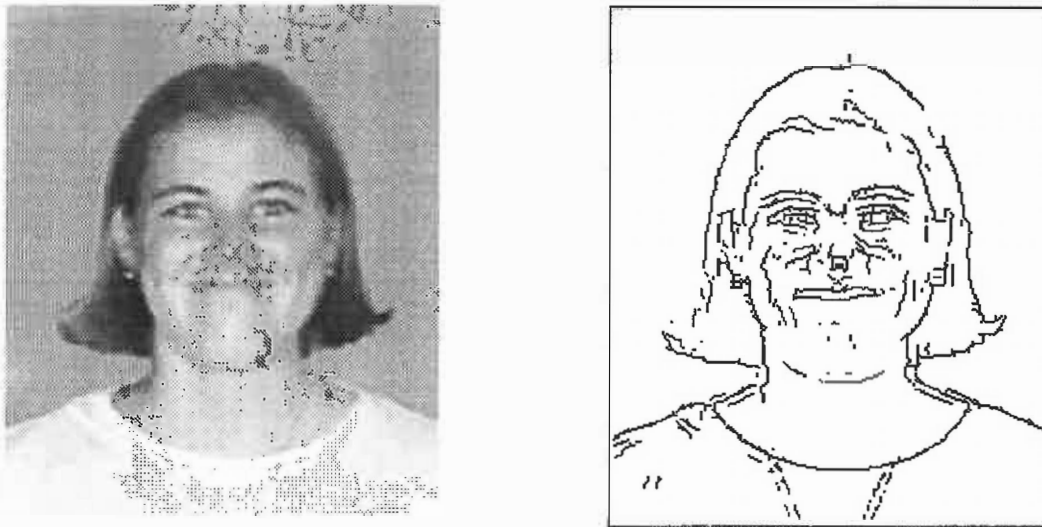
### 4.1 Introduction

Feature extraction facilitates efficient classification by a neural network, whilst face location ensures that insignificant data (i.e. part of the image that doesn't contain face information) is not used in the classification process. This is discussed in detail below. The face location technique, its advantages and disadvantages, as well as results are presented. A treatment of the DCT, its relevance to this project with respect to its decorrelation properties, and results using the DCT for feature extraction are discussed.

### 4.2 Face Location

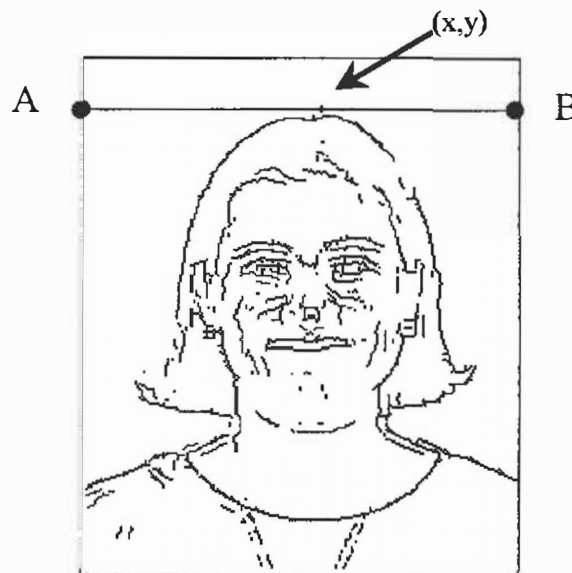
Whilst one can recognise faces despite orientation, lighting or partial occlusion, one experiences great difficulty in reproducing this process on computers[26]. One possible reason could be that humans compensate for changes in backgrounds in their face location process. The recognition of faces by a computer is inherently a pattern recognition exercise in high dimensional feature space. It is thus important that only relevant facial data be presented to the pattern recognition algorithm in a face recognition application. Separating the face from the background ensures that insignificant information (i.e. the background) is discarded. A simple method of locating faces captured against homogenous backgrounds, is described as follows:

Present a 256 (8 bit) level grey-scale image of a face to an edge detector (e.g. Canny edge detector[23]) to produce a binary image which is the edge detected output.



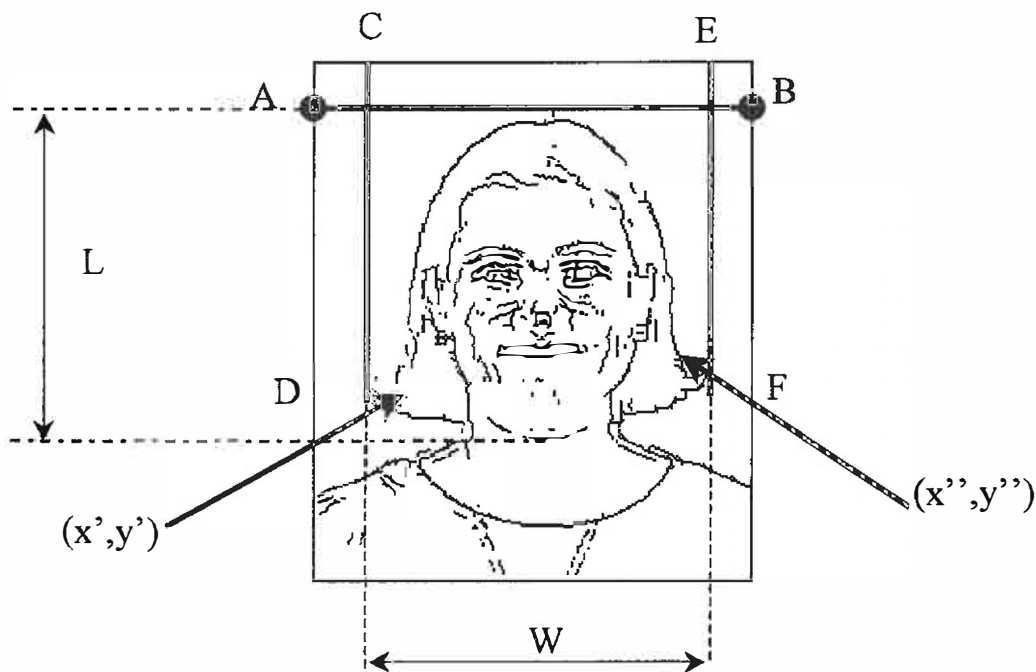
**Figure 4. 1 :** 256 grey-level image and its Canny[23] edge detected counterpart.

Translate a horizontal line from the top of the binary image until the first non-zero pixel is detected, say  $(x,y)$ . Draw a horizontal line that passes through  $(x,y)$ . (AB in Figure 4.2)



**Figure 4. 2** Line AB passes through the first non-zero pixel encountered.

Translate a vertical line whose length is approximately 0.75 the height of the image, from the left hand side of the image until the first non-zero pixel is found, say  $(x', y')$ . Draw a vertical line that passes through  $(x', y')$ . (line CD in Figure 4.3) A vertical line is translated from the right hand side of the image until the first non-zero pixel is found, say  $(x'', y'')$ . Draw a vertical line that passes through  $(x'', y'')$ , (line EF in Figure 4.3). Calculate  $W$ .



**Figure 4. 3** : Lines CD and EF isolate the sides of the image

It is clear that the difficulty associated with the simplistic technique described above is the location of the chin. A possible solution would be to use a table of values of  $W$  and corresponding measured values of  $L$ . By calculating  $W$ , one could look up a corresponding value for  $L$  and calculate the vertical position of the chin using this information. Although the proposed table lookup technique could account for scale variations to some extent, the face location technique works well only if the face is captured against a homogenous background. The orientation of the captured image is also an important factor to be considered and it can be inferred that the above algorithm would not be effective for images that are skew (i.e. rotations in the plane of the image).



Thus, a technique is required that is invariant to the scale of the image, its background and the image's orientation.

Many of the images that are presented to biological and machine vision systems are complex. Thus, these systems have to be able to cope with enormous amounts of information. However, the natural mechanisms of fixation and attention enable primates to reduce the amount of information and processing[16].

Most of the photo-receptors of the retina are located at the fovea (the part of the eye with the highest resolution) and the eyes rapidly move from one fixation point to the next. Moreover, resources are not allocated uniformly over the field of view; when a primate focuses his attention on a location, events occurring at that location are responded to more rapidly, giving rise to enhanced electrical activity and can be reported at a lower threshold[15].

These observations have inspired researchers in fields of active vision system heads and general active vision concepts and algorithms. Inspired by the intuitive notion of symmetry, researchers have introduced an interest operator as a computer vision analogue to fixation and attention[16].

#### **4.2.1 The Generalised Symmetry Transform - *Reisfeld et al*[15]**

A symmetry transform is presented by *Reisfeld et al*[15] which assigns a symmetry magnitude and orientation to every pixel at a low level vision stage which follows edge-detection. A symmetry map which, according to *Reisfeld et al*[15], is a new kind of edge map is computed where the magnitude and orientation of the edge depends on the symmetry associated with each pixel. Strong symmetry edges are natural interest points, while linked lines are symmetry axes. It is claimed that since the symmetry transform can be applied straight after the stage of edge detection, it can be used to detect higher level processes such as location and recognition, and can serve as a guide for locating objects[16].

The main idea behind the General Symmetry Transform is described below[15]:

The algorithm begins with an edge map and assigns a magnitude  $M$  that estimates the probability that there is a symmetric spatial configuration of edges around it. An orientation  $\alpha$ , that points in the direction of the main axis of symmetry around the pixel is also assigned. These two steps are applied to every pixel in the image. Thus, for example, the pixel (or pixels) in the centre of a circular, elliptic or rectangular area surrounded by edges, will be assigned a high value of  $M$ . This results in a symmetry map, where every pixel has a value and the highest peaks of symmetry could be detected[16].

*Reisfeld et al*[15] used the GST successfully for the location of the eyes and nose in a face recognition application.

The Generalised Symmetry Transform is context free in the sense that it operates on pixels and not on known objects. The following steps are suggested by *Reisfeld et al*[15] for the detection of the eyes and nose of a face:

- Computation of the symmetry magnitude and orientation.
- Computation of Radial Symmetry(RS). While the regular symmetry definition does not depend on the specific spatial organisation that contribute to the symmetry measure, this measure assigns high values to pixels that are surrounded by circular isotherms.
- Detection of the highest peaks of regular and radial symmetry in the image.
- Detection of the mid-line of the face by finding the peak in the auto-correlation function of the edge image.
- Detection of the eyes and mouth by including geometric considerations. This is carried out by finding the location of the highest peaks of the symmetry values, with the assumption that the eyes should be on either side of the mid-line[16].

### 4.2.2 Definition of the General Symmetry Transform

*Reisfeld et al*[15].

Mathematically, an object is regarded symmetric if it is invariant to the application of certain transformations, called symmetry transformations. The symmetry transform developed by *Reisfeld et al*[15] does not require knowledge of the object's shape. The transform applies a continuous measure to each point in the image, rather than a binary symmetry label.

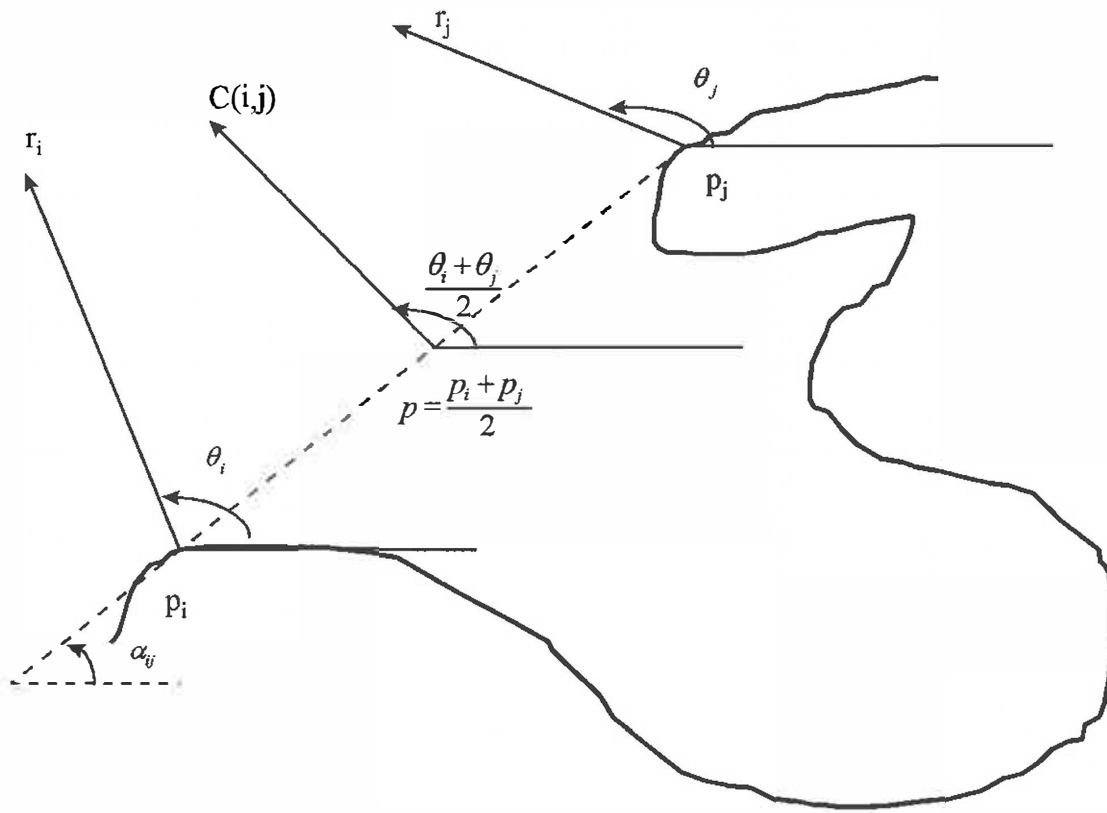
*Reisfeld et al*[15] defines the transform as follows : Firstly, a symmetry measure for each point is defined. Let  $p_k = (x_k, y_k)$  be any point ( $k=1\dots N$ ) and denote by  $\nabla p_k = (\frac{d}{dx} p_k, \frac{d}{dy} p_k)$  the gradient of the intensity at point  $p_k$ . A vector  $v_k = (r_k, \theta_k)$  is associated with each  $p_k$  such that

$$r_k = \log(1 + \|\nabla p_k\|) \quad 4-1$$

and

$$\theta_k = \arctan \left( \frac{\frac{d}{dy} p_k}{\frac{d}{dx} p_k} \right) \quad 4-2$$

For each two points  $p_i$  and  $p_j$ ,  $l$  is defined as the line passing through them, and  $\alpha_{ij}$  the angle counter clockwise between  $l$  and the horizon (see Figure 4.4).



**Figure 4. 4** : The contribution to symmetry of the gradients at  $p_i$  and  $p_j$

A set  $\Gamma(p)$ , a distance weight function  $D_\sigma(i,j)$  and a phase weight function  $P(i,j)$  is defined as follows[15] :

$$\Gamma(p) = \left\{ (i,j) \left| \frac{p_i + p_j}{2} = p \right. \right\} \quad 4-3$$

$$D_\sigma(i,j) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\|p_i - p_j\|}{2\sigma}} \quad 4-4$$

$$P(i,j) = (1 - \cos(\theta_i + \theta_j - 2\alpha_{ij}))(1 - \cos(\theta_i - \theta_j)) \quad 4-5$$

The contribution of the points  $p_i$  and  $p_j$  is defined as follows:

$$C(i, j) = D_\sigma(i, j)P(i, j)r_i r_j \quad 4-6$$

This measure can be easily normalised, and reflects the fact that each of its components modulates the other ones[16].

The symmetry magnitude or isotropic symmetry  $M_\sigma(p)$  of each point  $p$  is

$$M_\sigma(p) = \sum_{(i,j) \in \Gamma(p)} C(i, j) \quad 4-7$$

which averages the symmetry value over all orientations. The direction of the contribution of  $p_i$  and  $p_j$  is

$$\varphi(i, j) = \frac{\theta_i + \theta_j}{2} \quad 4-8$$

The symmetry direction is defined as  $\phi(p) = \varphi(i, j)$  such that  $C(i, j)$  is maximal for  $(i, j) \in \Gamma(p)$ . Therefore, the symmetry of point  $p$  is defined as

$$S_\sigma(p) = (M_\sigma(p), \theta(p)) \quad 4-9$$

The demand that the symmetry transform be local is reflected by the Gaussian weight distance  $D_\sigma(i, j)$ . Different values of  $\sigma$  imply different scales. The Gaussian has circular isotherms, i.e. it has no preferred orientation[16].

The phase weight function  $P(i,j)$  is composed of two terms. The first term  $1 - \cos(\theta_i + \theta_j - 2\alpha_{ij})$  allows maximum symmetry to be achieved when  $(\theta_i - \alpha_{ij}) + (\theta_j - \alpha_{ij}) = \pi$  i.e. when the gradients at  $p_i$  and  $p_j$  are oriented in the same direction towards each other. This is consistent with the intuitive notion of symmetry. This expression decreases continuously as the situation deviates from the ideal one. The same measure is achieved when the gradients are oriented towards each other or against each other. The first situation corresponds to symmetry within a dark object on a light background and the second corresponds to symmetry of a light object on a dark background. For the purposes of this study, only the case of gradients facing each other are considered, which correspond mainly to dark objects on a brighter background[15].




The second term,  $1 - \cos(\theta_i - \theta_j)$  is introduced since the first term attains its maximum whenever  $(\theta_i - \alpha_{ij}) + (\theta_j - \alpha_{ij}) = \pi$ . This includes  $\theta_i - \alpha_{ij} = \theta_j - \alpha_{ij} = \pi$ , which occurs on a straight edge. The current expression compensates for this situation[16].

The term  $r_i r_j$  is high when there is a strong correlation between two large gradients. Gradients rather than intensities are valid since the focal interest are edges that relate to object borders. For instance, a uniform intensity wall is highly symmetric. In natural scenes, the logarithm of the magnitude instead of the magnitude itself is used since it reduces the differences between high gradients and therefore the correlation measure is less sensitive to very strong edges[16]. The above defined transform detects reflectional symmetry. It is invariant under 2-D rotation and translation transformation. It is quite effective in detecting skewed symmetry as well.

### 4.2.3 Performance of the General Symmetry Transform

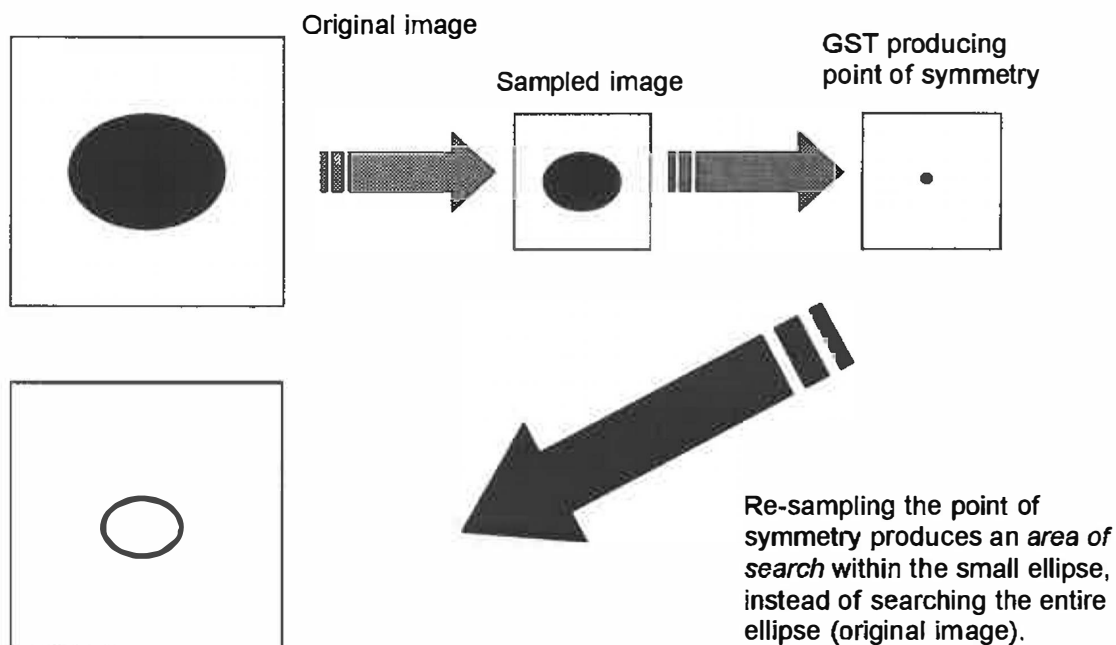
The transform was coded in C (see Appendix C for the algorithm and code listing), and compiled for Linux running on a Pentium™ 133 MHz computer. An image of size 215x215 pixels took approximately 113 minutes to be transformed using isotropic symmetry. Additional radial symmetry would take longer. The amount of time required by the transform to effectively find points of symmetry in the facial images was not practical. Hence, a speed enhancement study was performed on the GST, and as a first step, images were re-sized to 100x100 and 50x50 pixels. The following results were obtained using images digitised to 8 bit accuracy of dimensions 100x100.

**Table 4. 1:** Performance of GST for image of size 100x100.

Image	Comments
	Original image of dimension 100x100 pixels.
	GST output of original image. The transform took approximately 70.34 minutes. Note that the eye-brow and eye region has been assigned a higher value of symmetry.
	This output of the GST is obtained by thresholding (values below 60 were set to 0). The amount of time required by the transform to produce this image was approximately 30 minutes.

Another technique of speed enhancement was implemented via thresholding. Thresholding was implemented by setting all pixel values below a certain value to 0, and leaving remaining pixel values unchanged. This allows homogenous areas to form in the image, thus speeding up the transform (homogenous areas have zero gradient). This had a noted improvement on the speed of the transform, however it was still not fast enough (see Table 4.1).

Though there is a degradation of image quality when images are sampled to smaller dimensions, it was implemented to find the highest points of symmetry. These *points* of symmetry become *areas* of symmetry when the image is sampled to its original dimensions. Instead of presenting an entire 215x215 image to the GST, areas of the image would have been presented, thus reducing the time taken for the GST to operate on the original large image. This is explained graphically in the following figure:



(a)



(b)



(c)

**Figure 4. 5 :** a) Graphical description of sub-sampling image to obtain a smaller image, presentation to the GST, re-sampling points of symmetry to obtain "area of search" in original image.  
 b) 256 level grey scale image used as input to the General Symmetry Transform.  
 c) Output obtained from the General Symmetry Transform.



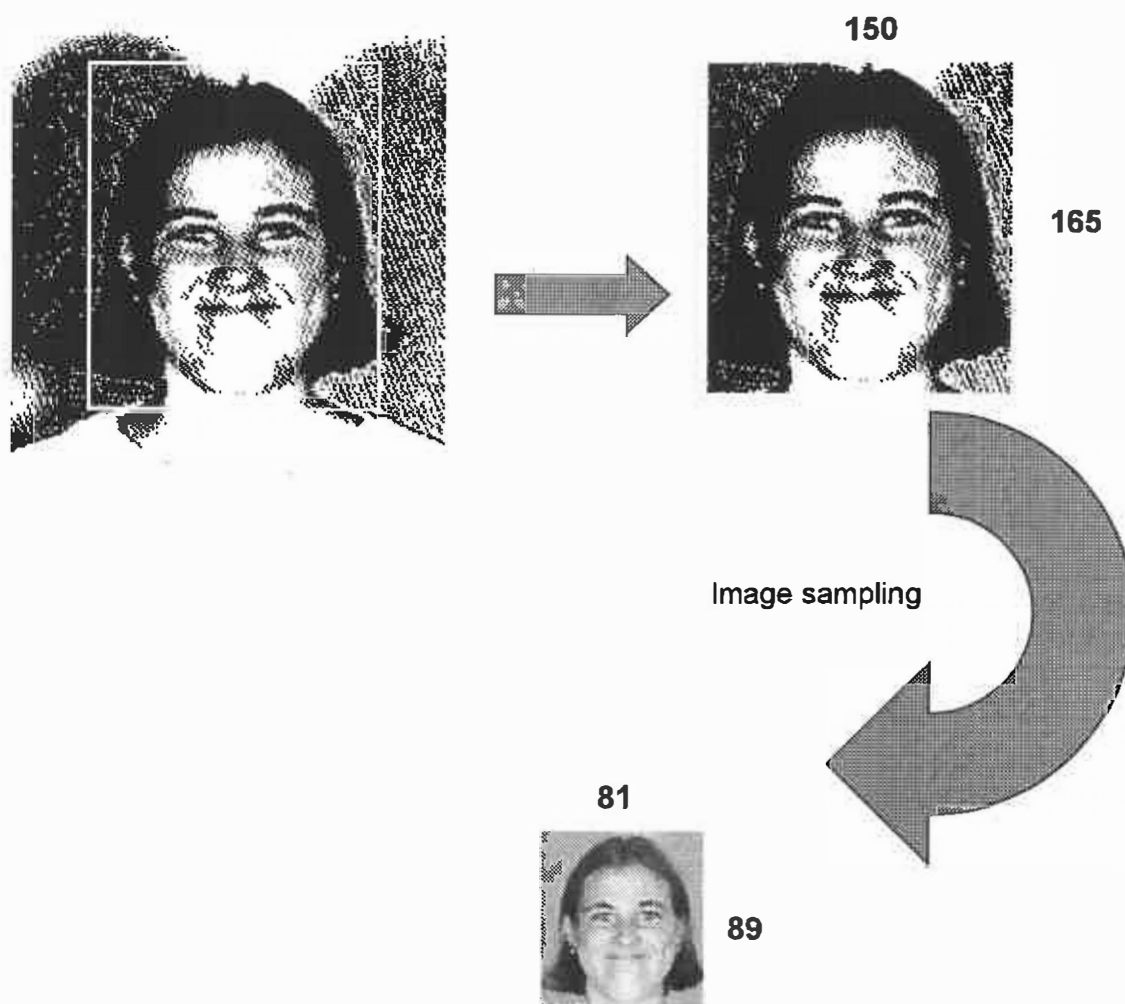
However, the time taken for the GST to operate on the sampled image (i.e. the smaller image) has to be taken into consideration as well. Images of size 50x50 pixels took approximately 5 minutes to complete. Although this is a vast improvement over images of size 100x100 pixels, it is still not practical.

Figure 4.5b and 4.5c are examples of what the output would look like after an area of search is obtained. Figure 4.5b represents an area of search that was obtained from points of symmetry. Figure 4.5c may not look like an image containing much information. However, the black dots that appear on the top half of the image, actually correspond to the exact location of the eyes (Figure 4.5a). The dots that are slightly lighter in colour, found in the middle of the image, correspond to the location of the nostrils, while the last few dots that occur right at the bottom of Figure 5b, are the location of the lips. However, this technique was not pursued, due to the amount of time required to process even small images (e.g. 50x50 images took approximately 5 minutes).

This transform is advantageous since it is independent of light levels, scales, orientation and is context free. Since no a priori information of the face is used, the algorithm is computationally intensive as is demonstrated in Table 4.1. Although the GST is highly accurate, it is slow. This is the principal reason that it was not used for the normalisation procedure. This stage of the automatic face recognition procedure has to be performed accurately and without delay. Hence, the face images were normalised manually.

#### 4.2.4 Manual Normalisation

Examination of the face database (see Appendix B) reveal that shapes of subjects' faces vary i.e. faces are "angular", "round", "square" and "oval shaped". It is imperative that the manual normalisation technique take this factor into account. It was also observed that the majority of faces fit into a 150x165 pixel rectangle. A rectangle of these dimensions was manually shifted over the image until the rectangle fitted over the face. This is best shown by way of an example:



**Figure 4. 6:** Graphical description of manual normalisation procedure.

Once the face was located under the rectangle, it was "cut out" of the original image and saved separately. As a result of hardware space constraints, the image was resized to 81x89 pixels by image sampling. The aspect ratio of the image is kept constant. The manual procedure is disadvantageous since it is subjective and liable to human error in

the normalisation process. The manual normalisation procedure was still faster than the general symmetry transform.

### 4.3 The Discrete Cosine Transform (DCT)

The dimensionality reduction technique employed in this project, reasons for choosing the DCT, fast algorithms of the DCT, as well as the performance of the DCT are discussed.

Suppose a signal in the form of a pure sinusoid is to be transmitted. One obvious way to do this would be to sample the waveform at more than twice the frequency of the signal and then transmit each sampled point in a sequential manner. However, to construct a deterministic sinusoid, all one needs is 5 pieces of information, viz., magnitude, phase, frequency, starting time and the fact that it is a sinusoid. From an information theoretic point of view, the sampled values of the deterministic waveform are highly correlated and the information content of the transmitted signal is low. On the other hand, the five pieces of information (*as mentioned above*) are completely uncorrelated and have exactly the same amount of information content as the total number of sampled values being transmitted. [18]

A natural question to pose is whether it is possible to take the  $N$  sampled points in the transmission and transform them to the five uncorrelated pieces of information.

The Karhunen-Loeve Transform (KLT) is the optimal method for reducing redundancy in a data-set [19]. This transform is said to be optimal because it has the following properties[17] :

1. It completely de-correlates the signal in the transform domain.
2. It minimises the mean square error (MSE) in bandwidth reduction or data compression.
3. It contains the most variance (energy) in the fewest number of transform coefficients.
4. It minimises the total representation entropy of the sequence.

Practical implementation of the KLT involves the estimation of the auto-covariance matrix of the data sequence, its diagonalisation and the construction of the basis vectors. The inability to pre-determine the basis vectors in the transform domain has made the KLT an ideal but impractical tool [17].

One may ask whether there are pre-determined basis functions that are good approximations to the KLT. Studies were undertaken to examine the diagonalisation of matrices that are asymptotically equivalent to the auto-covariance matrix of the KLT[17]. Such studies have led to the construction of other discrete transforms. Although the KLT provides no easy solutions to the problem of actual decorrelation, it does provide a benchmark against which other discrete transforms may be judged[18].

### 4.3.1 Motivation for use of the Discrete Cosine Transform

Wilder[9] implemented a face recognition system that used one-dimensional transforms. Values of pixels that were added along the horizontal and vertical directions of the image were used as input to the transforms. This gave rise to two one-dimensional signatures. Three orthonormal transforms: the KLT, DCT and the Hadamard transform were applied to raw input feature data.

<sup>1</sup>“There is a practical implication of the decorrelation property of these transforms in the face recognition application. Since every transform component is a linear combination of all input features, pattern distortion due to alteration of a group of neighbouring features is distributed over all the output features. Hence, addition of glasses or changing from a smile to a frown will not have a catastrophic effect on the transformed pattern vector. For classification purposes, the low spatial frequency and selected high frequency components are retained to differentiate these patterns that are similar except for some small details. The remaining components generally the highest

---

<sup>1</sup> Wilder, “Face Recognition Using Transform Coding of Gray Scale Projections and the Neural Tree Network”, *Artificial Neural Networks for speech and Vision – Edited by R.J. Mammone*, pgs 523-525.

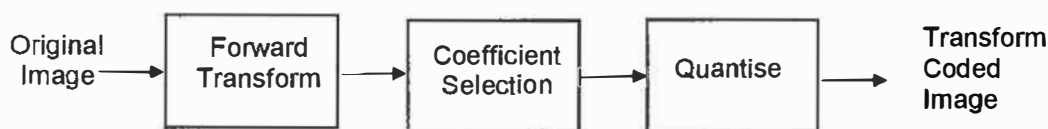
*frequency terms contain very little energy, and can be discarded without sacrificing information that is significant for classification."*

It was reported that all three transforms produced similar results in terms of class separability and recognition rate. However, the DCT and KLT performed slightly better than the Hadamard. The DCT became the transform of choice since unlike the KLT it has a fast algorithm that can be implemented[9].

The behaviour of the DCT in relation to the optimal KLT has been extensively documented by *Rao and Yip*[17]. They have shown that the DCT is asymptotically equivalent to the KLT.

### 4.3.2 The 2-D Discrete Cosine Transform

Probably the most common use of the two-dimensional transforms is image compression. Many transforms have been tried in compression and their performance can be compared by the fidelity of the recovered image to the original. The general flow of the image data in transform compression is as follows:



**Figure 4. 7 :** Transform Compression

The sampled image is transformed and several of the resulting coefficients are selected, according to the variance distribution of each coefficient (§ 4.3.5). The remaining coefficients are quantised with fewer bits. For the recovery of the image, the opposite process is performed: coefficients are re-quantised to the original number of bits, missing coefficients are replaced by fixed values (for instance 0) and the inverse transform is performed. The fidelity of the process can be measured using the difference in intensity level between the original and recovered images at each point in the array. Since the

DCT gives a MSE result near the theoretical limit of the KLT, the DCT is very popular for image compression.

### 4.3.3 2-D DCT Implementation by reduction to a 1-D DCT [17]

*Rao and Yip*[17] present many algorithms for the implementation of the a 2-D DCT. One such algorithm involves the computation of the 2-D DCT via a series of 1-D calculations. The algorithm is explained as follows:

Let  $[g]$  be an  $[M \times N]$  matrix representing the two dimensional data and  $[G]$  be the 2-D DCT of  $[g]$ . Then, the  $uv$  element of  $[G]$  is given by:

$$G_{uv} = \frac{2c(u)c(v)}{\sqrt{MN}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g_{mn} \cos\left[\frac{(2m+1)u\pi}{2M}\right] \cos\left[\frac{(2n+1)v\pi}{2N}\right] \quad 4-10$$

The separability property of the 2-D DCT can be illustrated as follows :

$$G_{uv} = \sqrt{\frac{2}{M}}c(u) \sum_{m=0}^{M-1} \left\{ \sqrt{\frac{2}{N}}c(v) \sum_{n=0}^{N-1} g_{mn} \cos\left[\frac{(2n+1)v\pi}{2N}\right] \right\} \cos\left[\frac{(2m+1)u\pi}{2M}\right] \quad 4-11$$

The inner summation is an N-point 1-D DCT of the rows of  $[g]$ , whereas the outer summation represents M point 1-D DCT of the semi-transformed matrix.

This means that the 2-D  $[M \times N]$  DCT can be implemented by M N point DCTs along the rows of  $[g]$ , followed by N M point DCTs along the columns of the matrix after the row-transformation.

The following 1-D DCT algorithm is presented in [17]:

Let  $\{x(n)\}$ ,  $n=0,1,\dots,N-1$  be a given sequence. Then an extended sequence

$\{y(n)\}$  symmetric about the  $\frac{(2N-1)}{2}$  point can be constructed so that

$$\begin{aligned} y(n) &= x(n) & n &= 0, 1, \dots, N-1 \\ &= x(2N-n-1) & n &= N, N+1, \dots, 2N-1 \end{aligned} \quad 4-12$$

Let  $W_{2N}$  be used to denote  $e^{\frac{-j2\pi}{2N}}$ . Thus it can be seen that the DFT of  $\{y(n)\}$  is given by

$$Y(m) = \sum_{n=0}^{2N-1} y(n) W_{2N}^{nm} \quad 4-13$$

and is easily reduced to

$$\begin{aligned} Y(m) &= \sum_{n=0}^{N-1} x(n) W_{2N}^{nm} + \sum_{n=N}^{2N-1} y(n) W_{2N}^{nm} \\ &= \sum_{n=0}^{N-1} x(n) W_{2N}^{nm} + \sum_{n=N}^{2N-1} x(2N-n-1) W_{2N}^{nm} \\ &= \sum_{n=0}^{N-1} x(n) W_{2N}^{nm} + \sum_{n=0}^{N-1} x(n) W_{2N}^{(2N-n-1)m} \\ &= \sum_{n=0}^{N-1} x(n) \left[ W_{2N}^{nm} + W_{2N}^{-(n+1)m} \right] \quad m=0, 1, \dots, 2N-1 \end{aligned} \quad 4-14$$

If both sides equation of equation 4-13 are multiplied by a factor of  $\frac{1}{2} W_{2N}^{m/2}$ , this yields

$$\frac{1}{2} W_{2N}^{m/2} Y(m) = \sum_{n=0}^{N-1} x(n) \cos \left[ (2n+1) \frac{m\pi}{2N} \right] \quad m=0, 1, \dots, N-1 \quad 4-15$$



The one-dimensional DCT of a sequence  $\{u(n), 0 \leq n \leq N-1\}$  is defined as:

$$v(km) = \alpha(k) \sum_{n=0}^{N-1} u(n) \cos \left[ \frac{\pi(2n+1)km}{2N} \right], \quad 0 \leq k \leq N-1$$

where

4- 16

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

Comparing equation (4-16) and (4-15), it is easily seen that except for the required scaling factors in (4-16), (4-15) is the DCT of the  $N$  point sequence  $\{x(n)\}$ . Hence, according to *Rao and Yip*[17], the one dimensional DCT can be implemented via the DFT. The `dct2` function of Matlab's Image Processing toolbox uses the algorithm described above and was thus used.

#### 4.3.4 Results of the 2-D Discrete Cosine Transform

The images of the subjects that were used to illustrate the 2-D DCT are depicted in Figure 4.8. The DCT was applied to each of these images (view number 3).



**Figure 4. 8:** Subjects A,B and C respectively

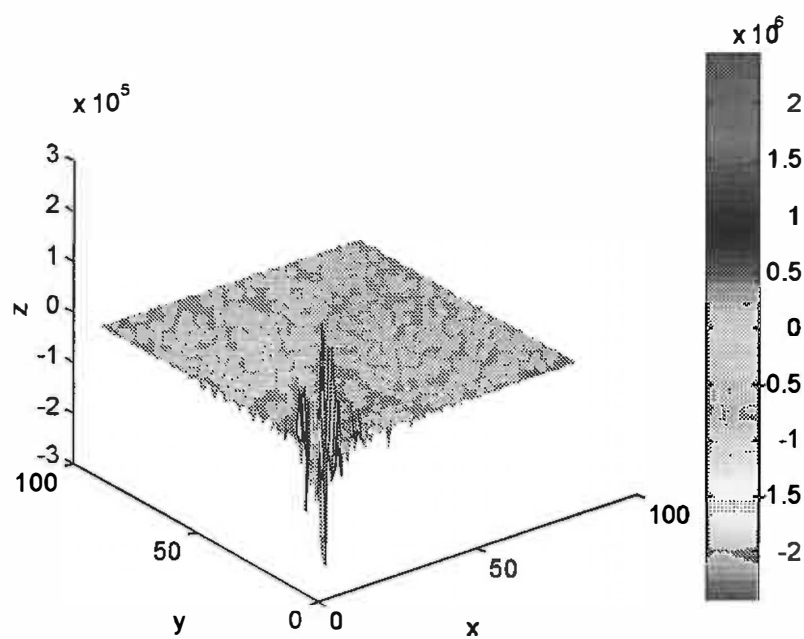


Figure 4. 9: 2-D DCT of Subject A

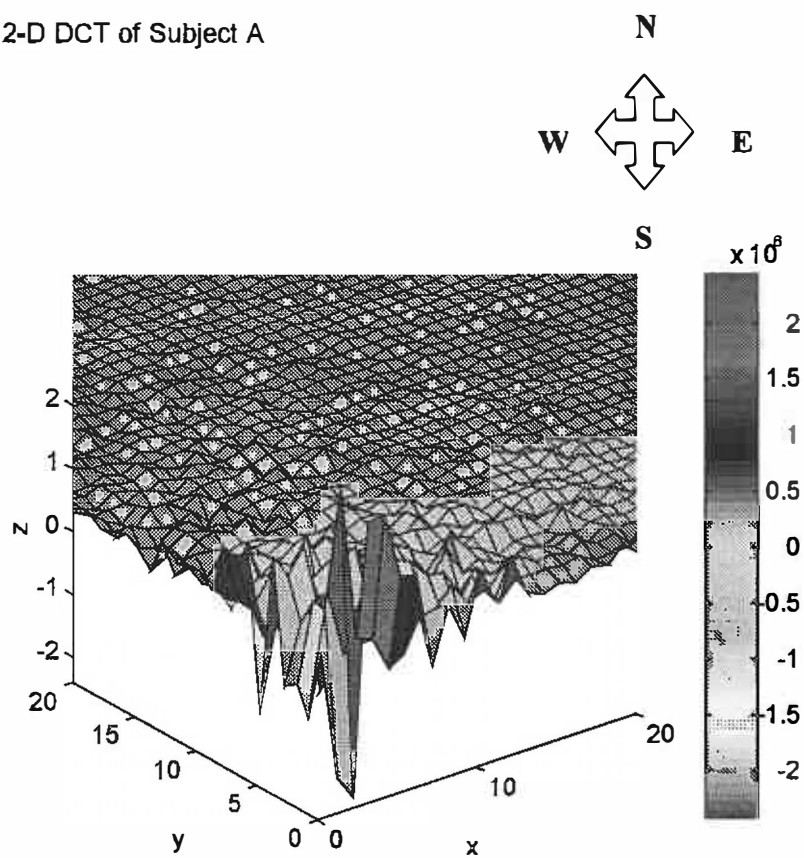


Figure 4. 10: Zoomed view of Figure 4.9

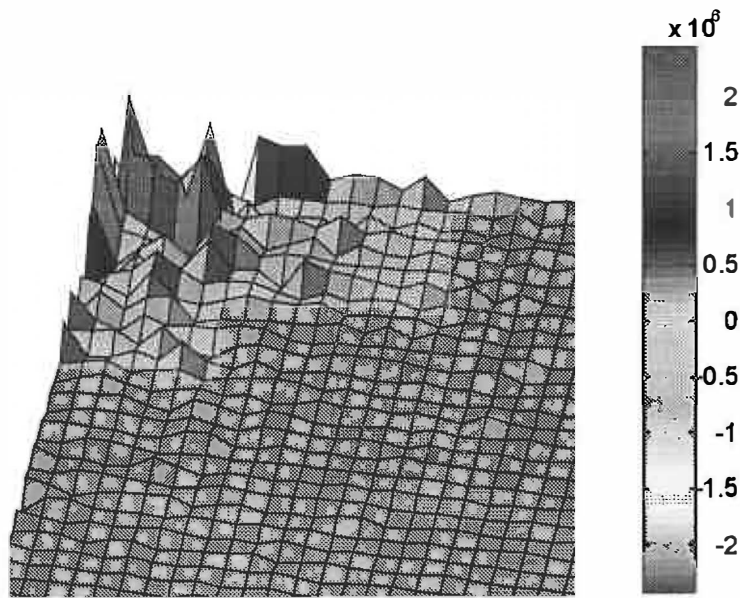


Figure 4. 11: View of Figure 4.10 looking from the North.

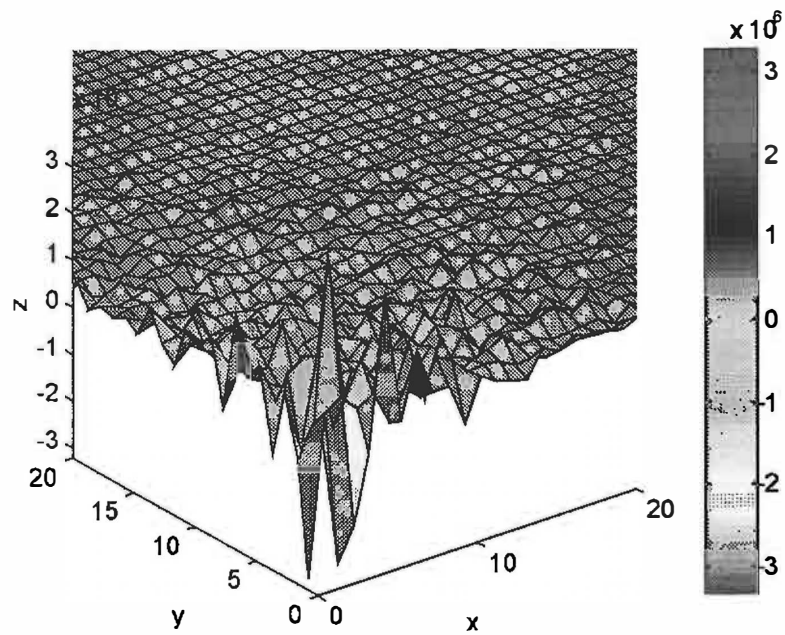
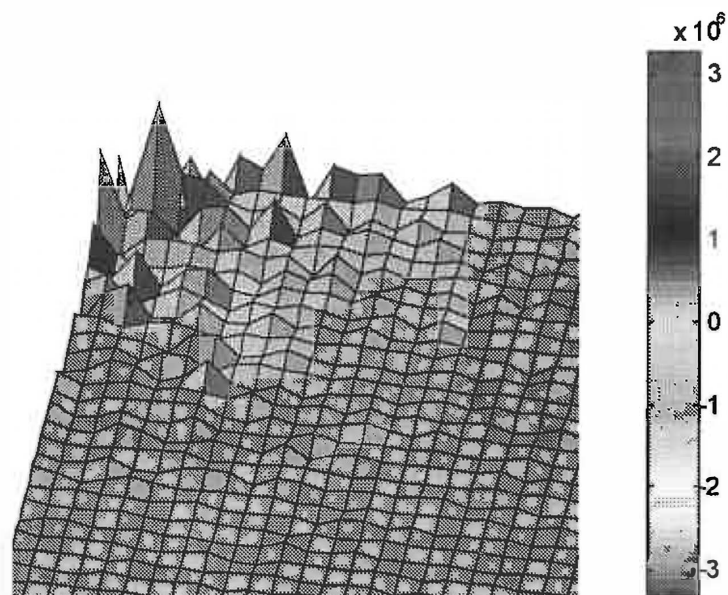
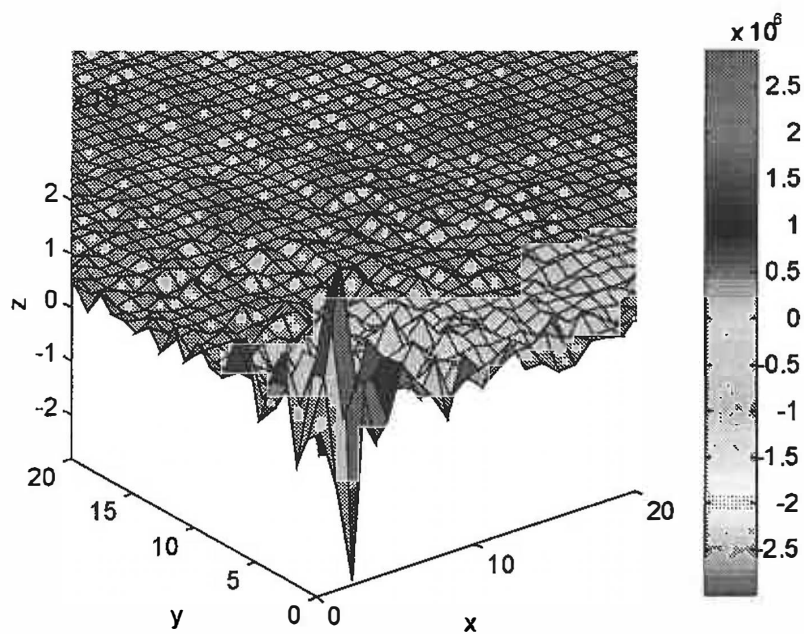


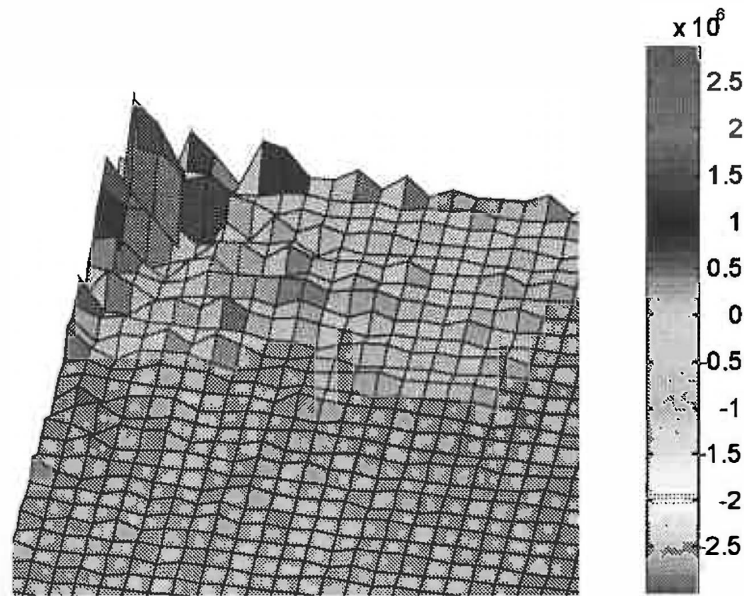
Figure 4. 12: 2-D DCT of Subject B (zoomed view)



**Figure 4. 13:** View of Figure 4.12 looking from the North (see Figure 4.10 for direction definition)



**Figure 4. 14 :** 2-D DCT of Subject C



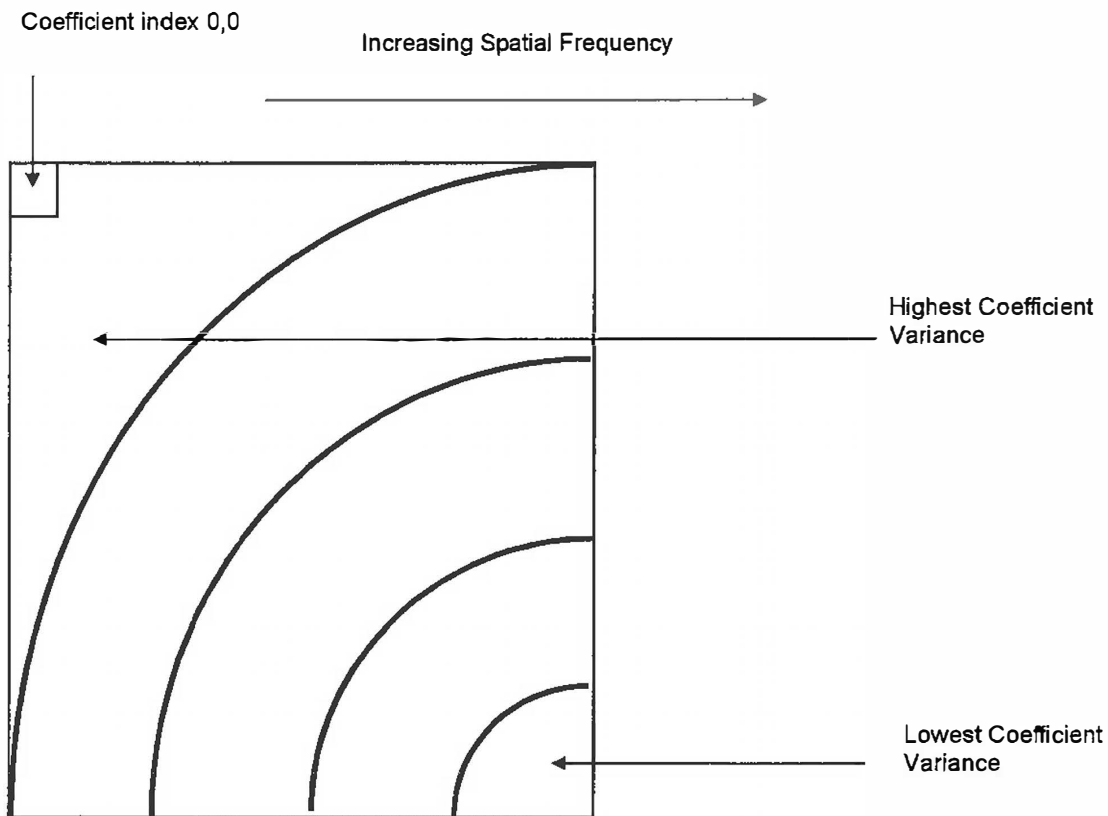
**Figure 4. 15** : View of Figure 4.14 looking from the North (see Figure 4.10 for direction definition)

### 4.3.5 Choosing the DCT coefficients

The reason image data can be compressed by large factors and successfully recovered with small errors is the large amount of redundancy in typical images. If an array of numbers has redundancy, it is theoretically possible to give the same information with less numbers. The purpose of performing the transform is to develop a set of numbers whose values are uncorrelated (i.e. each number in the array gives new information not given by the other numbers). Since the same information content is to be represented in the original and transformed arrays, some numbers in the transformed array give little or no information about the image and can be discarded.

The true measure of usefulness of a given coefficient is its variance over a set of images. Say coefficient  $k$  which resides at row  $m$  and column  $n$  does not change over a set of images, then it does not convey much information, and thus be discarded. Conversely, if a coefficient has high variance over the set, then it should not be discarded. With the

cosine transform, variances over the typical picture sets tend to have constant variance contours as shown in the following figure[19].



**Figure 4. 16:** Lines of constant variance in typical transformed image (reproduced from [19]).

The high-variance coefficients tend to be near the origin and the  $u$  and  $v$  axes.

In image compression most unitary transforms have a tendency to pack a large fraction of the average energy of the images into a relatively few components of the transform coefficients (i.e. the energy compaction property)[26]. Since the total energy is preserved, this means many of the transform coefficients will contain very little energy.

*Rao and Yip*[17] states that since the trace of the auto-covariance matrix in the transform domain for any unitary transform is invariant, one can judge the performance of a discrete transform by its variance distribution. They elaborate that since the variances represent the energy or information content of the corresponding transform

coefficients, the transform coefficients with larger variances are candidates containing significant features in a pattern recognition application.

For a two-dimensional random field  $U(m,n)$  whose mean is  $\mu_u(m,n)$  and the covariance is  $r(m,n;m',n')$ , its transform coefficients  $v(k,l)$  satisfy[29] :

$$\sigma_v^2(k,l) = E[|v(k,l) - \mu_v(k,l)|^2] \quad 4-17$$

Thus, the variances of the coefficients of the DCT are defined as above. Ten images of the same person were used to draw up a “variance map”. The variance map is a graphical representation of the variances of the coefficients, and shows where coefficients with the greatest variance lie.



**Figure 4. 17** : Views 1-9 were used in the calculation of the “variance map” for the subject.

Equation (4.17) was used to calculate the variance of the coefficient  $(x,y)$  over 9 images. Figures 4.18 – 4.22 are a variance maps of subject A (see Figure 4.8), where the  $(x,y)$  position gives the position of the coefficients in the 9 images, while the  $Z$  axis reflects the variance of that coefficient.

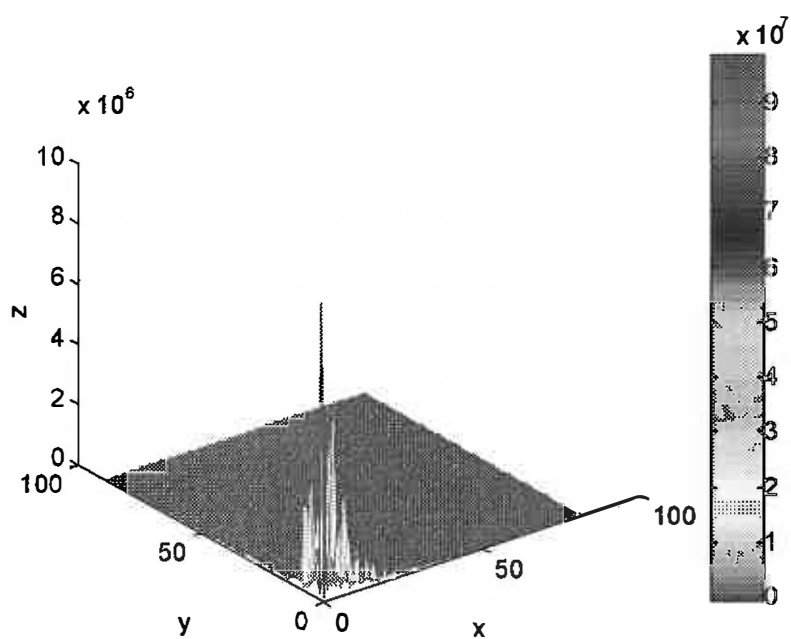


Figure 4. 18 : "Variance map" of Subject A, using views 1-10.

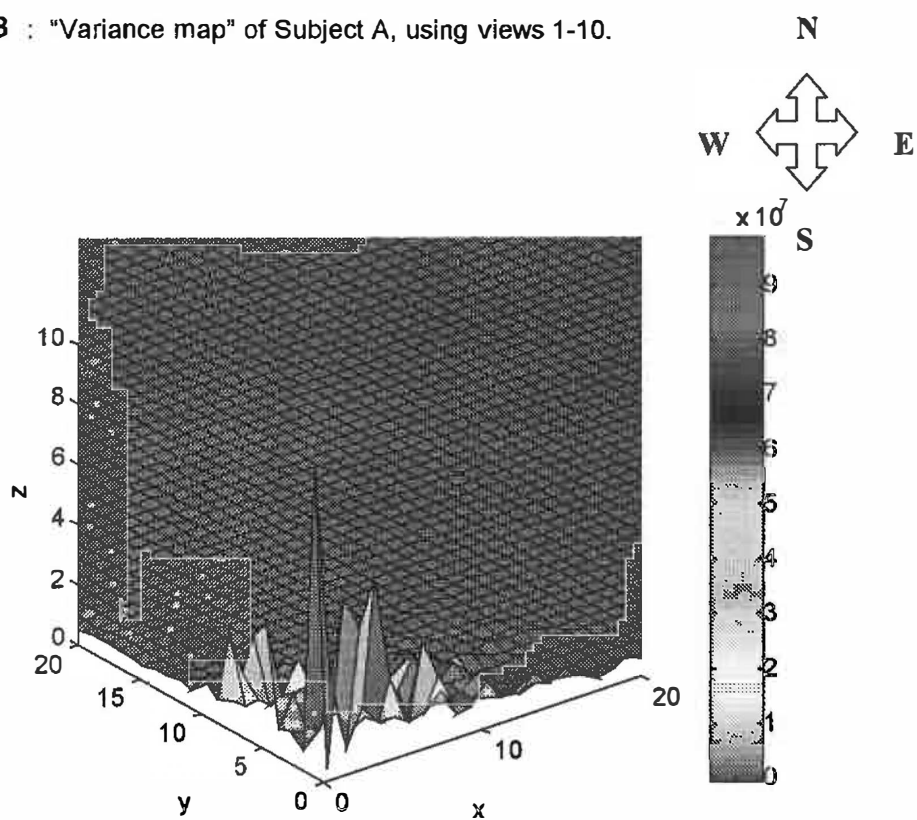


Figure 4. 19 : Zoomed view of Figure 4.18.



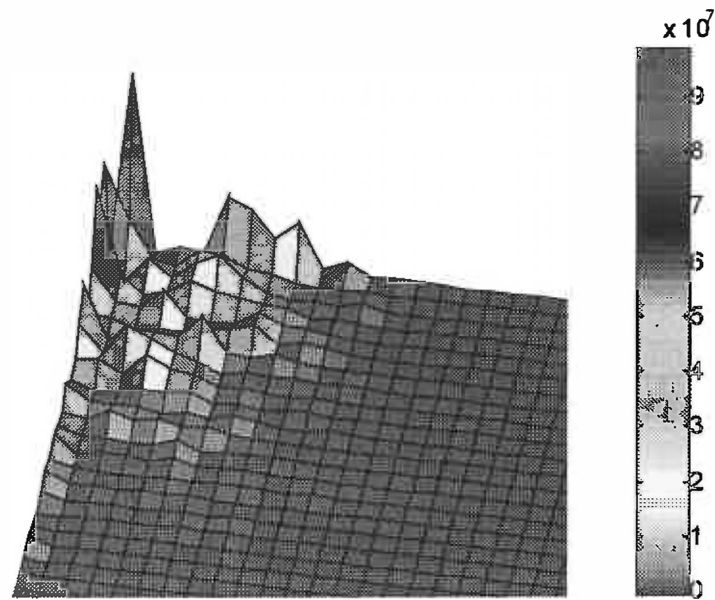


Figure 4. 20 : View of Figure 4.19, looking from the North.

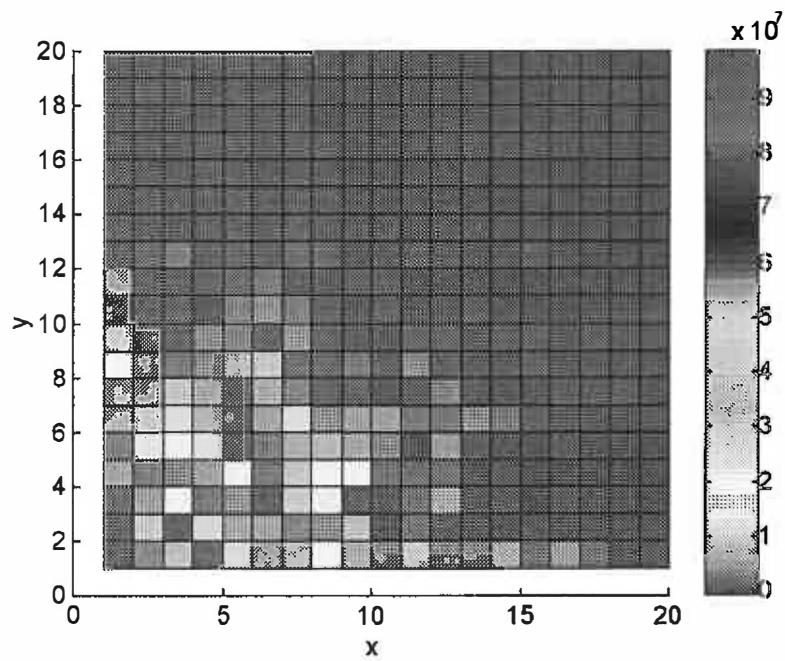


Figure 4. 21 : Top view of Figure 4.20.

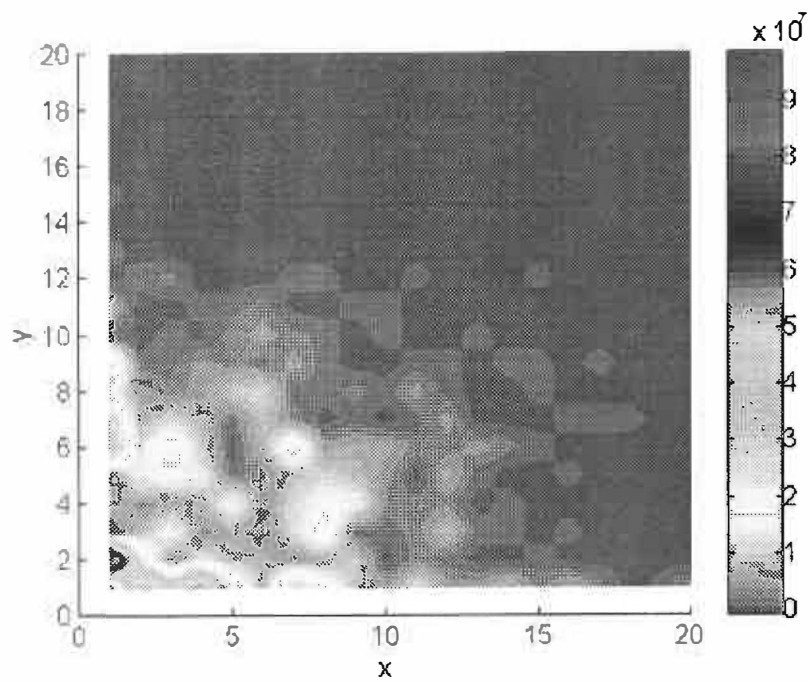


Figure 4. 22 : Interpolated top view of Figure 4.20

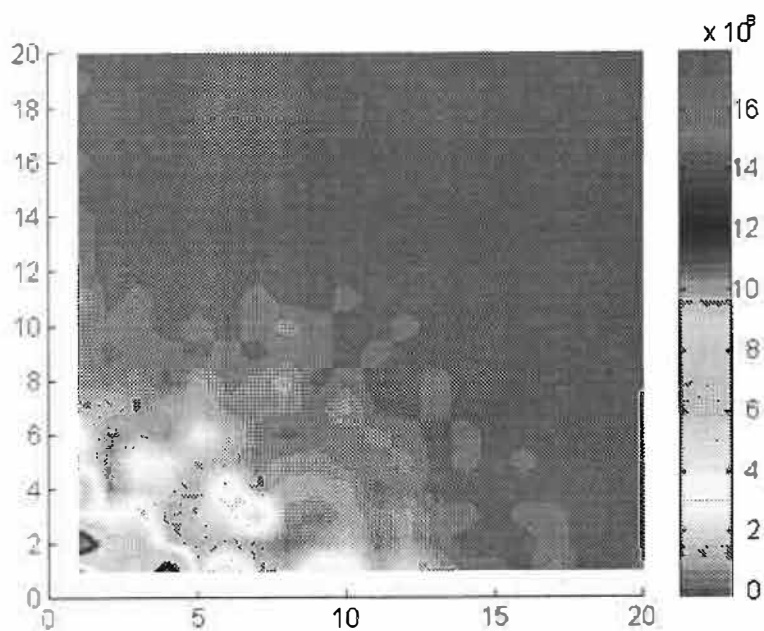


Figure 4. 23 : Top view of variance map for Subject C.

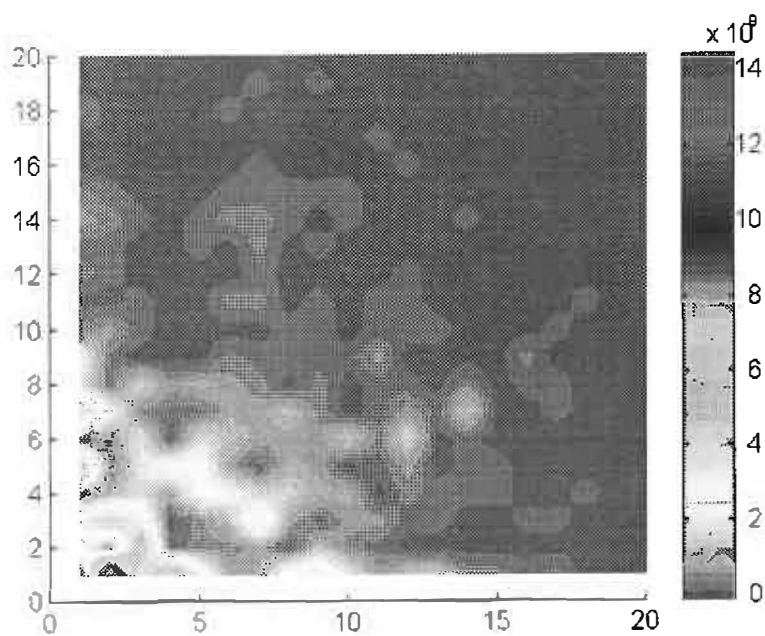


Figure 4. 24 : Top view of variance map for subject B.

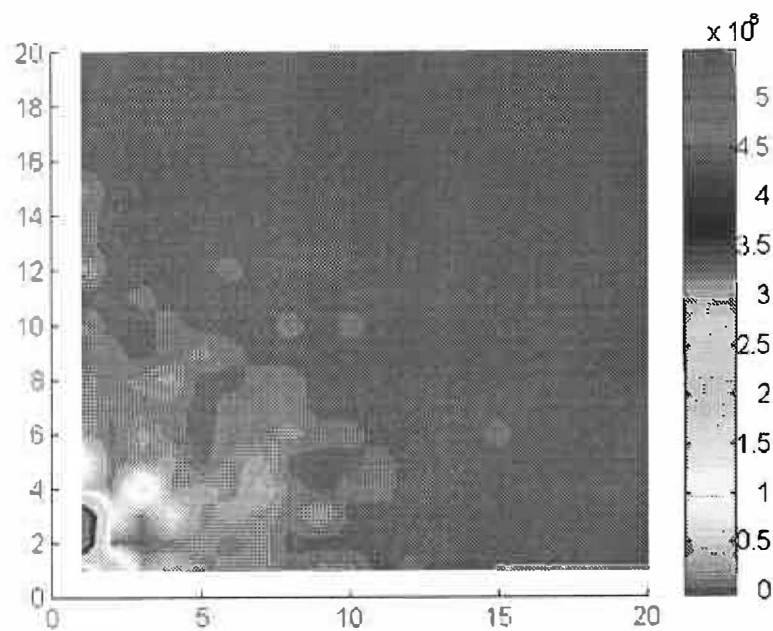


Figure 4. 25 : Top view of "variance map" for the entire database.

The examination of Figures 4.22-4.24 reveal that the coefficients with the highest variance lie in the region near (0,0). Closer study of the figures shows that coefficients that lie (approximately) in a square of 10x10 pixels contain the greatest amount of variance. This information is helpful for intra-class variances. In any pattern recognition exercise, information regarding inter-class variances, or inter-class pattern boundaries is far more useful. A further variance map was calculated, based on the entire database. Figure 4.25 reveals that coefficients that lie (approximately) in an 8x8 triangle contain significant inter-class variance information.

# Chapter 5

## CLASSIFICATION TECHNIQUE

---

### 5.1 Introduction

A general understanding of artificial neural networks is assumed but a short description of the back-propagation network is presented in this chapter to define the terminology used. The radial basis function network with dynamic decay adjustment and the counter-propagation networks are not as popular and are described in more detail. A full treatment of neural networks may be found in [28]. This chapter thus discusses the back-propagation network and its importance to the project as well as presenting a detailed description of the counter-propagation and radial basis function networks.

Artificial Neural Networks are biologically inspired and contain a large number of simple processing elements that perform in a manner that is similar to the most elementary functions of neurons[27].

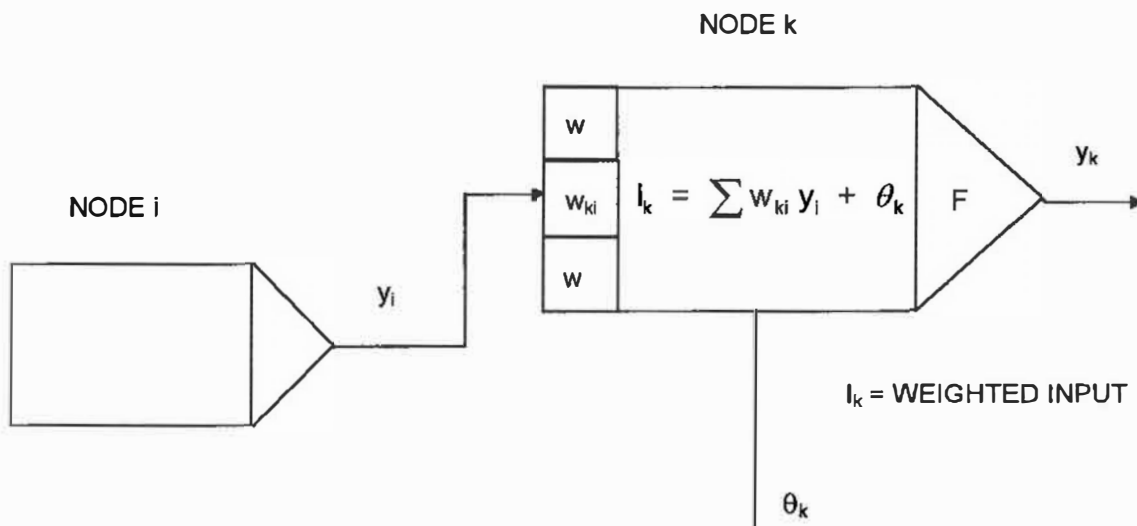
The point of view held by *Krose and Van Der Smagt*[28] is that there is still so little known (even at the lowest cell level) about biological systems, that the models that are being used for artificial neural systems seem to introduce an oversimplification of the 'biological models'.

## 5.2 What is a Neural Network? *Hecht-Nielsen*[10]

A neural network is a parallel, distributed information structure consisting of processing elements. These processing elements can possess a local memory and can carry out localised information processing operations. The processing elements are interconnected via unidirectional signal channels called connections that branch (“fan out”) into as many collateral connections as desired; each carrying the same processing element output signal. This signal can be of any mathematical type desired. The information processing that goes on within each processing element can be defined arbitrarily with the restriction that it must be completely local. This implies it must depend only on the current values of the input signals arriving at the processing element via impinging connections and on values stored in the processing element local memory[28].

According to *Krose and Van Der Smagt*[28], a neural network can be described by the following mathematical descriptors :

1. A state of activation,  $y_k$ , associated with each node (processing element)  $k$  (output of a unit).
2. A real-valued weight  $w_{ki}$ , associated with each connection  $(i,k)$  between two nodes  $i$  and  $k$ . This value determines the effect of signal  $k$  on unit  $i$ .
3. A real-valued bias  $\theta_k$ , associated with each node  $k$ .
4. A transfer function,  $F_k [y_i, w_{ki}, \theta_k, k]$ , defined for each node.  $F_k$  is a mathematical formula that determines the state of the node as a function of :
  - the output states of the nodes feeding it and connected to it by incoming connections.
  - the weights associated with these connections and
  - the bias associated with each node



**Figure 5. 1:** Basic components of an artificial neural network. Reproduced from [28].

### 5.2.1 Processing Elements [28]

The task of each unit (processing element) is to receive input from neighbours or external sources and use this to compute an output signal. This signal is then propagated to other units. Apart from this processing, a second task is adjusting of the weights.

There are 3 types of units :

1. Input Units - receive data from outside the neural network
2. Output Units - send data out of the neural network
3. Hidden Units - whose input and outputs remain inside the network, and function within the system.

Units can be updated either synchronously or asynchronously. Synchronous updating occurs when all units update their activation simultaneously. With asynchronous updating, each unit usually has a fixed probability of updating its activation at a time  $t$  and usually only one unit will do this at a time  $t$ .

It is assumed that each unit provides an additive contribution to the input of the unit with which it is connected. The total input to unit  $k$  is simply the weighted sum of the separate outputs from each of the connected units plus a bias or offset term  $\theta_k$ :

$$s_k(t) = \sum_j w_{kj}(t)y_j(t) + \theta_k(t) \quad 5-1$$

Positive  $w_{kj}$  is considered as an excitation and negative  $w_{kj}$  as an inhibition. The function  $F_k$ , which takes the total input  $s_k(t)$  and the current activation  $y_k(t)$  and produces a new value of the activation of the unit  $k$ , is normally a non-decreasing function of the total input of the unit:

$$y_k(t+1) = F_k(s_k(t)) = F_k\left(\sum_j w_{kj}(t)y_j(t) + \theta_k(t)\right) \quad 5-2$$

There are many different types of activation functions. Some of them are[28]:

- hard limiting threshold functions (sgn function)
- linear or a semi-linear function,
- smoothly limiting function. The smoothly limiting function or sigmoid will be widely used in this study. The other functions are included for completeness.

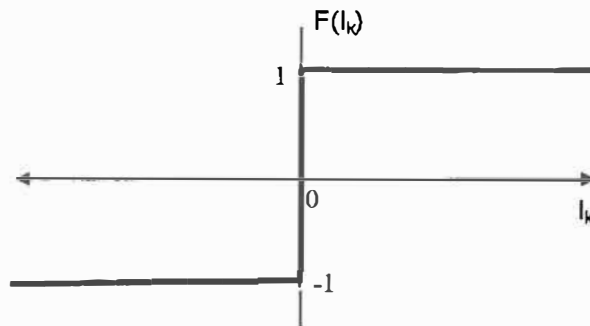
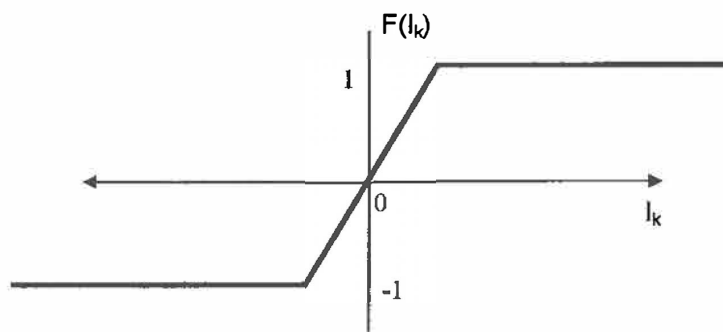


Figure 5. 2: a) Threshold function



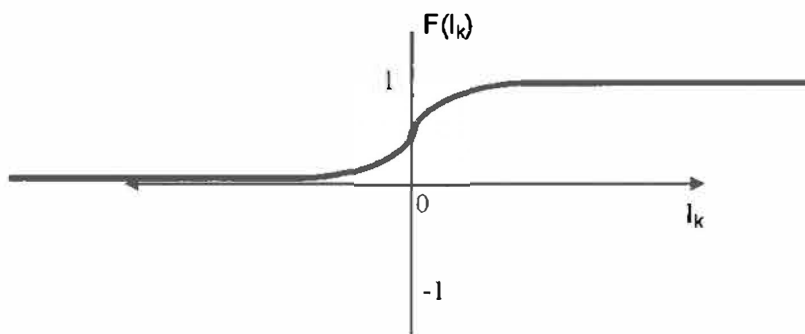


**Figure 5. 3:** b) Hard limiter

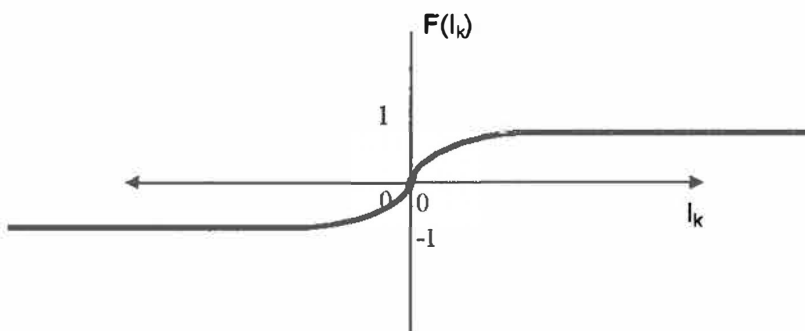
The sigmoid can be defined by

$$y_k = F(I_k) = \frac{1}{1 + e^{-I_k}}$$

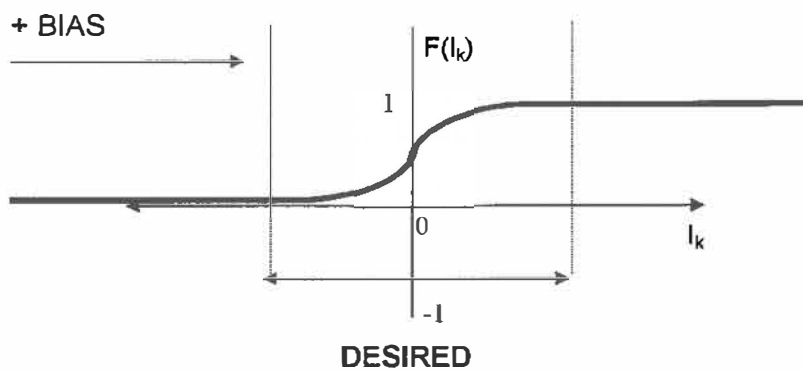
5- 3



**Figure 5. 4:** c) Sigmoidal function



**Figure 5. 5:** d) Hyperbolic tangent function[28]



**Figure 5. 6:** Effect of Bias on an input unit

The bias is specified as a processing element with a constant output that is input to a node that it is influencing. This constant value when added or subtracted shifts the input so that the transfer function is in an appropriate range.

### 5.2.2 Network Topologies

The main distinctions in the pattern of connections of the processing elements, as well as the flow of information is defined by *Krose and Van Der Smagt*[28] as follows:

- *Feed-forward* networks: The data flow from input to output units is strictly in a feed-forward manner. No feedback connections are present (connections extending from outputs of units to inputs of units in the same layer or previous layers) even though the data processing can extend over layers of units.
- *Recurrent* networks: These networks do contain feedback connections.

Only feed-forward artificial neural networks will be considered due to their ability to cope with very high dimensional data, thus making them excellent candidates to perform face recognition from high dimensional space[16].

### 5.2.3 Weights

The weights of the neural network are largely responsible for its ability to predict or classify, and this is perhaps one of the most enigmatic aspects of a neural network. The initial values chosen for these weights are not critical. The main difference between various types of neural networks is the configuration procedure of the weights (training).

Only networks that adjust their weights, rather than their connectivity were investigated. The mean squared error  $F(\mathbf{w})$  is a function of the weight vector  $\mathbf{w}$  of the neural network being evaluated. *Hecht-Nielsen*[10] defines this as the error surface of the neural network. A different mean squared error arises for each  $\mathbf{w}$ .  $F(\mathbf{w})$  can be thought of as a surface sitting “above” the weight space of the network, where  $F$  is the height of the surface at weight value  $\mathbf{w}$ [10].

Because  $F$  is a non-negative function, the error-surface always lies at a non-negative altitude above the weight space.

Figure 5.7 illustrates a typical cross-section of an error surface:

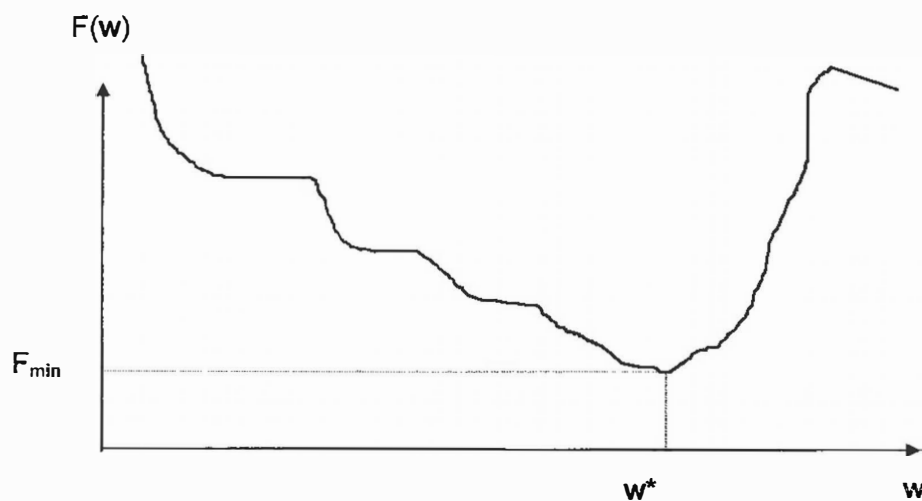


Figure 5. 7: Error surface showing a global minimum[10].

The idea is to find those weights that minimise  $F$ . As the neural network is unable to exactly implement the desired mapping, the minimum value of  $F$  will not be zero, i.e. typically  $F_{\min} > 0$ . The aim of all training algorithms is to find a weight vector  $\mathbf{w}^*$  for which  $F(\mathbf{w}^*) = F_{\min}$ . If this is found, the neural network would do the best possible job of approximating the mapping (as measured using the mean squared error)[10].

### 5.3 Training of Artificial Neural Networks

The training of neural networks can largely be described as the configuration of the weights of the network such that the application of a set of inputs produces the desired set of outputs. Feeding the network teaching patterns and letting it change its weights according to some learning rule is one of the most common forms of training. Learning can be of two different types[28]:

- *Supervised learning* or *Associative learning* in which the network is trained by providing it with input and matching output patterns.
- *Unsupervised* or *Self-organisation* in which an (output) unit is trained to respond to clusters of patterns within the input. The system is supposed to discover statistically the salient features of the input population and there are no *a priori* categories into which the patterns are to be classified.

### 5.4 The Back-propagation Network

The BPNN is a powerful mapping network that has been used in fields as diverse as credit application scoring to image compression[10].

The network consists of 3 basic layers (input, hidden and output). Each layer consists of processing elements. The fanout processing elements ( $N_i$ ) of the input layer simply accept individual components of the input vector and distribute them without modification to all the processing elements in the 2nd layer or hidden layer. The activation of a hidden unit is a function  $F_i$  of the weighted inputs plus some fixed bias.

The output of the hidden units is then distributed to the next layer ( $N_{h,2}$ ) of hidden units, until the last layer of hidden units, of which the outputs are fed into a layer of  $N_o$  output units.

During the first phase the input  $\mathbf{x}$  is presented and propagated forward through the network to compute the output values  $y_o^p$  for each output unit. This output is compared with its desired value  $d_o$ , resulting in an error signal  $\delta_o^p$  for each output unit. The second phase involves a backward pass through the network during which the error signal is passed to each unit in the network and appropriate weight changes are calculated.

Continuous, non-linear functions  $F$  are used in back-propagation networks. Tradition has led the sigmoid function to be the favourite choice but any monotonic, bounded, differentiable function is acceptable. The popular back-propagation training algorithm can be found in several literatures [10],[28], and is thus not given.

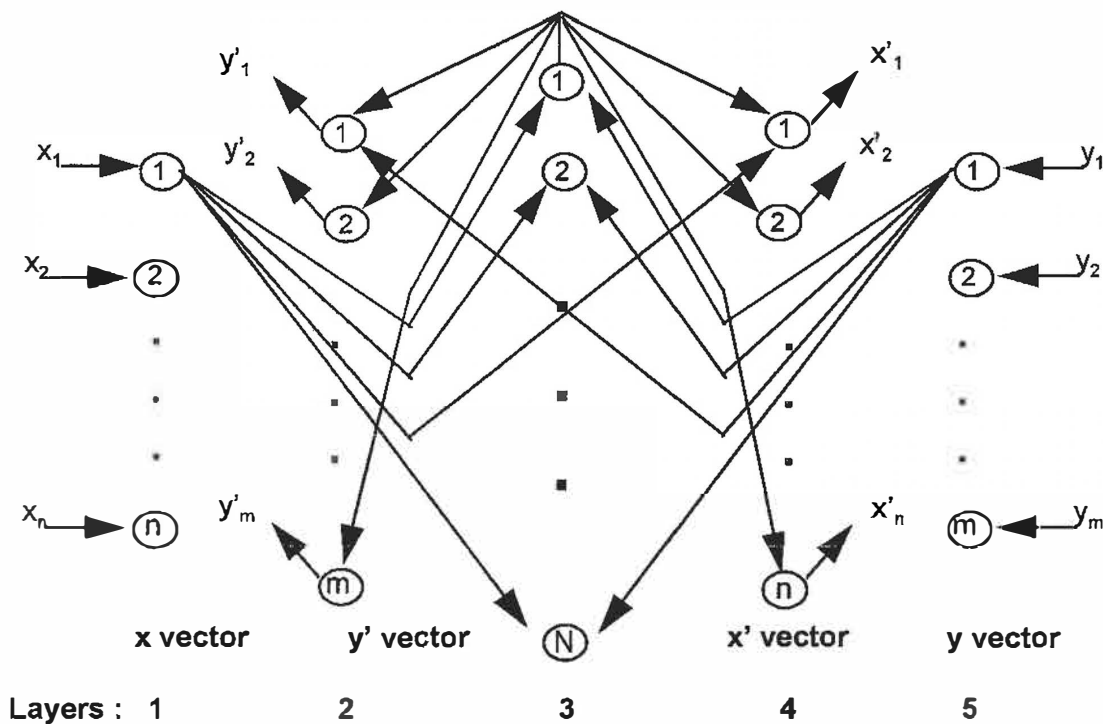
Three basic facts are known about back-propagation error surfaces[10]:

- These functions typically have large numbers of global minima (which may lie at infinity for some problems) as a result of combinatoric permutations of the weights that leave the network input-output function unchanged. As a result, the error surfaces become highly degenerate and have numerous “troughs”. Extensive flat areas and troughs that have very little slope are thus characteristics of the error surface.
- The error surfaces have areas with shallow slopes in multiple dimensions simultaneously. When the combination of certain weights causes the processing elements of one or more hidden layers to be large in magnitude, the output of the processing elements become insensitive to weight changes. The effect is that the weighted sum value moves back and forth along one of the shallow tails of the sigmoid function.
- Local minima exist.

Hence, a momentum factor is introduced which tends to keep the weight changes moving in the same direction. This allows the algorithm to skip over small local minima. It can also improve the speed of learning. But as in the case of learning rate, a large momentum factor may cause the side effect of skipping too much. In past research, momentum factors typically have been set between 0 and 1[12].

## 5.5 The Counter-propagation Network

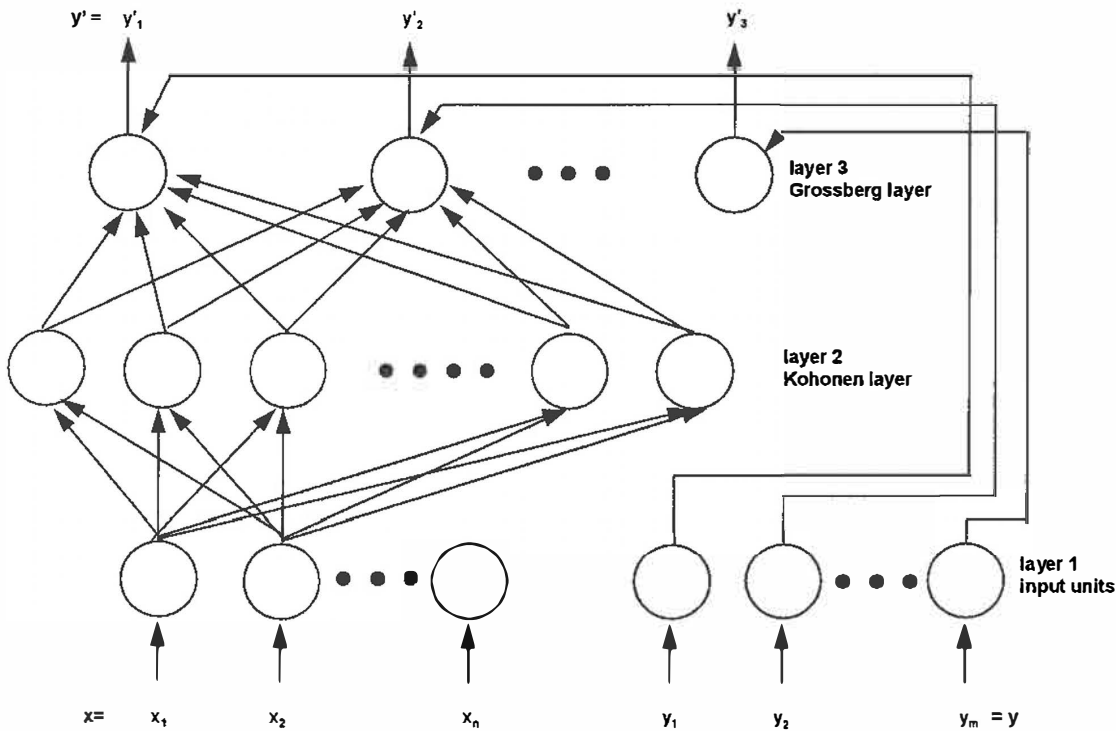
The counter-propagation network was invented by *Hecht-Nielsen*[10] in 1986 while investigating a technique to use self-organising maps to learn explicit functions. The architecture is modelled by combining the self-organising map of *Kohonen*[13] and the outstar structure of *Grossberg*[11].



**Figure 5. 8:** Topology of the counter-propagation network[10]

During learning, pairs of examples  $\mathbf{x}$ ,  $\mathbf{y}$  are presented to the network at layers 1 and 5 respectively. These vectors propagate in a counterflow manner to yield  $\mathbf{x}'$  and  $\mathbf{y}'$

(approximations of  $\mathbf{x}$  and  $\mathbf{y}$ ), hence the name counter-propagation. However, during this study only the feed-forward variant of the typical counter-propagation network was considered. The following figure is the model of the forward only counter-propagation network.



**Figure 5. 9:** Topology of forward-only counter-propagation network [10]

Notice that the architecture of the feed-forward variant has 3 layers, as compared to Figure 5.8, which has 5 layers. The 3 layers consist of

- layer 1 - input layer for distributing signals  $x_1, x_2, \dots, x_N$
- layer 2 - Kohonen layer with  $N$  processing elements that have output signals  $z_1, z_2, \dots, z_N$
- layer 3 - Grossberg outstar layer with  $m$  processing element output signals  $y'_1, y'_2, \dots, y'_M$

The primed outputs represent approximations to the function  $y = \varphi(\mathbf{x})$ . During training the network is exposed to examples from the mapping function  $\varphi$ , i.e. both  $\mathbf{x}$  and  $\mathbf{y}$ .

These networks are trained via supervised training as opposed to graded or self-organisation[11].

**The following explanation as well as interpretation of the workings of the network are from [10] and [11].**

During training, the transfer function equations for the Kohonen layer are

$$z_i = \begin{cases} 1 & \text{if } i \text{ is the smallest integer for which} \\ & \|\mathbf{w}_i^{old} - \mathbf{x}\| \leq \|\mathbf{w}_j^{old} - \mathbf{x}\| \text{ for all } j \\ 0 & \text{otherwise} \end{cases} \quad 5-4$$

Equation 5-4 shows that each element of the input vector is transmitted to each processing element of the Kohonen layer[11]. Each Kohonen unit then calculates its input intensity  $I_i$ , where

$$I_i = D(\mathbf{w}_i, \mathbf{x}) \quad 5-5$$

where

$$\mathbf{w}_i = (w_{i1}, w_{i2}, w_{i3}, \dots, w_{iN})^T$$

$$\mathbf{x} = (x_1, x_2, x_3, \dots, x_N)^T$$

and  $D(\mathbf{u}, \mathbf{v})$  is treated as the Euclidean distance

$$D(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\| \quad 5-6$$

After the intensity of each Kohonen unit is calculated, a competition is held based on which unit has the smallest input intensity. There are various techniques that exist to hold the competition[10] :

1. A scheduling unit of the Kohonen layer can collect all inputs from the Kohonen units. After sorting these, it can broadcast the number of the winning unit.
2. Use lateral inhibition (on centre/off surround type connections); thus only the Kohonen unit which wins is activated.



3. Make the scheduling unit transmit a threshold value  $\Gamma$ , which starts at 0. The first unit to have a distance  $< \Gamma$  wins the competition.

The unit that wins the competition has its output ( $z_i$ ) set to 1 while the output of all other units is set to zero. After the competition, the weights are modified according to the Kohonen learning law as follows [11] :

$$\mathbf{w}_i^{new} = \mathbf{w}_i^{old} + \alpha(\mathbf{x} - \mathbf{w}_i^{old})z_i \quad 5-7$$

As is evident from Equation 5-7, only the winning element's weights are updated, while all others remain the same. Hence, this learning law can be rewritten as:

$$\mathbf{w}_i^{new} = (1 - \alpha)\mathbf{w}_i^{old} + \alpha\mathbf{x} \quad 5-8$$

for the winning processing element, and as

$$\mathbf{w}_i^{new} = \mathbf{w}_i^{old} \quad 5-9$$

for the losing elements. Hence the losing elements do not adjust their weights.

Thus, the Kohonen learning law causes the weight vector to move a fraction  $\alpha$  along the length of the straight line to the vector  $\mathbf{x}$ . As new  $\mathbf{x}$  vectors are input to the network, the weight vectors are drawn to them and form a "cloud" where the  $\mathbf{x}$  vectors actually appear[11].

It has been suggested by [10] and [11] that  $\alpha$  should be set to a high value (e.g. 0.8) and as the  $\mathbf{w}_i$  vectors move into the area of the data,  $\alpha$  is lowered to 0.1 or less for final equilibrium. The aim of the training is that the weights vectors distribute themselves in

an equiprobable configuration and after training has ended the weight vectors are frozen and only Equation 5-4 is used.

Once the Kohonen layer has stabilised (i.e. the  $\mathbf{w}_i$  vectors are frozen), learning takes place in layer 3.

Layer 3 of the network receives its input signals ( $\mathbf{z}$  vector, of which a single element is 1 while the rest are all zero) from the outputs of layer 2. Since this architecture is fully connected, each processing element of layer 3 receives  $\mathbf{z}$ . The learning that takes place in this layer is governed by the following equations[10]:

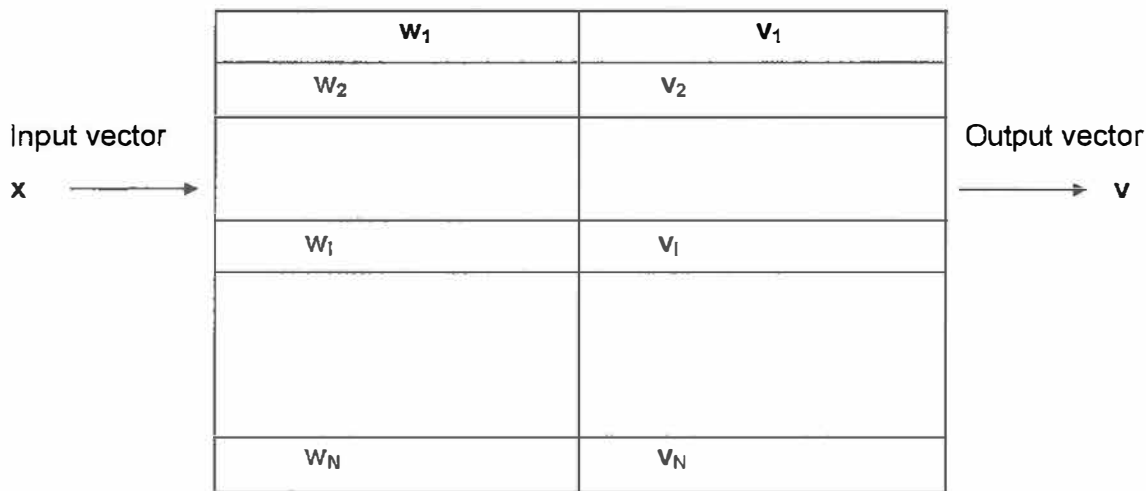
$$\begin{aligned} y_j^t &= \sum_{i=1}^N u_{ji}^{old} z_i \\ u_{ji}^{new} &= u_{ji}^{old} + \alpha(-u_{ji}^{old} + y_j)z_i \end{aligned} \quad 5-10$$

where

$\mathbf{u}_j = (u_{j1}, u_{j2}, \dots, u_{jN})$  is the weight vector associated with the  $j^{\text{th}}$  processing element of layer 3 and  $\alpha$  is the learning rate ( $0 < \alpha < 1$ ) of the Grossberg learning law. It is quite clear from Equation 5-10 that the output of the  $j^{\text{th}}$  processing element of the Grossberg layer is equal to the sum of each Kohonen unit's output signal  $z_i$  times its input weight. The Grossberg equation modifies only the weight associated with input from the winning processing element of layer 2. Over time this weight is modified to learn the average of the correct  $y_j$  values associated with the  $\mathbf{x}$  vector values that cause processing element  $i$  of layer 2 to win the final competition. As layer 2 begins to stabilise, the layer 3 weights begin to learn the averages of the  $y_j$  associated with the  $\mathbf{x}$  vector within the equiprobable win regions of the processing elements of layer 2[10].

When the network is sufficiently trained, it outputs a vector  $\mathbf{v}_i = (v_{j1}, v_{j2}, \dots, v_{mi})$  whenever processing element  $i$  wins the final layer 2 competition. This vector is very close to the vector average of the correct  $\mathbf{y}$  vectors associated with  $\mathbf{x}$  input vectors that cause processing element  $i$  of layer 2 to win[11].

If the constant  $\alpha$  is small enough, the  $\mathbf{v}_i$  vectors also equilibrate. This is the point at which training is generally stopped and the weights of the network are frozen. The behaviour of the network after training is like that of a lookup table.



**Figure 5. 10:** Model of counter-propagation network functioning as an adaptive lookup table[10].

With reference to Figure 5.10, the input vectors  $\mathbf{x}$  is compared to the layer 2 weight vectors  $\mathbf{w}$  using the distance metric  $D(\mathbf{u}, \mathbf{v})$ . The vector  $\mathbf{v}_i$  associated with the  $\mathbf{w}_i$  is output by the network.

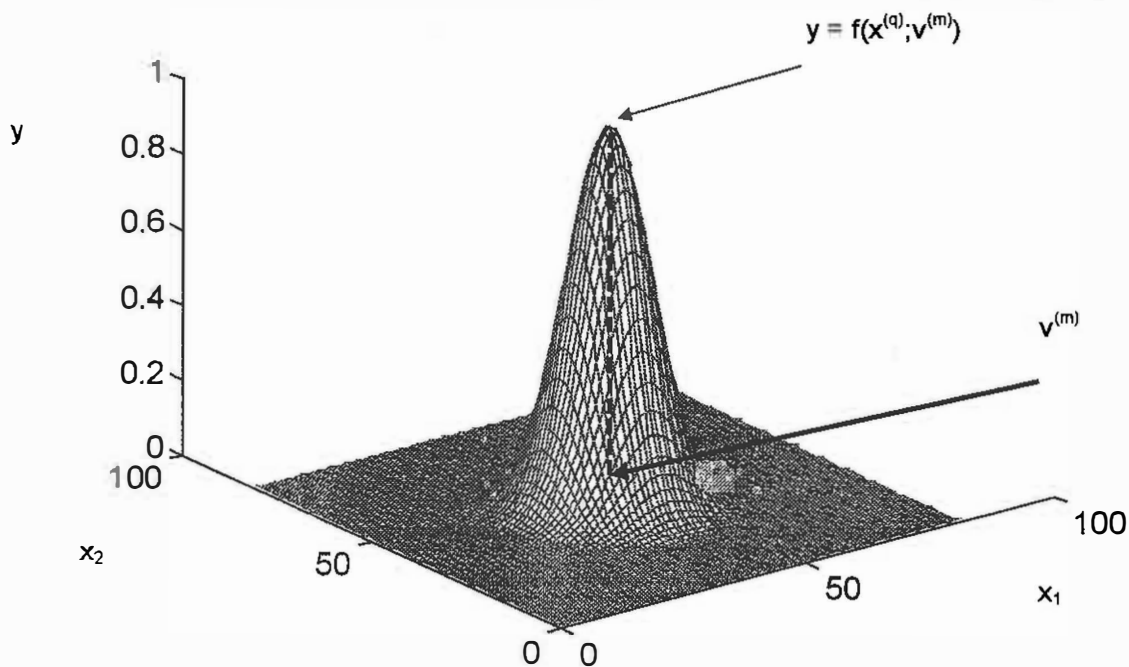
According to Hecht-Nielsen[10], the counter-propagation network is optimal in two ways

- the  $\mathbf{w}_i$  vectors are equiprobable
- the  $\mathbf{v}_i$  output vectors are statistical averages of the  $\mathbf{y}$  vectors associated with the  $\mathbf{x}$  vectors that activate the associated layer 2 processing elements - thus taking on values that are, on average best representative of the function's value in each case.

## 5.6 The Radial Basis Function Network

Radial Basis Function Networks (RBFN) are an extremely powerful type of feed-forward network that is becoming popular because of their quick training, conceptual clarity and elegance. The principle of radial basis functions (RBFs) is derived from the theory of functional approximation. The main difference between the MLP and the RBFN are the activation functions and propagation rules that are used in the hidden layer. RBFs use localised radial Gaussians as an activation function instead of the common sigmoids that are used in MLPs [12].

The RBFN network can best be understood as follows: The feature space is covered with  $M$  overlapping circular hyperball regions. For each region, there are continuous radial basis functions that assumes its maximum value at the centre of the region. Values near zero are outside the region. Each of the  $M$  regions has a centre vector  $\mathbf{v}^{(m)}$  that represents a neuron. Such a region is represented in 2-D space as follows[21]:



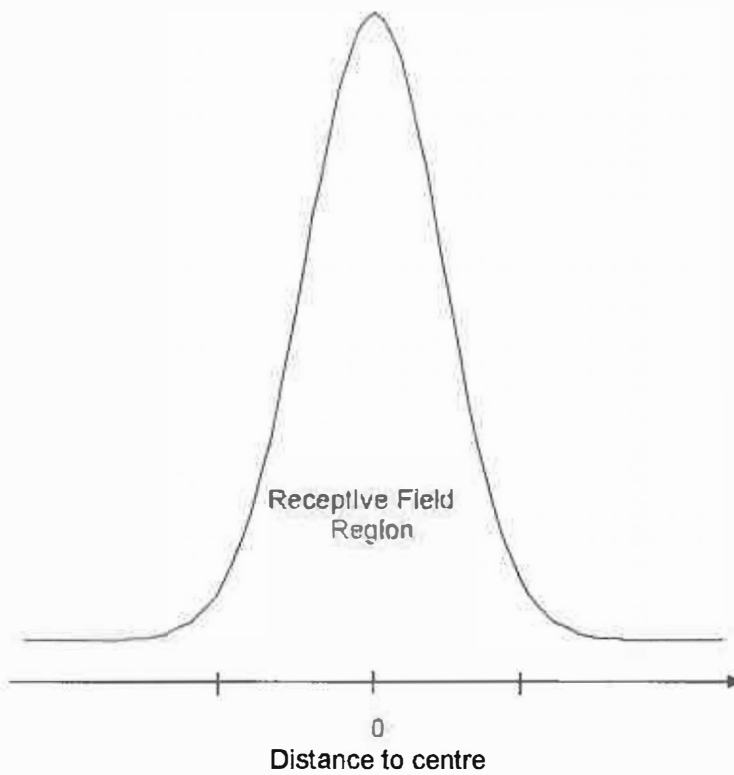
**Figure 5. 11:** A cluster centre and it's RBF[21].

The function shown in Figure 5.11 is a radial basis function, and one of the characteristics is that the  $\mathbf{x}$  vectors are equidistant to the centre  $\mathbf{v}^{(m)}$  and have equal functional values.

The RBF has the form[21]:

$$f_m(\mathbf{x}^{(q)}, \mathbf{v}^{(m)}) = e^{\left[ \frac{-\|\mathbf{x}^{(q)} - \mathbf{v}^{(m)}\|^2}{2\sigma_m^2} \right]} \quad 5-11$$

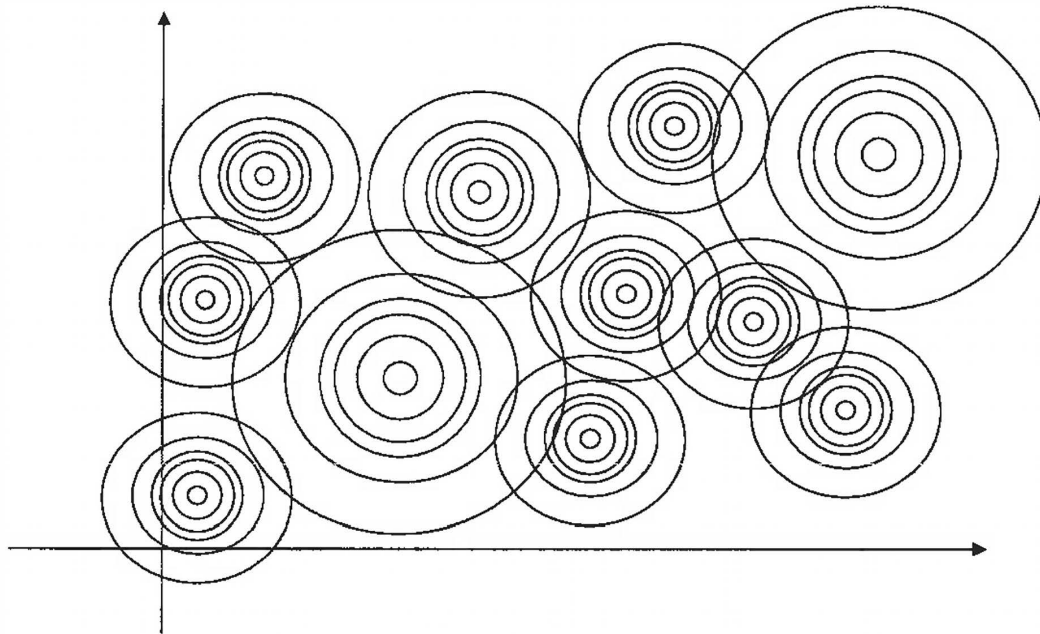
A slice through Figure 5.11 for which the horizontal axis is  $\|\mathbf{x} - \mathbf{v}^{(m)}\|$ , reveals[21]:



**Figure 5. 12:** An RBF Slice [21].

The receptive field region is the region in feature space where  $f_m(\mathbf{x}, \mathbf{v}^{(m)})$  is high. It is clear from Figures 5.11 and 5.12 that each RBF is influential only in its receptive field and this is a small region of feature space. The RBF is activated when an input vector  $\mathbf{x}$

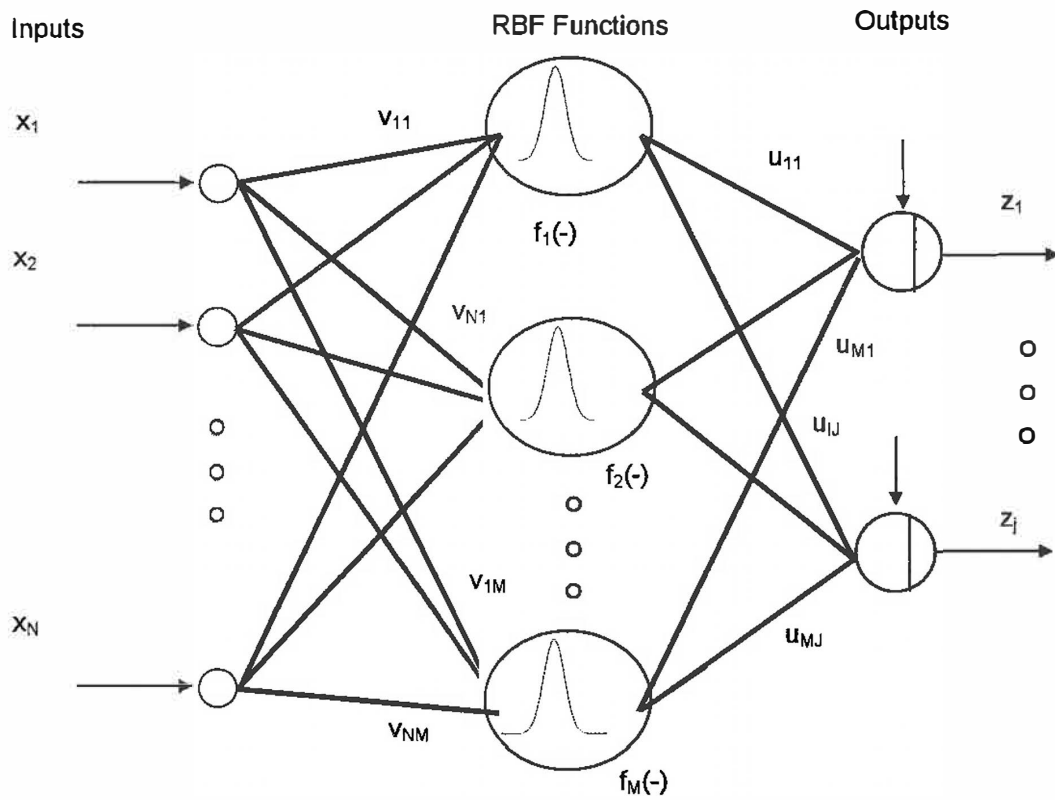
is near  $\mathbf{v}^{(m)}$ . There is no response when  $\mathbf{x}$  is far from  $\mathbf{v}^{(m)}$ . Thus the RBF responds to the small convex region (receptive field) of the feature space. The following figure shows a portion of a two-dimensional feature space covered by RBFs[21].



**Figure 5. 13:** RBF contour curves in the plane[21].

The radial basis function network (see Figure 5.14) contains the following :

- An input layer of neurons, much like an MLP.
- A hidden layer of neurons where each neuron has a special type of activation function centred on the centre vector or sub-cluster in the feature space.
- An output layer.



**Figure 5. 14:** General RBF Network [21].

A bias at each output neuron assures non-zero values of the sums

$$r_j = u_1 y_1 + \dots + u_M y_M + b_j \quad 5-12$$

The neurons represented by the  $M$  centres make up the hidden layer of the  $N$ - $M$ - $J$  feed-forward network. A sigmoid function could be used at the output, however, it is more efficient to use an averaging squashing function[21]:

$$z_j = \frac{1}{S} \sum_{m=1}^M u_{mj} y_m \quad 5-13$$

or

$$z_j = \frac{1}{M} \sum_{m=1}^M u_{mj} y_m \quad 5-14$$

*A brief summary of the operation of the RBFN [21]:*

When an input feature vector  $\mathbf{x}$  is applied to a trained network, it is processed at each hidden neuron to produce an output  $y_m = f_m(\mathbf{x}, \mathbf{u}^{(m)})$ . If the input vector is close to one of the centres  $\mathbf{v}^{(m^*)}$ , the output  $y_m^*$  of the corresponding hidden neuron is greater than any other  $y_m, m \neq m^*$ .

The algorithm that was used for training the RBFN was the dynamic decay adjustment for radial basis functions (RBF-DDA). The RBF-DDA is an algorithm that offers easy training and construction of the RBFN. It has been reported that RBFNs trained with the DDA algorithm attain high classification accuracy and training is significantly faster[42].

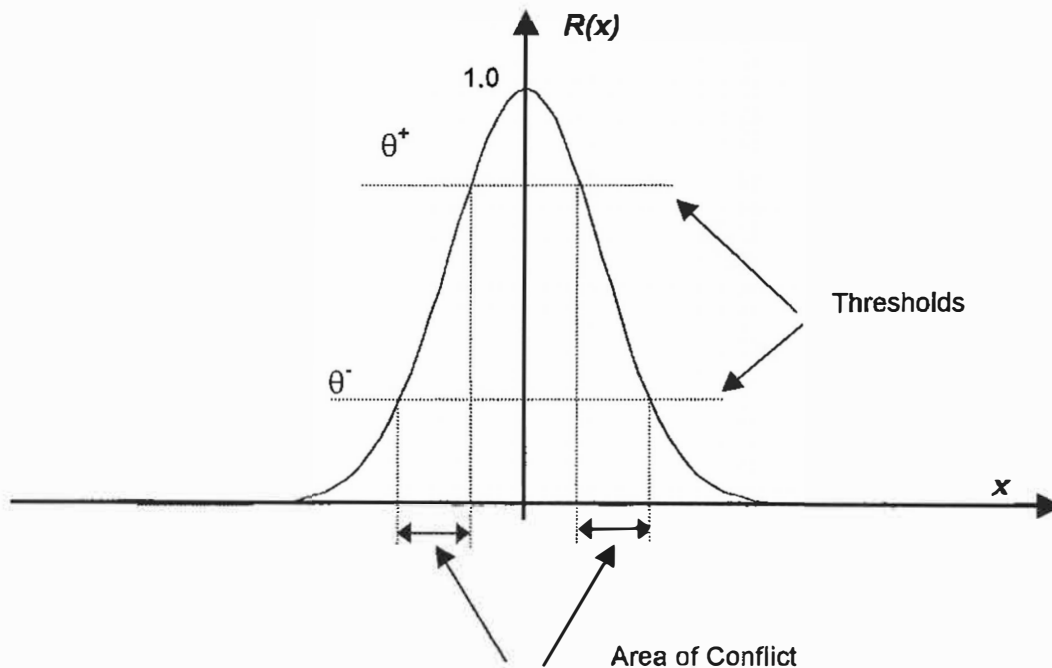
Figure 5.14 is an example of a full RBF-DDA. No shortcut connections exist between input and output units. The RBF-DDA network consists of the following:

- An input layer which represents the dimensionality of the input space
- A hidden layer which contains RBF units. During training, RBF units are added to this layer. Thus, when training begins, only input and output layers of the network exist.
- An output layer where each unit represents one possible class, resulting in a 1-of-n encoding. The processing element with the highest output is the winner and each hidden unit is connected to one output unit.



### 5.6.1 The Dynamic Decay Adjustment Algorithm

The dynamic decay adjustment (DDA) algorithm introduces the concept of matching and conflicting neighbours in an area of conflict.



**Figure 5. 15:** Radial basis function used by the dynamic decay adjustment algorithm[21].

Two thresholds,  $\theta^+$  and  $\theta^-$  are introduced, with  $\theta^+$  generally being greater than  $\theta^-$ , which defines an area of conflict. No prototype of a conflicting class is allowed to exist in this area. That is, neither matching nor conflicting patterns are allowed to lie here. The network uses these thresholds to dynamically construct the network, and adjust the radii individually [12].

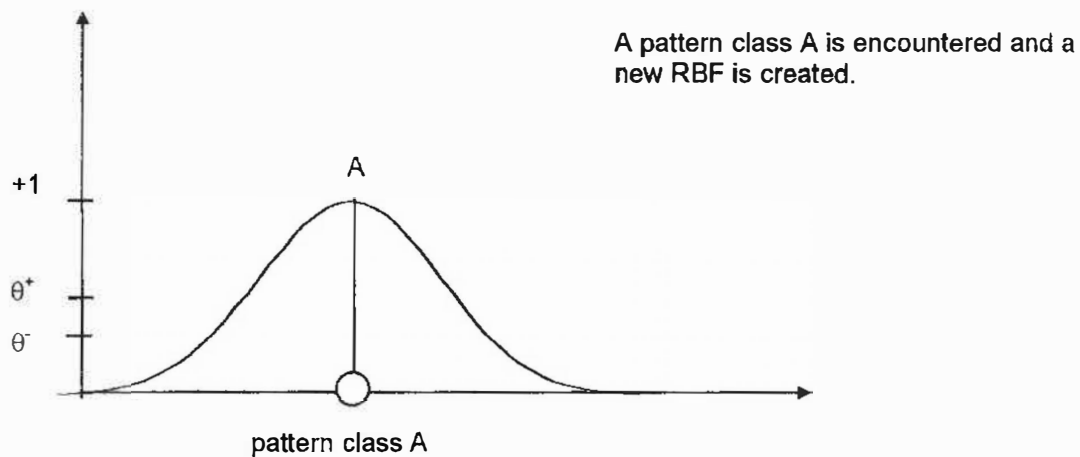
The DDA algorithm that was implemented in SNNS<sup>1</sup> has the following properties [12]:

- **Constructive training:** The network develops from an input layer and an output layer, and the hidden units are added whenever necessary during training.

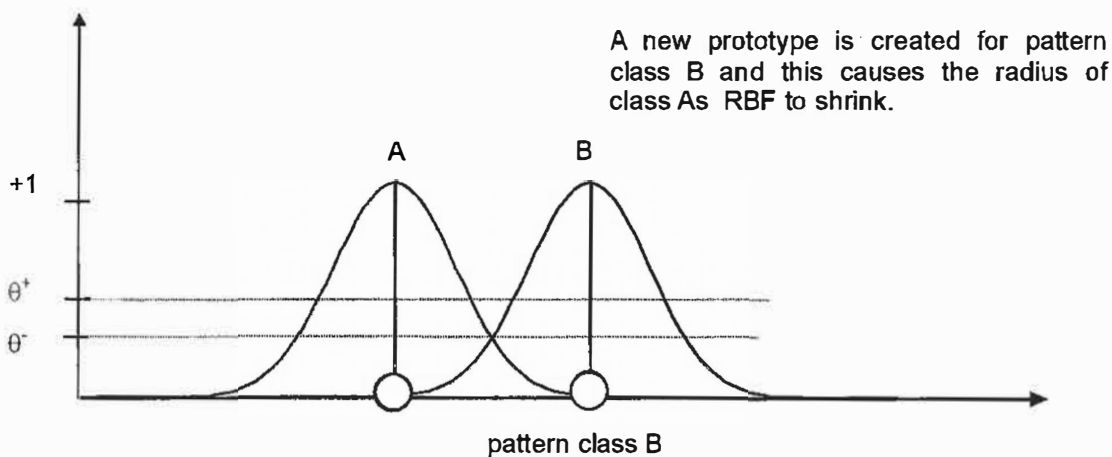
<sup>1</sup> Stuttgart Neural Network Simulator

- **Fast Training:** Few epochs are required for training to reach completion. The authors of SNNS suggest 5 epochs, however, results (see Chapter 6) indicate that 4 epochs are sufficient. The end of training is clearly indicated by either no change in the network topology, or no change in the sum squared error of the network.

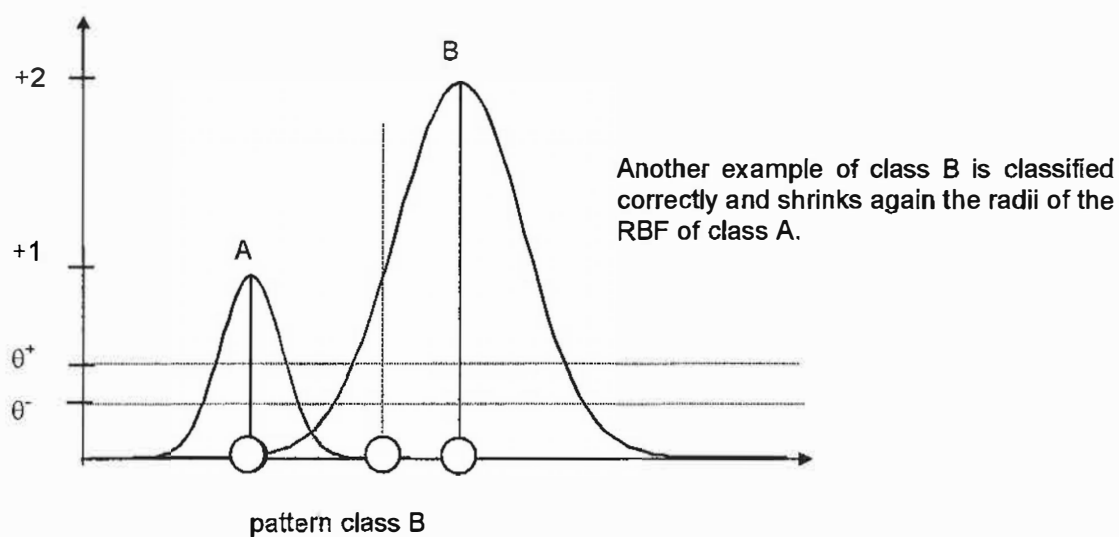
Whenever a pattern is misclassified, either a new RBF unit with an initial weight 1 is introduced or the weight of an existing RBF (which covers the new pattern) is incremented. This process is called commit. The radii of the conflicting RBFs (those RBFs that belong to the incorrect class) are reduced, a process known as shrinking. The advantage of this is that each pattern of the training class has an RBF that covers it which prevents conflicting classes from being classified correctly. The following figures are graphical examples of how the DDA algorithm functions.



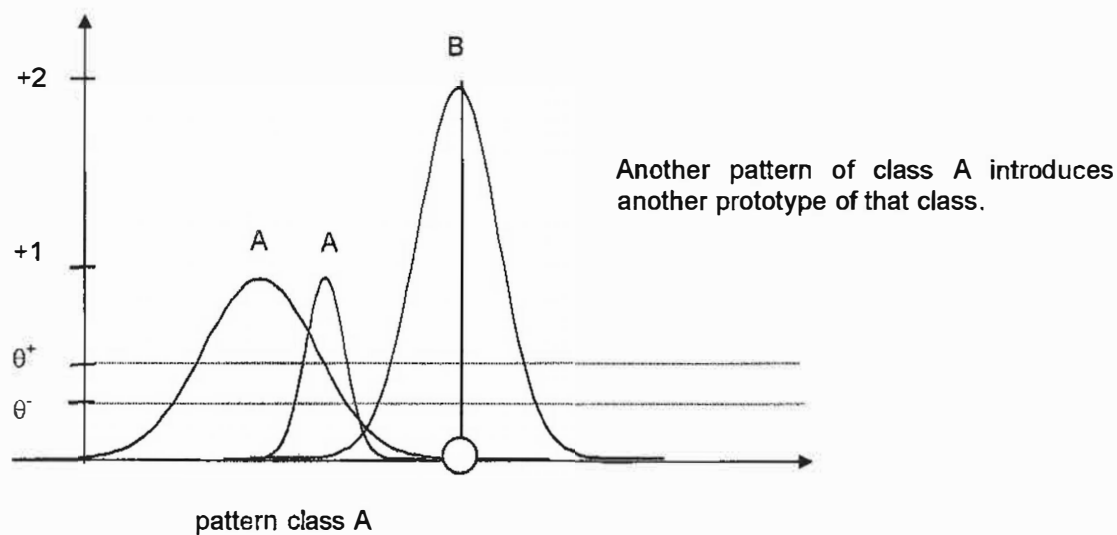
(a)



(b)



(c)



(d)

**Figure 5. 16:** (a) – (d) – Graphical demonstration of DDA algorithm (reproduced from Zell et al[12].)

The results obtained from the investigations pertaining to the neural networks discussed is contained in Chapter 6.

## 5.7 About SNNS

SNNS is the Stuttgart Neural Network Simulator that was developed at the Institute for Parallel and Distributed High Performance Systems at the University of Stuttgart. The goal of the project which began in 1989, is (this is an ongoing project) to create an efficient and flexible simulator environment for research on and application of neural networks.

The software is distributed by the University of Stuttgart as freeware (not public domain) and is available via anonymous ftp from:

```
ftp.informatik.uni.stuttgart.de      (129.69.211.2)
```

in the directory

```
/pub/SNNS
```

as file

```
SNNSV4.1.tar.gz
```

In 1991 the SNNS software was awarded “Deutscher Hochschul – Software Preis 1991” (German Federal Research Software Prize) by the German Federal Minister of Science and Education.

All neural network simulations presented in this study was carried on SNNSv4.1 which was run on a Pentium 100MHz on the Linux (2.0.0) operating system.

# Chapter 6

## RESULTS & DISCUSSION

---

### 6.1 Introduction

This chapter discusses the results obtained from the back-propagation network that was used in a network per person implementation. Results from the counter-propagation network and the radial basis function network (with the DDA algorithm) which were used in a database approach are also presented. The generation of the training and testing sets, as well as the choice of views in the training set is discussed.

### 6.2 Some Definitions

For the purposes of this study, a valid subject is defined as the subject that is to be classified correctly. The valid subject is one

- for which a network has been specially trained for correct classification, or
- who belongs to a database of other valid subjects for whom one network has been trained.

An impostor is defined as one who doesn't belong to the class of valid subjects.

### 6.2.1 Biometric system performance

Two measures are usually used in characterising biometric system performance [29]:

- *FAR* (False Acceptance Rate) - the frequency of fraudulent accesses due to imposters claiming a false identity.
- *FRR* (False Rejection Rate) - the frequency of rejections relative to people that should be correctly verified.

The *FAR* and *FRR* is dependent on some cut-off threshold  $t$  which is used to set the desired security level. If a “tight” threshold setting is used to make it harder for imposters to gain access, some authorised people may find it harder to gain access[29].

These measures can also be defined in terms of negatives and positives:

- *True positives* : Valid subjects that are supposed to be classified as valid subjects.
- *False positives* : Imposters that were classified as valid subjects.
- *True negatives* : Imposters that are supposed to be classified as imposters.
- *False negatives* : Valid subjects that were classified as imposters.

### 6.3 Database approach (DA)

Two different approaches were adopted during this project :

- Database approach (DA): All the subjects are contained in one database, and are assigned a distinct pattern number to facilitate correct identification during the testing phase.
- Network per person approach (NPPA): This approach is unique in that a specific network is trained for each candidate in the database. A portion of the database is used as true negatives during the training process.

The NPPA was implemented as a means of achieving high true positive and high true negative rates. These results will be compared against the database approach.

### 6.3.1 Training and Testing Sets for the Database Approach

The strength of a classifier depends heavily on the data that it receives during training. Ideally, a large number of training examples would map the input space quite well[6]. These examples should span the entire input space. However, at the same time, in a study such as this, one has to be careful not to include all of one's data in the training set. By doing this, it is difficult to determine whether the neural network can actually generalise after learning. It is with this in mind that various types of training data were used as input to the neural network. The performance of the network was monitored and where possible conclusions were drawn about the input data.

It is evident from Chapter 4 that the coefficients with the greatest variance were found in the 10x10 region, thus the input vectors to the neural network were extracted from this region. Data that was normalised in the interval of [-1 1], as well as unnormalised data was used in the training and testing sets. The normalisation was performed as follows :

$$\mathbf{w} = \frac{\mathbf{v}}{\|\mathbf{v}\|} \quad 6-1$$

where  $\mathbf{v}$  is the vector to be normalised, and  $\mathbf{w}$  is the normalised vector.

82 subjects (10 views per subject) were used to generate the training and testing set. Five of the ten views (view numbers 2, 5, 8, 9 and 10 ) of each subject was used in the training set, while the other five (view numbers 1, 3, 4, 6 and 7) were used in the testing set. The views in the training set correspond to the  $\frac{3}{4}$  views mentioned in §2.1. There is no overlap between the training and testing set as far as the views presented to the network is concerned. Different configurations of training and testing set data were used. This is detailed in Tables 6.1 and 6.2. It is necessary to define the way in which the testing and training sets were named. The name consists of 7 fields and each field contains specific information about the file:

Field number

→ 1 2 3 4 5 6 7

7 fields of set name

X X X X X X X ←

**Table 6. 1:** Description of characters used in testing/training set names.

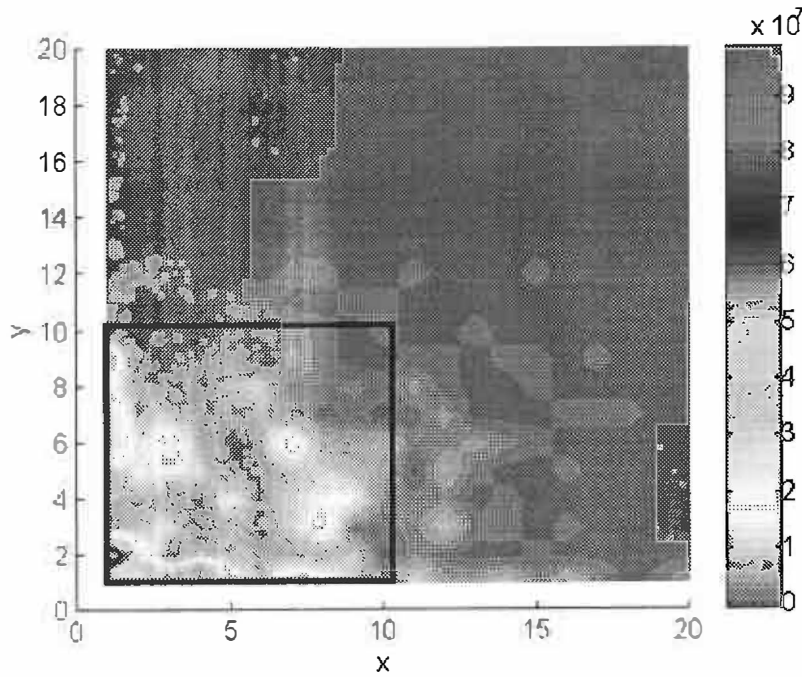
Field Number	Description
1	The database number that the training/testing set belongs to. This number is 1 or 2. Omission of this number means that the complete database was used (i.e. 82 distinct subjects).
2	This field describes whether the training/testing set is using normalised (the field uses a <b>n</b> to show this) or unnormalised (the field uses an <b>u</b> to denote this) data.
3	The region from which the data was extracted is denoted in this field. It contains either <b>8</b> or <b>10</b> , thus denoting an 8x8 or 10x10 square region. An <b>x</b> in this field means that the region is not square.
4	This field describes whether the region that the data was extracted from was square or triangular. A <b>t</b> is used to denote triangular, while a <b>s</b> is used to denote square.
5	A <b>n</b> in this field denotes that no mixed views were used in the training/testing sets, while a <b>m</b> reflects that mixed views were used.
6	This field denotes whether the dc term of the dct was used or not in the training/testing set. If it was used then a <b>t</b> (true) is used in this field, if it wasn't then a <b>f</b> (false) is used.
7	This character defines whether the set is a training or a testing set. An <b>e</b> is used to denote a testing set while a <b>r</b> is used to denote a training set.

(Table 6.2 contains full descriptions of the file names. See p.12 for some examples)

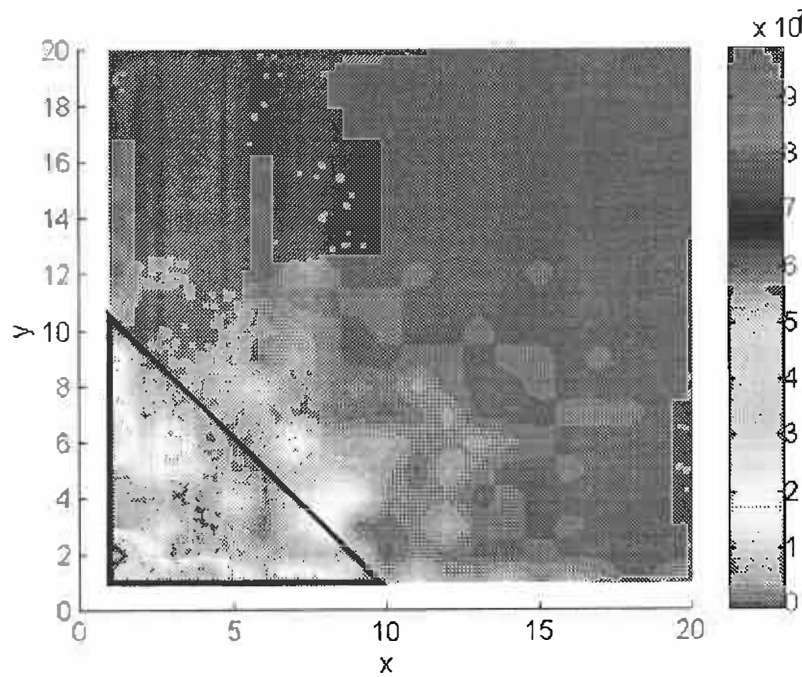
From Table 6.1, it is further necessary to define what square and triangular regions are.

This is best explained by viewing Figures 6.1 and 6.2.





**Figure 6. 1:** Interpolated top view of Figure 4.20. The thick black square denotes the region which constitutes a 10x10 square.



**Figure 6. 2:** Interpolated top view of Figure 4.20. The thick black triangle denotes the region from which the data was extracted. (referred to later as triangular data)

In a set with no mixed views, all subjects in the set contain the same view numbers.

Hence, in a set with no mixed views, all subjects are trained on views 2, 5, 8, 9 and 10

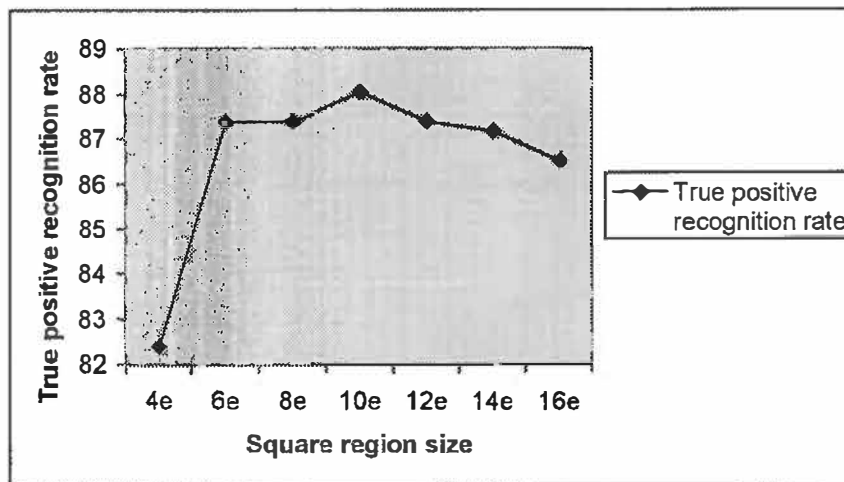
(these are known as the  $\frac{3}{4}$  views) and tested on views 1, 3, 4, 6 and 7 (later called “remaining views”). In a set with mixed views however, subjects in the set contain different view numbers. In a training set with mixed views, half the subjects are trained on views 2, 5, 8, 9 and 10, while the other half are trained on views 1, 3, 4, 6 and 7. In the testing set with mixed views, those subjects that were trained on views 2, 5, 8, 9 and 10, are tested on views 1, 3, 4, 6 and 7, and subjects that were trained on views 1, 3, 4, 6 and 7 are tested on views 2, 5, 8, 9 and 10. This type of training/testing set was generated to determine whether the network would be able to generalise better, if presented with an input space that attempted to span the entire feature space. Table 6.2 is a description of the training/testing sets that were generated to investigate the performance of different input data to the network.

In addition to obtaining results from one database with 82 subjects, results were also obtained by splitting this large database of 820 faces to two smaller databases with 41 unique subjects in each smaller database (an empty intersection exists between the two databases). Hence, as reflected in Table 6.2, there are some training/testing sets that belong to database 1, and some that belong to database 2. Networks were trained for each of these databases and tested. Networks that were trained for subjects that were in database 1, were tested with the testing set of database 1 and 2, as well as the training set of database 2. This afforded an opportunity to investigate the robustness of the networks, as well as their generalisation ability.

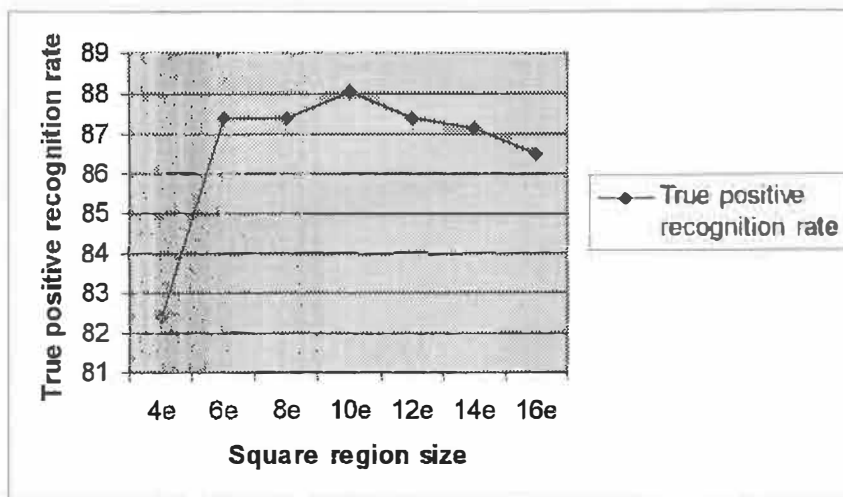
Chapter 4 showed the general location of coefficients with high variance that could be used in pattern recognition applications. However, this was a visual analysis and susceptible to human error. Hence, investigations were performed with different sizes of data sets, extracted from different regions to determine the optimum size and region from which the data would be extracted to be used in further investigations.

Figures 6.3 and 6.4 reflect results obtained from using data extracted from square regions of the transformed images. The x-axis is calibrated from 0 to 16e. These are indicative of the size of the square that was extracted (e.g. 4e 4x4 square; 16e 16x16

square). The e suffix denotes that the network was trained on “remaining views” and tested on  $\frac{3}{4}$  views. The r suffix (see Figure 6.4) denotes that the network was trained on  $\frac{3}{4}$  and tested on remaining views. None of the data sets contained mixed views. Each of the legends of Figures 6.3-6.6 denote an RBF-DDA network that was initialised, and trained for 5 epochs. Each test (i.e. each legend) was performed three times, and the best of three results is reported.

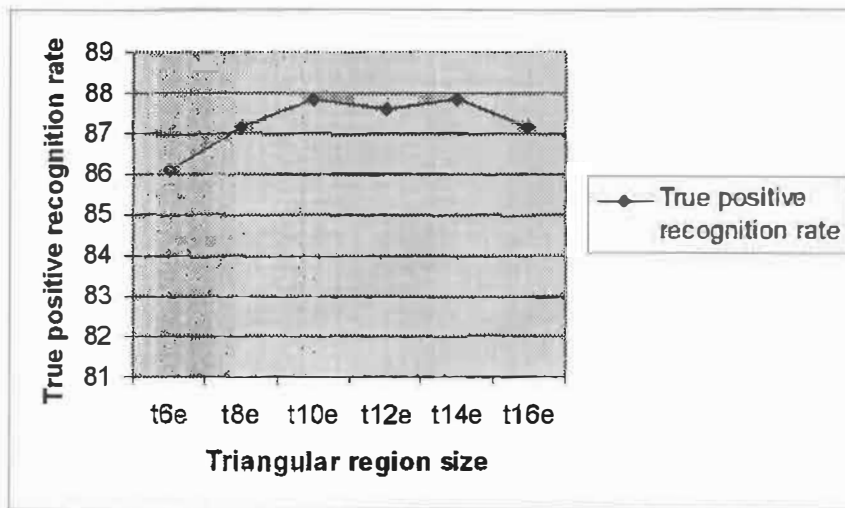


**Figure 6. 3:** True positive recognition rates for various square region data sets (trained on remaining views and tested on  $\frac{3}{4}$  views).

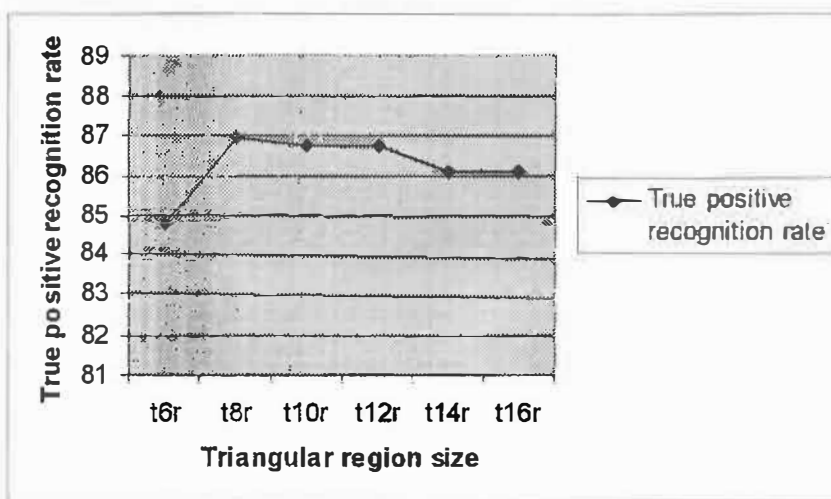


**Figure 6. 4:** True positive recognition rates for various square region data sets (trained on  $\frac{3}{4}$  views and tested on remaining views).

Figures 6.5 and 6.6 contain results for data that was extracted from a triangular region. Like Figures 6.3 and 6.4, the x-axis denotes the size of the two arms of the isosceles triangle (in terms of number of coefficients). No mixed views were used in these data sets.



**Figure 6. 5:** True positive recognition rates for various triangular region data sets.



**Figure 6. 6:** True positive recognition rates for various triangular region data sets

Figure 6.3 shows that high recognition rates are obtained when the length of one side of the square region from which the data is extracted is between 8 to 12 (the recognition rates were 87.4%, 88.0% and 87.4% respectively) coefficients “long”. For data that is

trained on  $\frac{3}{4}$  views, the optimum length of the square is also between 8 to 12 coefficients long (the recognition rates were 87.07%, 87.2% and 85.4% respectively).

Figure 6.5 reveals that for triangular region data, the optimum length for the equal sides of the isosceles triangle is between 10 to 14 (the recognition rates were 87.9%, 87.6% and 87.8% respectively) coefficients long for networks trained on remaining views and tested on  $\frac{3}{4}$  views. The opposite results of this train/test operation are shown in Figure 6.6, and it is suggested that the optimum side of the triangle is between 8 to 12 coefficients (the recognition rates were 87.0%, 86.7% and 86.1%). The choice of the size of the data sets, as well as the region from which they were extracted was based on the true positive recognition rates of Figures 6.3-6.6. For square region data, data of size 10x10, and 8x8 was extracted. The 8x8 was chosen over the 12x12 because of its performance in Figure 6.4, as well as the fact that fewer number of coefficients were required to obtain the same recognition rate. For triangular region data, data from the "10x10" region was used. The data sets used, as well as the views chosen for each data set is given in Table 6.2.

The headings of Table 6.2 represent the following:

Training/Testing set	-	These are files that were used in the train/test procedure.
Database Number	-	As explained on pg. 6-6.
Norm.	-	This indicates that the data was normalised
Unnorm.	-	This indicates that the data was un-normalised.
Square	-	Indication of the shape of area from which the data was taken. (see Figure 6.1)
Triang.	-	Indication of the shape of area from which the data was taken. (see Figure 6.2)
Mixed Views	-	Indicates whether mixed views were used.
No. of distinct patterns	-	These are the unique number of subjects in the database being used for training/testing.

**Table 6. 2:** Description of files used in the training and testing process.

Training/ Testing set	Database number	Norm.	Unnorm.	Square	Triang.	Mixed Views	No. of distinct patterns
U10snfe/r			✓	✓			82
U10smfe/r			✓	✓		✓	82
U8snfe/r			✓	✓			82
U8smfe/r			✓	✓		✓	82
Uxtnfe/r			✓		✓		82
Uxtmfe/r			✓		✓	✓	82
N10snfe/r		✓		✓			82
N10smfe/r		✓		✓		✓	82
N8snfe/r		✓		✓			82
N8smfe/r		✓		✓		✓	82
Nxtnfe/r		✓			✓		82
Nxtmfe/r		✓			✓	✓	82
1n10smfe/r	1	✓		✓		✓	41
1n10snfe/r	1	✓		✓			41
1u10smfe/r	1		✓	✓		✓	41
1u10snfe/r	1		✓	✓			41
1n8smfe/r	1	✓		✓		✓	41
1n8snfe/r	1	✓		✓			41
1u8smfe/r	1		✓	✓		✓	41
1u8snfe/r	1		✓	✓			41
1nxtmfe/r	1	✓			✓	✓	41
1nxtnfe/r	1	✓			✓		41
1uxtmfe/r	1		✓		✓	✓	41

Training/ Testing set	Database number	Norm.	Unnorm.	Square	Triang.	Mixed Views	No. of distinct patterns
1uxtnfe/r	1		✓		✓		41
2n10smfe/r	2	✓		✓		✓	41
2n10snfe/r	2	✓		✓			41
2u10smfe/r	2		✓	✓		✓	41
2u10snfe/r	2		✓	✓			41
2n8smfe/r	2	✓		✓		✓	41
2n8snfe/r	2	✓		✓			41
2u8smfe/r	2		✓	✓		✓	41
2u8snfe/r	2		✓	✓			41
2nxtmfe/r	2	✓			✓	✓	41
2nxtnfe/r	2	✓			✓		41
2uxtmfe/r	2		✓		✓	✓	41
2uxtnfe/r	2		✓		✓		41

Some examples of what the filenames mean are as follows:

- U10snfe is a file that has un-normalised data taken from a square region with no DC term and no mixed views and is used as a testing set.
- 2uxtnfr is a file that is derived from database 2 and contains un-normalised data extracted from a triangular region. The file has no mixed views or DC term and is used as a training set.

The dc term is dependent on the light in the image, and can be used with great effect in recognition problems, as demonstrated by *Wilder*[9]. However, the face recognition model would then be partially dependent on light, hence this term was excluded in all the pattern files.

## 6.4 The Radial Basis Function Network

It is evident from Chapter 5 that for the radial basis network that was implemented with a dynamic decay algorithm, just two parameters need to be optimised:  $\theta^+$  and  $\theta^-$ . A preliminary study was performed to determine these parameters. These results are tabulated in Appendix D from Table D.1 to Table D.7. For the purposes of this chapter, all results will be presented graphically.

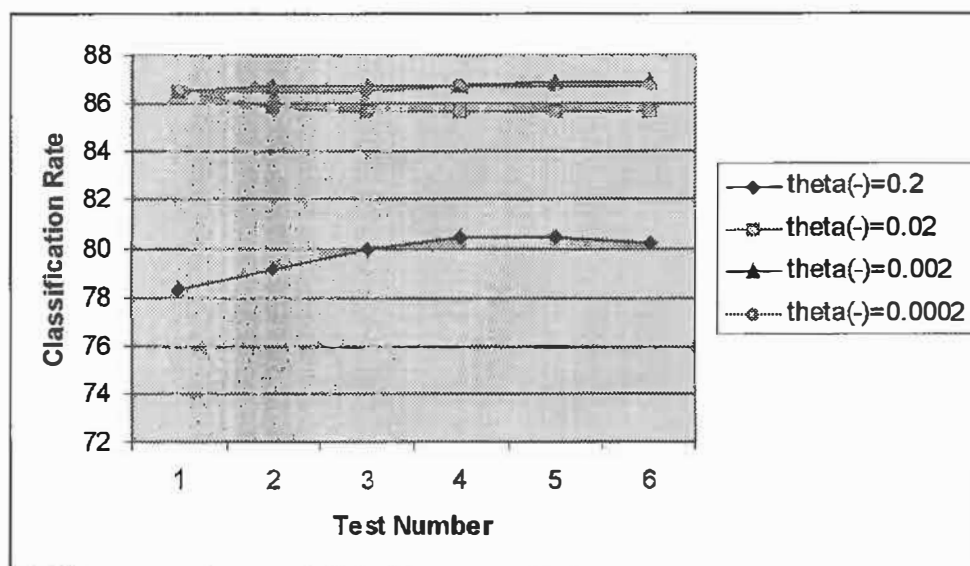
Unnormalised data that was extracted from a 10x10 square region, with no “mixed views” and no dc term was used in the preliminary study. The learning function was set to RBF-DDA in SNNS. No initialisation or update functions are required when using RBF-DDA. Zell *et al*[12] suggest that  $\theta^+$  be set to 0.4 and  $\theta^-$  be set to 0.2. They mention that in theory these parameters should be dependent on the dimensionality of the feature space, but in practice the values are not critical. Each investigation was performed three times, and the best of the three results are presented. It was noted that even though there were minor differences in the sum squared errors (SSE) and mean squared errors (MSE) of the three tests, the classification rate was always the same. Table 6.3 details the investigations that were undertaken.

**Table 6.3:** Table of parameters that were changed during investigations to determine optimum values for  $\theta^+$  and  $\theta^-$ .

Investigation Number.	Description of parameters changed
1 ( Figure 6.7 & Table D.1)	$\theta^-$ is kept constant at 0.2, while $\theta^+$ is varied in increments of 0.1 from 0.4 to 0.9 .
2 ( Figure 6.7 & Table D.2)	$\theta^-$ is kept constant at 0.02, while $\theta^+$ is varied in increments of 0.1 from 0.4 to 0.9 .
3 ( Figure 6.7 & Table D.3)	$\theta^-$ is kept constant at 0.002, while $\theta^+$ is varied in increments of 0.1 from 0.4 to 0.9 .
4 ( Figure 6.7 & Table D.4)	$\theta^-$ is kept constant at 0.0002, while $\theta^+$ is varied in increments of 0.1 from 0.4 to 0.9 .



5 ( Figure 6.8 & Table D.5)	$\theta^+$ is set equal to $\theta^-$ . Both values are initially set to 0.1 and increment in steps of 0.1 to 0.9 .
6 ( Figure 6.8 & Table D.6)	$\theta^+$ is set equal to $\theta^-$ . Both values are initially set to 0.01 and increment in steps of 0.01 to 0.09 .
7 (Table D.7)	$\theta^-$ is kept constant at 0.001, while $\theta^+$ is varied in increments of 0.1 from 0.4 to 0.9 .



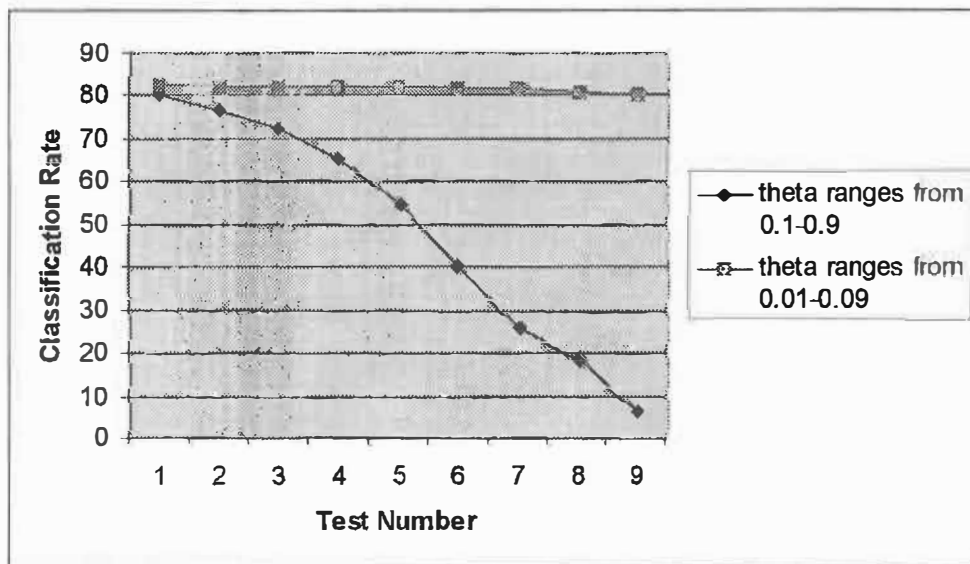
**Figure 6.7:** Graph of classification (true positive) rates for investigations 1 to 4. (see Tables D.1 to D.4) The architecture of the network that was used to carry out these investigations consisted of 99 inputs and 82 outputs (there are 82 distinct subjects in the training and testing sets). The DDA algorithm automatically inserts hidden units (see §5.10 on RBF with DDA in Chapter 5).

From Figure 6.7, it is evident that the true positive rate is greater between tests 3 and 6 when  $\theta^- = 0.2$ . Further tests were carried out to investigate the effect of increasing the ratio of  $\theta^+:\theta^-$ .  $\theta^-$  was kept constant at 0.02 and  $\theta^+$  was varied from 0.4 to 0.9 in increments of 0.1. An improvement in the classification rate is clearly noted. The best classification rate of investigation 1 is 80.4%, while the best classification rate of

investigation 2 is 86.5%. The ratio of  $\theta^+:\theta^-$  was once again increased by lowering  $\theta^-$  to 0.002 and increasing  $\theta^+$  from 0.4 to 0.9 in steps of 0.1.

The results obtained for investigation 3 are an improvement over those of investigation 2. The best classification rate for investigation 2 was 86.5%, while that of investigation 3 is 86.9%. The ratio  $\theta^+:\theta^-$  was increased further by keeping  $\theta^-$  constant at 0.0002 and incrementing  $\theta^+$  from 0.4 to 0.9.

In investigation 4, increase in the ratio of  $\theta^+:\theta^-$  had no significant increase in the classification rate. For completeness, investigations 5 and 6 were carried out by setting  $\theta^+$  equal to  $\theta^-$ . In investigation 5, both parameters are set to 0.1 and incremented in steps of 0.1 to 0.9. The same procedure is used in investigation 6, but the step size is 0.01, and the initial values of  $\theta^+$  and  $\theta^-$  are 0.01 and 0.01 respectively. The results of investigation 5 and 6 are denoted in Figure 6.8 (theta refers to  $\theta^+$  and  $\theta^-$ ).



**Figure 6. 8:** Graph of classification rate for investigations 5 and 6. Theta in the figure refers to  $\theta^+$  and  $\theta^-$ . The architecture of the network that was used to carry out these investigations consisted of 99 inputs and 82 outputs (there are 82 distinct subjects in the training and testing sets). The DDA algorithm automatically inserts hidden units (see §5.10 on RBF with DDA in Chapter 5).

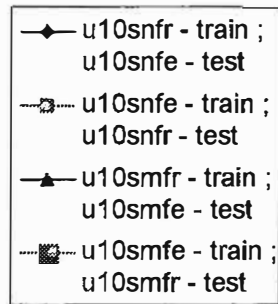
There is a clear decrease in the performance of the network when trained with values ranging from 0.1 to 0.9. The performance of the network when trained with values ranging from 0.01 to 0.09 is not as convincing as the results of investigations 3 and 4. From the preliminary results, it was decided to confine the values of  $\theta^+$  and  $\theta^-$  to those shown in Table 6.4.

**Table 6. 4:** Values of  $\theta^+$  and  $\theta^-$  that were used in the investigations of the various training/test sets.

$\theta^+$	$\theta^-$
0.4	0.01
0.4	0.002
0.4	0.0002
0.8	0.002
0.8	0.0002

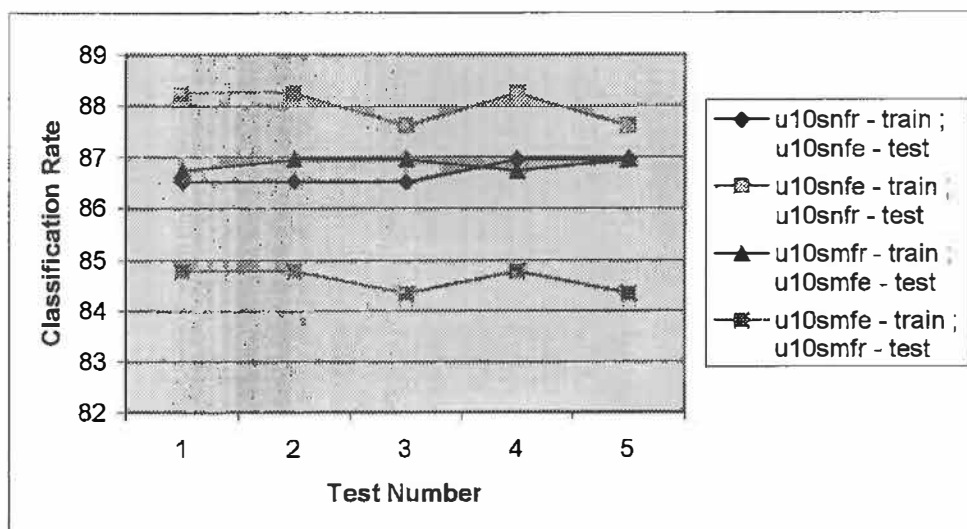
### 6.4.1 Tests with unnormalised data

The first set of results that are presented, are performed with full databases (as opposed to “split databases”), as well as unnormalised data. The size of the data in the training/testing set is also varied, (i.e. 10x10 square, 8x8 square and triangular region data) to determine which data set size yields the best performance. All results of this section are presented in a graphical format, and are essentially a summary of the results contained in the Tables D8-D19. Each test (depicted by a single legend on the graph) was performed three times, and the best of the three results are reported. It is necessary to define the legend that is provided on the right hand side of each figure. This presents information with regard to which files were used for testing, and which were used for training.



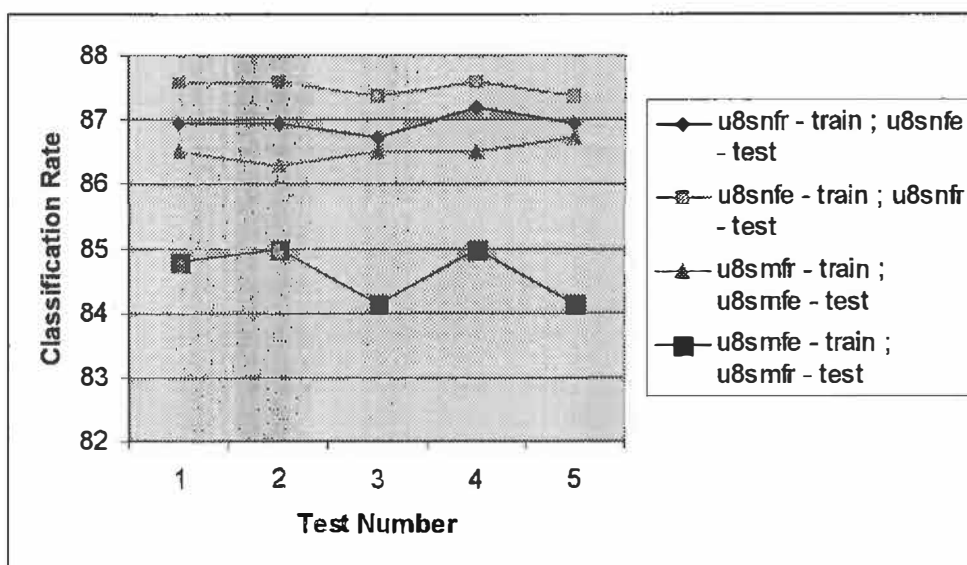
**Figure 6. 9:** Example of legend that is used in figures containing results.

Where possible, results are given in a comparative format. The legends in Figure 6.9 indicate that four separate tests were done with different configurations of data. The first legend indicates that u10snfr (data with no mixed views) was used to train the network, while u10snfe (the testing file counterpart) was used to test the data. An entirely new network was then trained on u10snfe data, and tested on u10snfr data (denoted by the second legend). This exercise was carried out to determine whether  $\frac{3}{4}$  views were better suited to the training set, or the testing set.. An entirely new network was once trained on u10smfr (the mixed views) data and tested on u10smfe data. Once again, the role of the original training and testing sets were swapped, and a new network was trained on u10smfe data, and tested on u10smfr data. This exercise was carried out to determine whether the network would be able to generalise better if it was presented with an input space that attempted to map the entire feature space. Each legend in the following figures depict a specific network that was reset, initialised, trained and tested by the data that is specified by that legend.



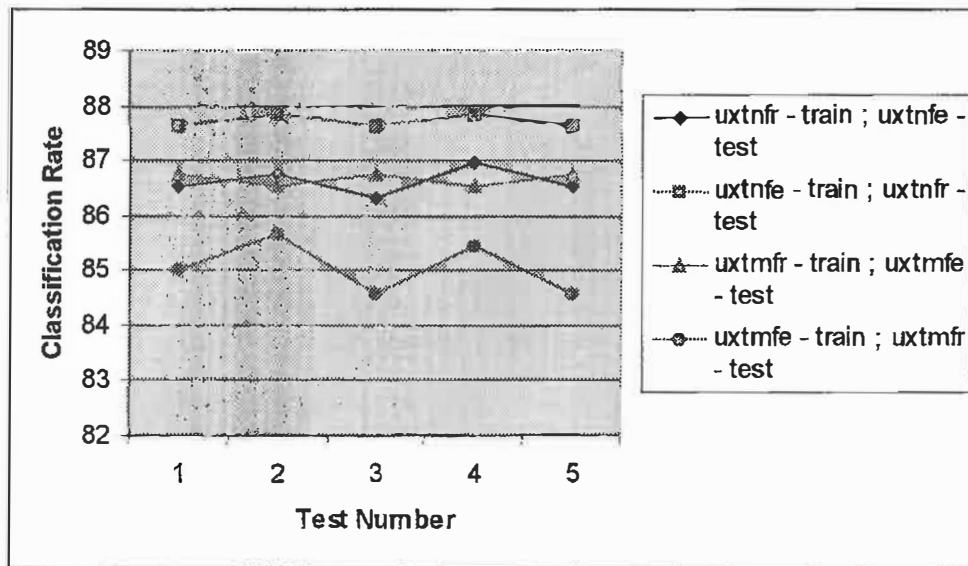
**Figure 6. 10:** Comparative results of tests performed with u10snfr/e and u10smfr/e data (see Tables D.8-D.11).

The architecture of the network that was used to carry out the investigations in Figure 6.10 consisted of 99 inputs and 82 outputs (there are 82 distinct subjects in the training and testing sets). The DDA algorithm automatically inserts hidden units (see §5.10 on RBF with DDA in Chapter 5).



**Figure 6. 11:** Comparative results of tests performed with u8snfr/e and u8smfr/e data (see Tables D.12-D.15).

The architecture of the network that was used to carry out the investigations in Figure 6.11 consisted of 63 inputs and 82 outputs (there are 82 distinct subjects in the training and testing sets). The DDA algorithm automatically inserts hidden units (see §5.10 on RBF with DDA in Chapter 5).



**Figure 6. 12:** Results of tests performed with uxtnfr/e and uxtnmfr/e data (see Tables D.16-D.19).

The architecture of the network that was used to carry out the investigations in Figure 6.12 consisted of 54 inputs and 82 outputs (there are 82 distinct subjects in the training and testing sets). The DDA algorithm automatically inserts hidden units (see §5.10 on RBF with DDA in Chapter 5).

With respect to Figure 6.10, the highest classification rate (true positive rate) obtained was 88.3% when the network was trained on u10snfe data and tested on u10snfr data. The results of Figure 6.10 suggests that tests performed with the mixed views data is not as good as those obtained from non-mixed views data. The highest true positive rate obtained in Figure 6.11 was 87.6% when the network was trained on u8snfe data and tested on u8snfr data. Once again, results obtained from tests performed with the non-mixed views data were better in comparison to mixed views data. In Figure 6.12, the highest classification rate obtained was 87.8% when the network was trained on uxtnfe

data and tested on uxtnfr data. In this test as well, results from tests using non-mixed views data were better than mixed views data.

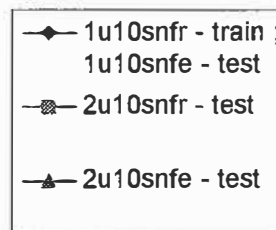
**Some general conclusions that can be drawn for unnormalised data and full databases can be listed as follows:**

- It can be clearly observed that networks that were trained on files that contained “remaining” view information (pattern files that have an “e” suffix, and no mixed views) and tested on the  $\frac{3}{4}$  views (pattern files that have an “r” suffix, and no mixed views) performed significantly better than other networks. The highest true positive recognition rates, in Figures 6.10 - 6.12 were obtained when the networks were trained/tested on pattern files described above. Hence, for full databases, and unnormalised data, it can be generally concluded that it is better to train the network on pattern files that contain remaining views, and test on the  $\frac{3}{4}$  views.
- In all of the tests, non-mixed views data generally performed better than mixed views data. Mixed views didn’t enhance the generalisation ability of the network, instead, general classification rate decreased when mixed views data was used in the pattern files (see Figures 6.10 – 6.12).

The “triangular” data (highest true positive recognition rate was 87.8%) performed marginally better than the 8x8 square data (highest true positive recognition rate was 87.8%). The reason for this is attributed to the fact that the triangular region contained data with higher variance. This leads to clearer pattern boundaries and facilitates easier classification. The best performance however, was obtained from the 10x10 square region data (highest true positive recognition rate was 88.3%). Although this figure may seem high, it is unacceptable in a practical system. Hence, it was decided to split the pattern sets that were used into 2 “databases” with 41 subjects each. There was no intersection between database 1 and 2 with respect to the distinct subjects. Tests with normalised data and full databases was not performed due to the poor results obtained in some preliminary tests.

### 6.4.2 Tests with unnormalised data and “split databases”

The results that are presented, are performed with split databases and unnormalised data. The size of the data set is varied to determine the optimum configuration presented to the neural network. Each test (depicted by a single legend on the graph) was performed three times, and the best of the three results are reported. The legend that is contained on the side of the graphs is shown in the following figure.



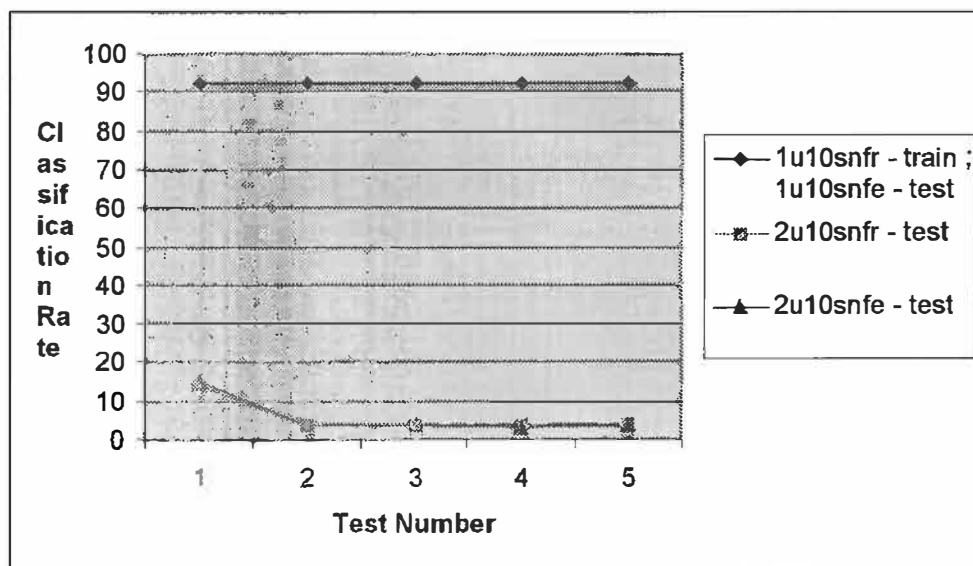
**Figure 6. 13:** Example of legends that is used in figures containing results.

It is possible that Figure 6.13 can be interpreted as 1u10snfr was used for training, which is correct. However, the assumption that the network was trained on 1u10snfr and tested on 1u10snfr is incorrect. The legend next to 1u10snfr actually relates to 1u10snfe. It must be emphasised that the network was trained on 1u10snfr and tested on 1u10snfe, and it is these results (i.e. from 1u10snfe) that are plotted on the figures.

It is important to note that the training file 1u10snfr contains only 41 distinct subjects, while the other 41 distinct subjects are contained in database 2. Hence, the network was trained only on the 41 subjects of database 1, and was not presented with the other 41 of database 2 during the training phase. Thus, the subjects of database 2 constitute “unseen data”. Since the network was only trained for the distinct subjects of database 1, it should only classify subjects that belong to that database and should not be able to classify (within some acceptable error) subjects of database 2.

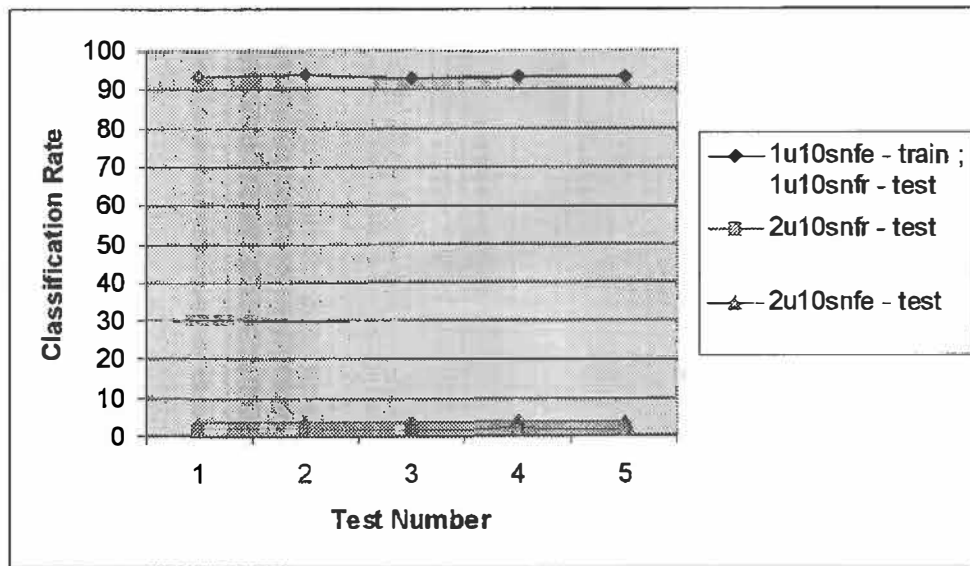


**NB:** This section has a long list of results (graphs), which is necessary to illustrate the performance of different pattern files. A summary of these results, and general conclusions is given at the end of this section.



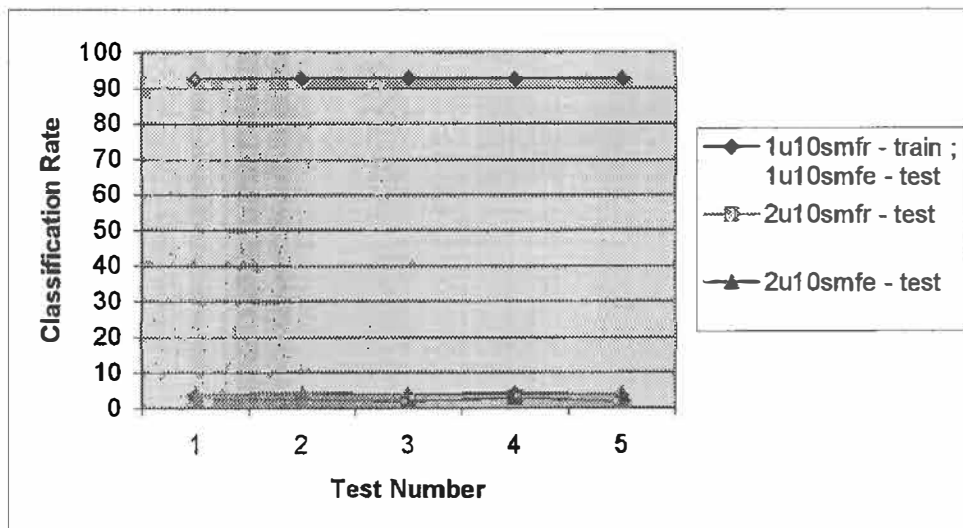
**Figure 6. 14:** Results of tests performed with 1u10snfr/e and 2u10snfr/e data (see Table D.20)

The architecture of the RBF network that was used to carry out tests reflected in Figures 6.14 – 6.21 (see Table D.20-D.27) consisted of 99 inputs and 41 outputs (there were 41 distinct subjects in the training and testing sets). The highest true positive rate obtained in Figure 6.14 was 92.2% when the network was trained on 1u10snfr data and tested on 1u10snfe data. The highest false positive rate was 14.9%.



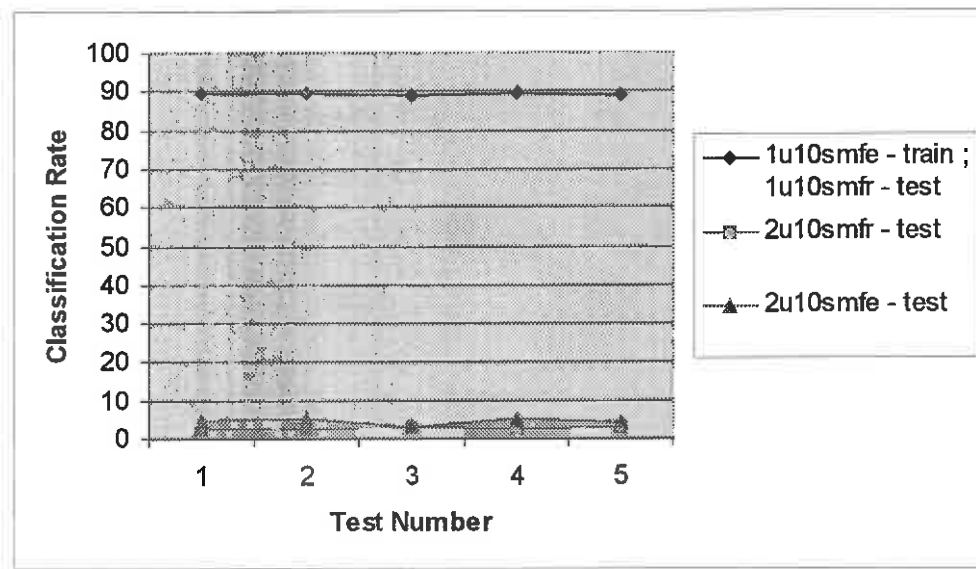
**Figure 6. 15:** Results of tests performed with 1u10snfr/e and 2u10snfr/e data (see Table D.21)

The highest true positive rate in Figure 6.15 was 93.9% when the network was trained on 1u10snfe and tested on 1u10snfr. The highest false positive rate was 3.7%.



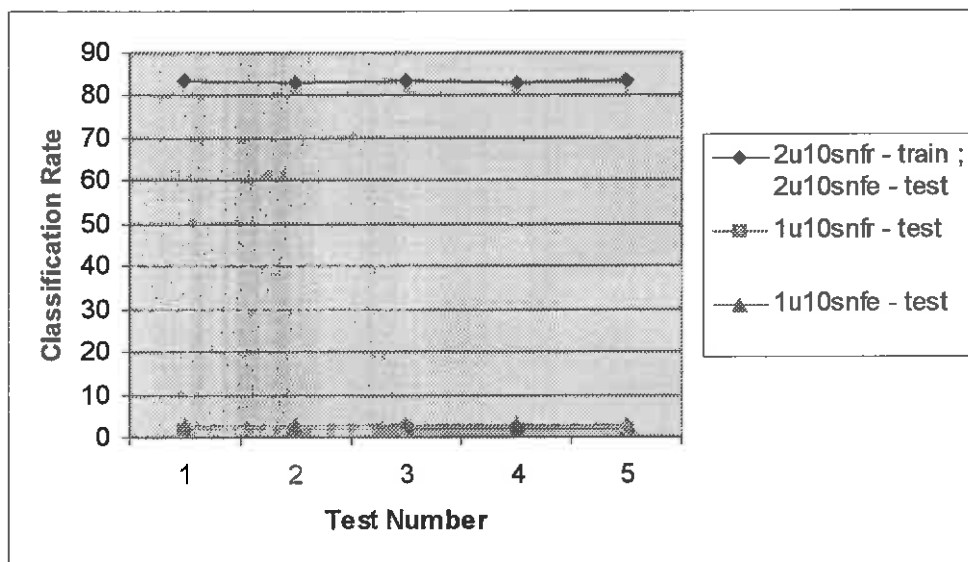
**Figure 6. 16:** Results of tests performed with 1u10smfr/e and 2u10smfr/e data (see Table D.22)

The highest true positive rate in Figure 6.16 was 92.7% when the network was trained on 1u10smfr and tested on 1u10smfe. The highest false positive rate was 4.2%.



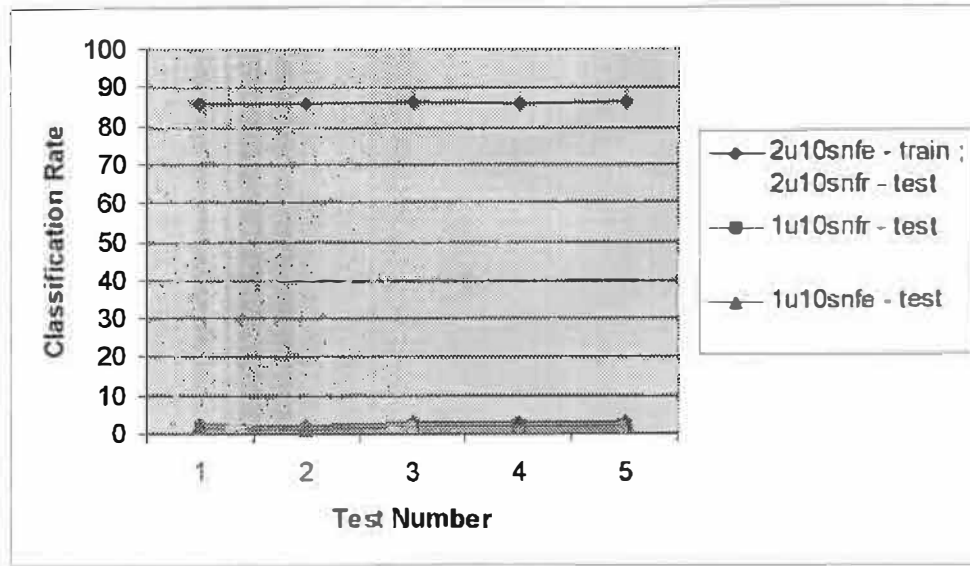
**Figure 6. 17 :** Results of tests performed with 1u10smfr/e and 2u10smfr/e data (see Table D.23)

The highest true positive rate in Figure 6.17 was 90.0% when the network was trained on 1u10smfe and tested on 1u10smfr. The highest false positive rate was 5.1%.



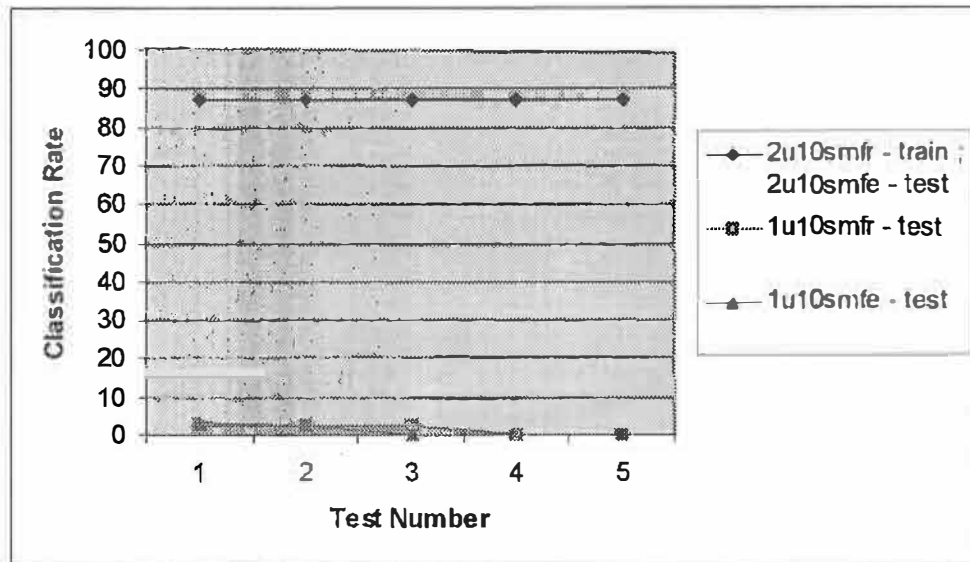
**Figure 6. 18:** Results of tests performed with 2u10snfr/e and 1u10snfr/e data (see Table D.24)

The highest true positive rate in Figure 6.18 was 83.3% when the network was trained on 2u10snfr and tested on 2u10snfe. The highest false positive rate was 2.9%.



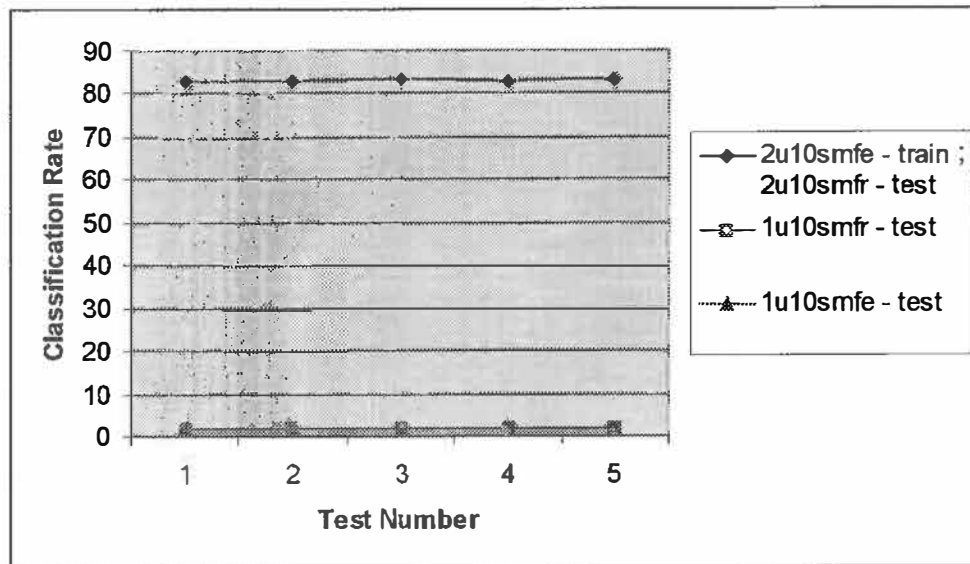
**Figure 6. 19:** Results of tests performed with 2u10snfr/e and 1u10snfr/e data (see Table D.25)

The highest true positive rate in Figure 6.19 was 86.5% when the network was trained on 2u10snfe and tested on 2u10snfr. The highest false positive rate was 2.9%.



**Figure 6. 20:** Results of tests performed with 2u10smfr/e and 1u10smfr/e data (see Table D.26)

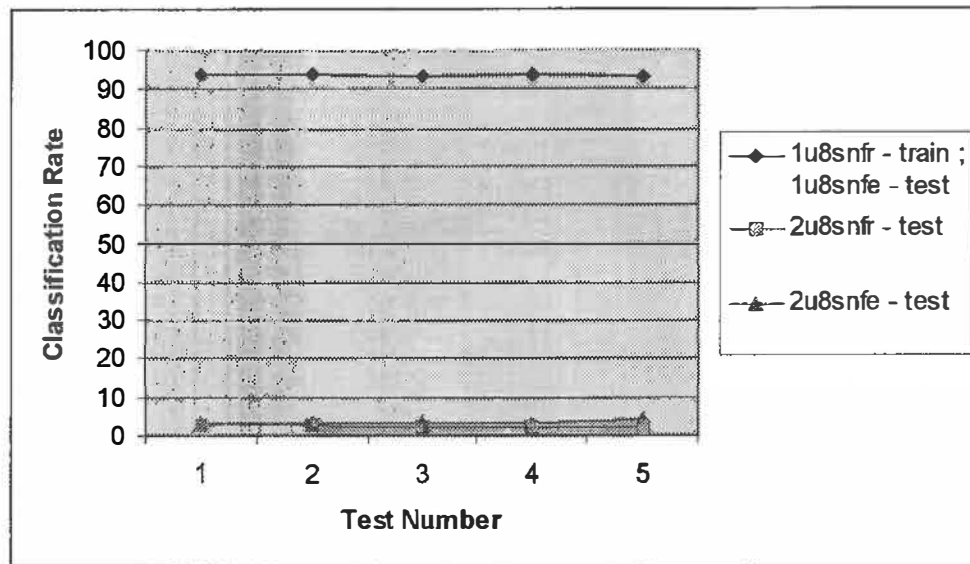
The highest true positive rate in Figure 6.20 was 87.0% when the network was trained on 2u10smfr and tested on 2u10smfe. The highest false positive rate was 2.9%.



**Figure 6. 21:** Results of tests performed with 2u10smfr/e and 1u10smfr/e data (see Table D.27)

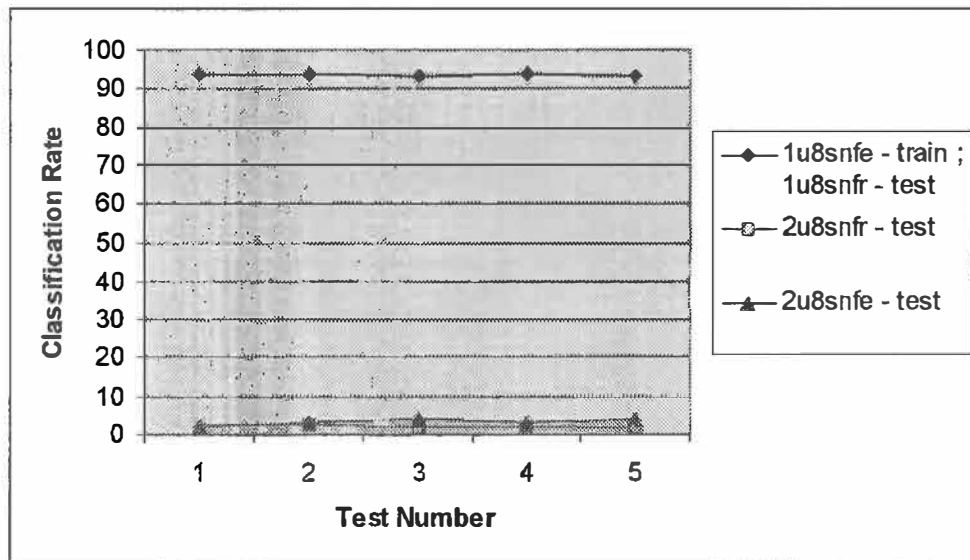
The highest true positive rate in Figure 6.21 was 83.3% when the network was trained on 2u10smfe and tested on 2u10smfr. The highest false positive rate was 2.0%.

The architecture of the RBF network that was used to carry out tests reflected in Figures 6.22 – 6.29 (see Table D.28-D.35) consisted of 63 inputs and 41 outputs (there were 41 distinct subjects in the training and testing sets).



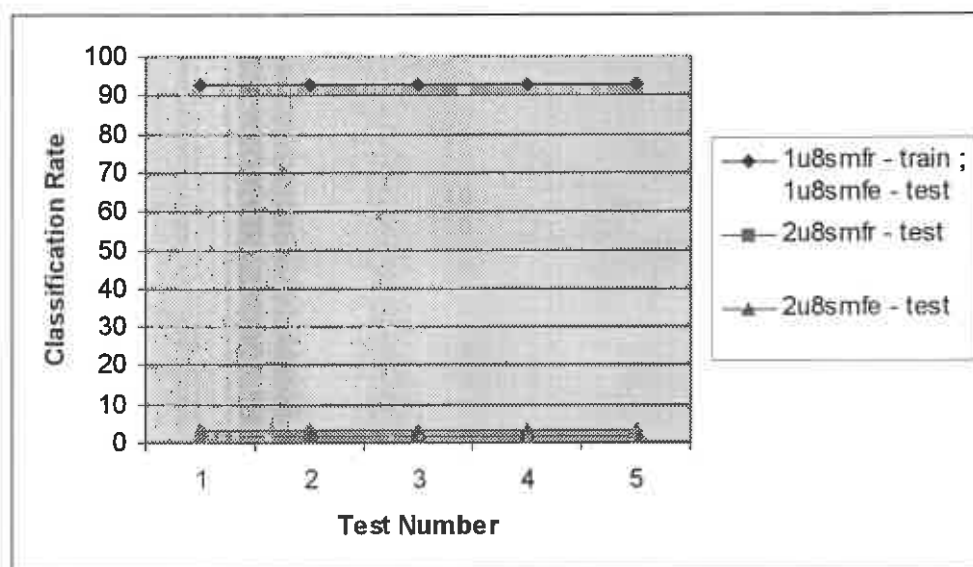
**Figure 6. 22:** Results of tests performed with 1u8snfr/e and 2u8snfr/e data (see Table D.28)

The highest true positive rate in Figure 6.22 was 93.0% when the network was trained on 1u8snfr and tested on 1u8snfe. The highest false positive rate was 3.7%.



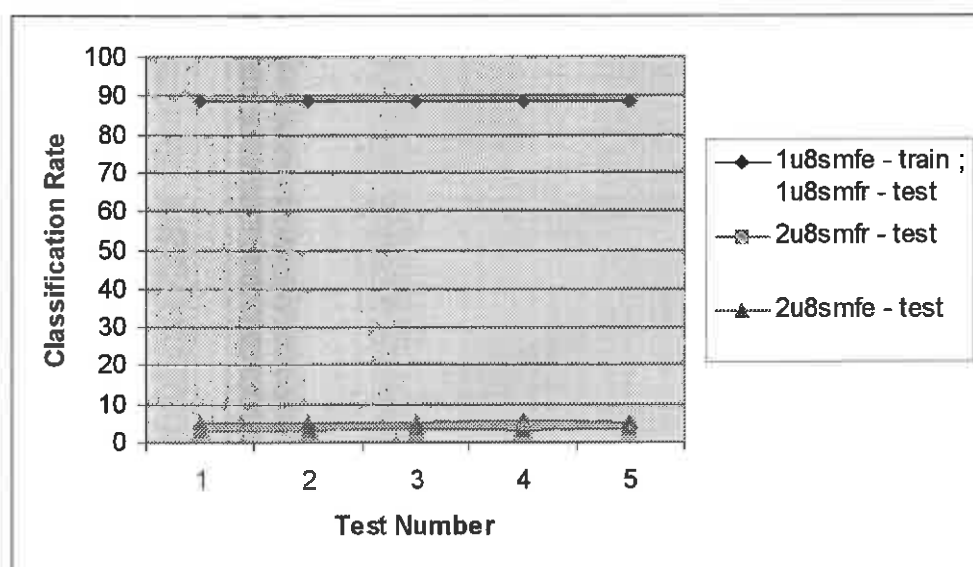
**Figure 6. 23 :** Results of tests performed with 1u8snfr/e and 2u8snfr/e data (see Table D.29)

The highest true positive rate in Figure 6.23 was 93.9% when the network was trained on 1u8snfr and tested on 1u8snfe. The highest false positive rate was 4.2%.



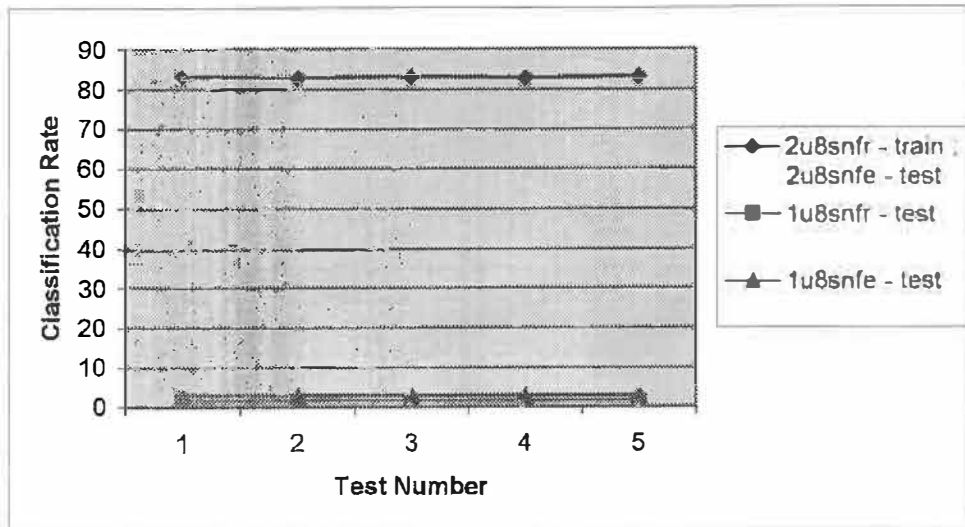
**Figure 6. 24:** Results of tests performed with 1u8smfr/e and 2u8smfr/e data (see Table D.30)

The highest true positive rate in Figure 6.24 was 92.7% when the network was trained on 1u8smfr and tested on 1u8smfe. The highest false positive rate was 3.3%.



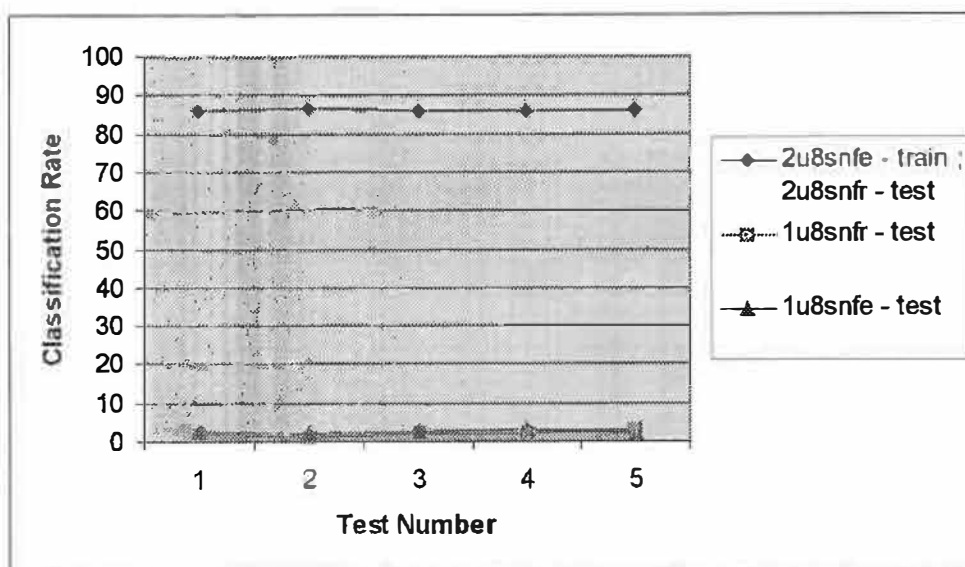
**Figure 6. 25:** Results of tests performed with 1u8smfr/e and 2u8smfr/e data (see Table D.31)

The highest true positive rate in Figure 6.25 was 88.6% when the network was trained on 1u8smfe and tested on 1u8smfr. The highest false positive rate was 5.1%.



**Figure 6. 26:** Results of tests performed with 2u8snfr/e and 1u8snfr/e data (see Table D.32)

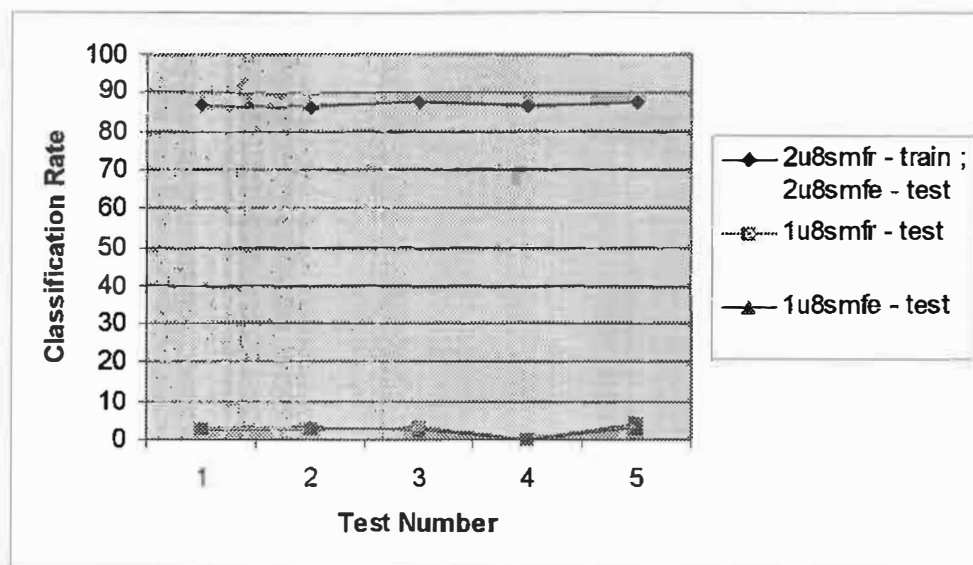
The highest classification rate in Figure 6.26 was 83.7% when the network was trained on 2u8snfr and tested on 2u8snfe. The highest false positive rate was 2.9%.



**Figure 6. 27:** Results of tests performed with 2u8snfr/e and 1u8snfr/e data (see Table D.33)

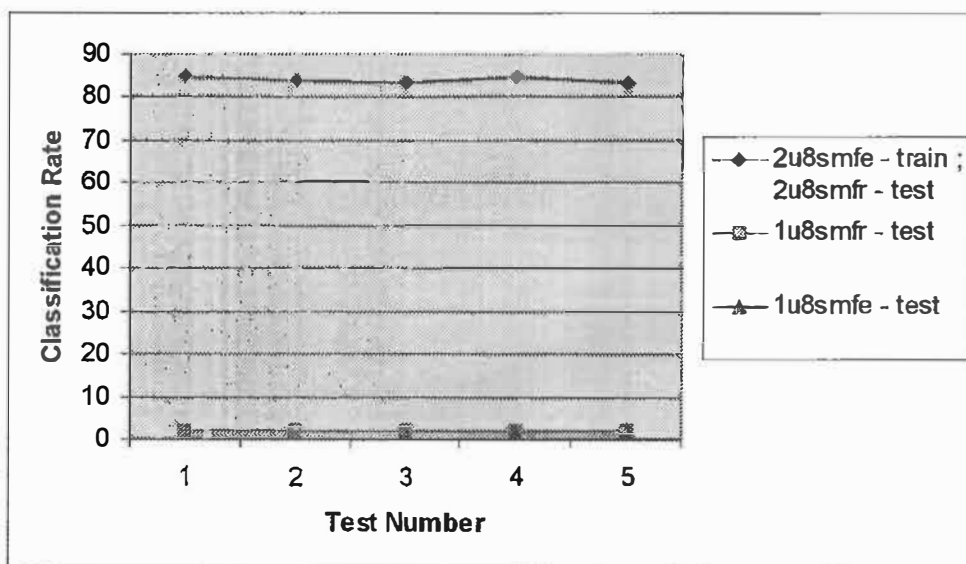


The highest classification rate in Figure 6.27 was 86.5% when the network was trained on 2u8snfe and tested on 2u8snfr. The highest false positive rate was 2.9%.



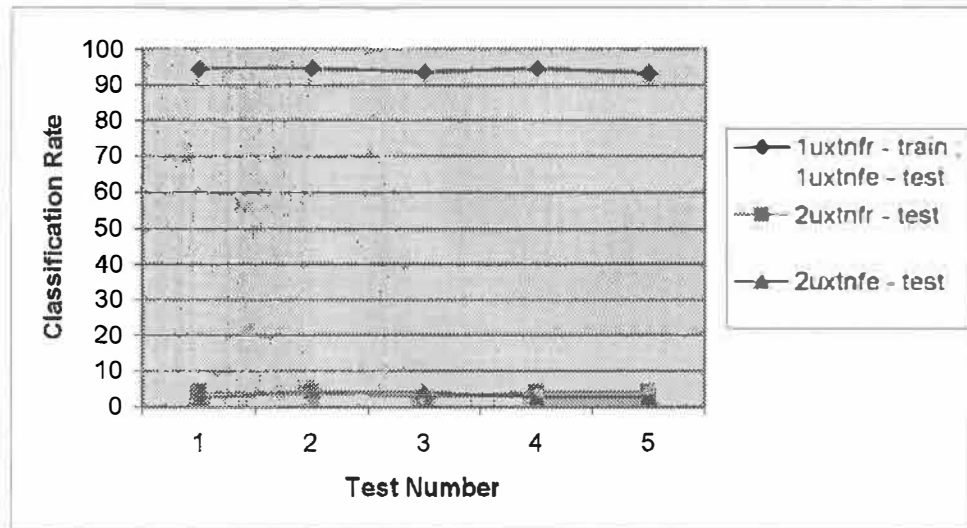
**Figure 6. 28:** Results of tests performed with 2u8smfr/e and 1u8smfr/e data (see Table D.34)

The highest classification rate in Figure 6.28 was 87.4% when the network was trained on 2u8smfr and tested on 2u8smfe. The highest false positive rate was 4.1%.



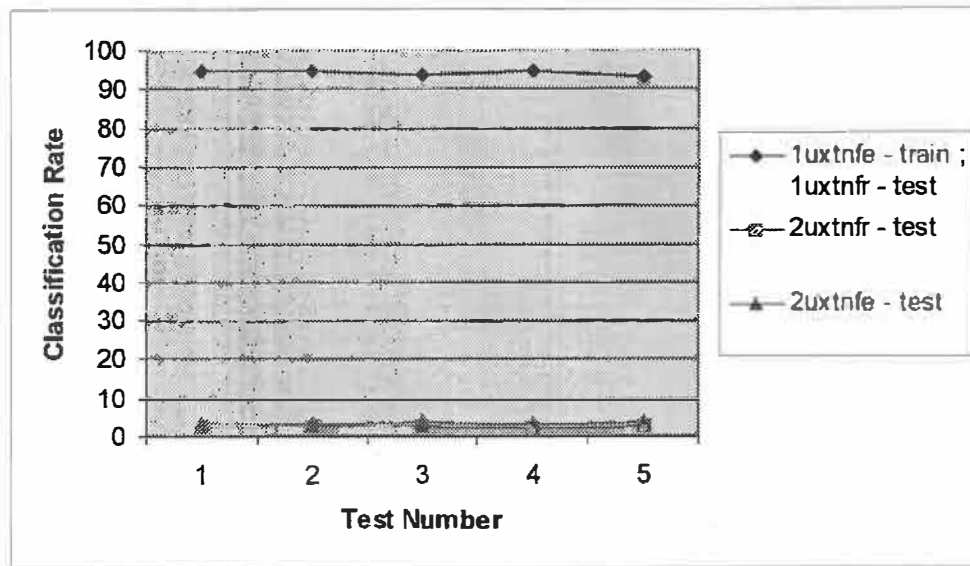
**Figure 6. 29:** Results of tests performed with 2u8smfr/e and 1u8smfr/e data (see Table D.35)

The highest classification rate in Figure 6.29 was 84.7% when the network was trained on 2u8smfe and tested on 2u8smfr. The highest false positive rate was 2.0%.



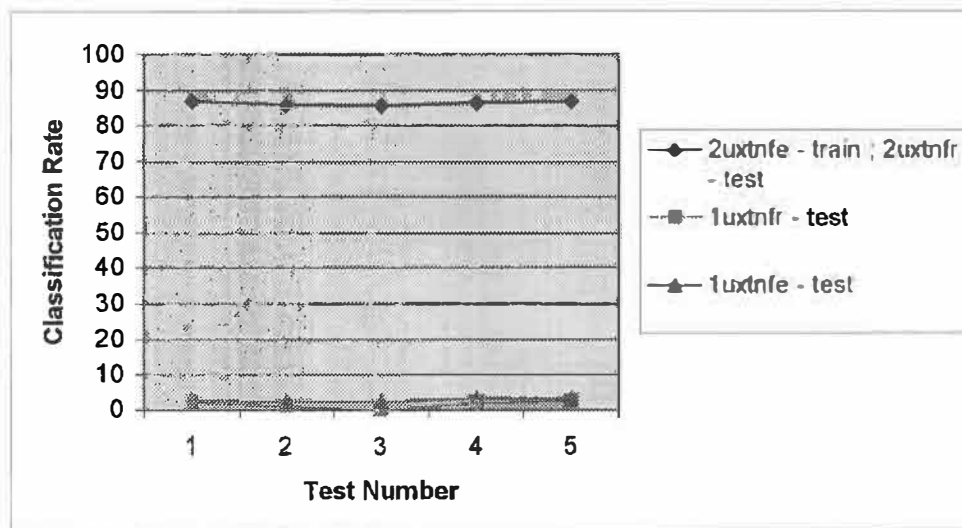
**Figure 6. 30:** Results of tests performed with 1uxtnfr/e and 2uxtnfr/e data (see Table D.36)

The architecture of the RBF network that was used to carry out tests reflected in Figures 6.30 – 6.33 (see Table D.36-D.39) consisted of 54 inputs and 41 outputs (there were 41 distinct subjects in the training and testing sets). The highest classification rate in Figure 6.30 was 93.5% when the network was trained on 1uxtnfr and tested on 1uxtnfe. The highest false positive rate was 4.2%.



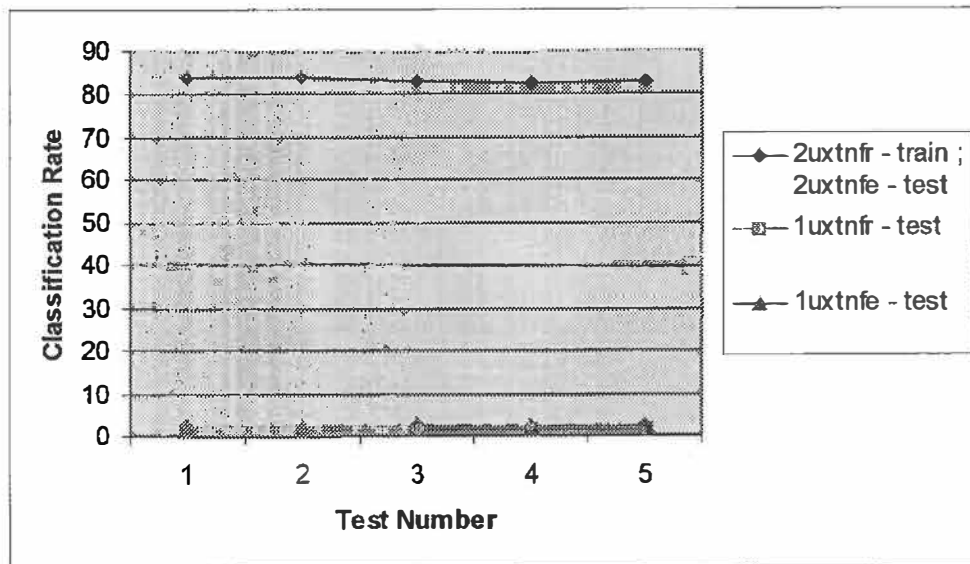
**Figure 6. 31:** Results of tests performed with 1uxtne/e and 2uxtne/e data (see Table D.37)

The highest classification in Figure 6.31 was 94.7% when the network was trained on 1uxtne and tested on 1uxtne. The highest false positive rate was 3.3%.



**Figure 6. 32:** Results of tests performed with 2uxtne/e and 1uxtne/e data (see Table D.39)

The highest classification rate in Figure 6.32 was 87.0% when the network was trained on 1u8smfr and tested on 1u8smfe. The highest false positive rate was 3.3%.



**Figure 6. 33:** Results of tests performed with 2uxtnfr/e and 1uxtnfr/e data (see Table D.38)

The highest classification rate in Figure 6.33 was 83.7% when the network was trained on 2nxtnfr and tested on 2nxtnfe. The highest false positive rate was 2.4%.

**Table 6. 5:** Summary of results for tests done with split databases and unnormalised data.

Pattern file network was trained on	Pattern file network was tested on	Highest true pos. rate (%)	Highest false pos. rate (%)	Figure Number.
1u10snfr	1u10snfe, 2u10snfr, 2u10snfe	92.2	14.9	6.14
2u10snfr	2u10snfr, 1u10snfr, 1u10snfe	83.3	2.9	6.18
1u10snfe	1u10snfr, 2u10snfe, 2u10snfr	93.9	3.7	6.15
2u10snfe	2u10snfr, 1u10snfe, 1u10snfr	86.5	2.9	6.19
1u10smfr	1u10smfe, 2u10smfr, 2u10smfe	92.7	4.2	6.16
2u10smfr	2u10smfe, 1u10smfe, 1u10smfr	87.0	2.9	6.20
1u10smfe	1u10smfr, 2u10smfr, 2u10smfe	90.0	5.1	6.17
2u10smfe	2u10smfr, 1u10smfr, 1u10smfe	83.3	2.0	6.21
1u8smfr	1u8smfe, 2u8smfr, 2u8smfe	92.7	3.3	6.24
2u8smfr	2u8smfe, 1u8smfe, 1u8smfr	87.4	4.1	6.28
1u8smfe	1u8smfr, 2u8smfr, 2u8smfr	88.6	5.1	6.25
2u8smfe	2u8smfr, 1u8smfr, 2u8smfe	84.7	2.0	6.29

1u8snfr	1u8snfe, 2u8snfr, 2u8snfe	93.0	3.7	6.22
2u8snfr	2u8snfe, 1u8snfe, 1u8snfr	83.7	2.9	6.26
1u8snfe	1u8snfr, 2u8snfr, 2u8snfe	93.9	4.2	6.23
2u8snfe	2u8snfr, 1u8snfe, 1u8snfr	86.5	2.9	6.27
1uxtifr	1uxtife, 2uxtifr, 2uxtife	93.5	4.2	6.30
2uxtifr	2uxtife, 1uxtifr, 1uxtife	83.7	2.4	6.33
1uxtife	1uxtifr, 2uxtifr, 2uxtife	94.7	3.3	6.31
2uxtife	2uxtifr, 1uxtifr, 1uxtife	87.0	3.3	6.32

**From Table 6.5, and the relevant figures, the following conclusions can be made :**

- An interesting observation of Table 6.5 is that networks that were trained on files that contained remaining view information (pattern files that have an “e” suffix, and no mixed views) and tested on the  $\frac{3}{4}$  views (pattern files that have an “r” suffix, and no mixed views) performed better than if the network was trained the other way around. One of the highest true positive recognition rates in Table 6.5 (see Figures 6.31 and 6.30 ) was obtained when the network was trained/tested on pattern files described above. Hence for a split database and unnormalised data, it can be generally concluded that it is better to train the network on pattern files that contain remaining views, and test on the  $\frac{3}{4}$  views.
- In all of the tests, non-mixed views data generally performed better than mixed views data. Mixed views didn’t enhance the generalisation ability of the network.

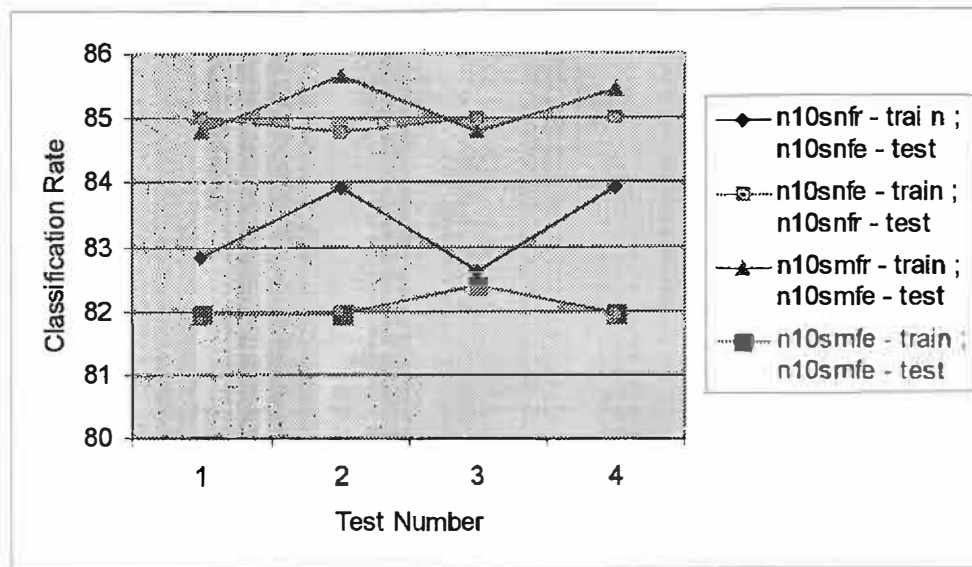
Perhaps one of the most important observations of the results reported in Table 6.5 is the low false positive recognition rate that was obtained on all the networks. This is an encouraging point, since it clearly demonstrates the networks ability to generalise, and reject impostors. This result can be attributed to the manner in which the RBF-DDA operates. A radial basis function is placed over the feature that is to be classified. After training, features that don’t fall in the boundary of the radial basis function, are

automatically rejected (i.e. classified as a negative). This powerful characteristic of the RBF neural network allows the low false positive recognition rates to be obtained.

The previous section saw the data that was contained in the 10x10 region performing the best (i.e. 88.3%, as compared to the 87.8% of the data from the triangular region). In these set of results however, it is data from the triangular region (i.e. 1uxtnfe, 1uxtnfr, 2uxtnfe and 2uxtnfr) that performs the best. In the previous section, there were 82 outputs (i.e. the network had to discriminate between 82 different classes) and in this section, the maximum number of outputs is limited to 41. It can be argued that fewer features are required to perform efficient classification when there are fewer outputs. The network would need more information to perform classification when there are a larger number of classes, hence the discrepancies in data set performance between the previous section and the current one. For a fewer number of classes, data from the triangular region forms better pattern boundaries.

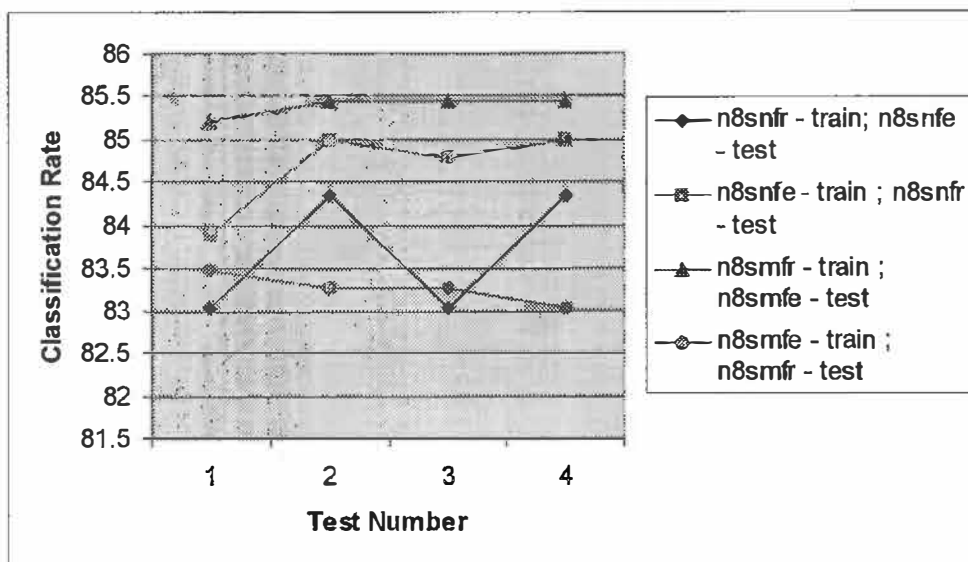
### 6.4.3 Tests with normalised data

The tests performed in this section, are exactly like those performed in §6.4.1 (Tests with unnormalised data), however, the data used in the pattern files is now normalised (see equation 6-1). The legends have the same function as described in §6.4.1. Each legend depicts a specific network that was reset, initialised, trained and tested by the data that is specified by the legend.



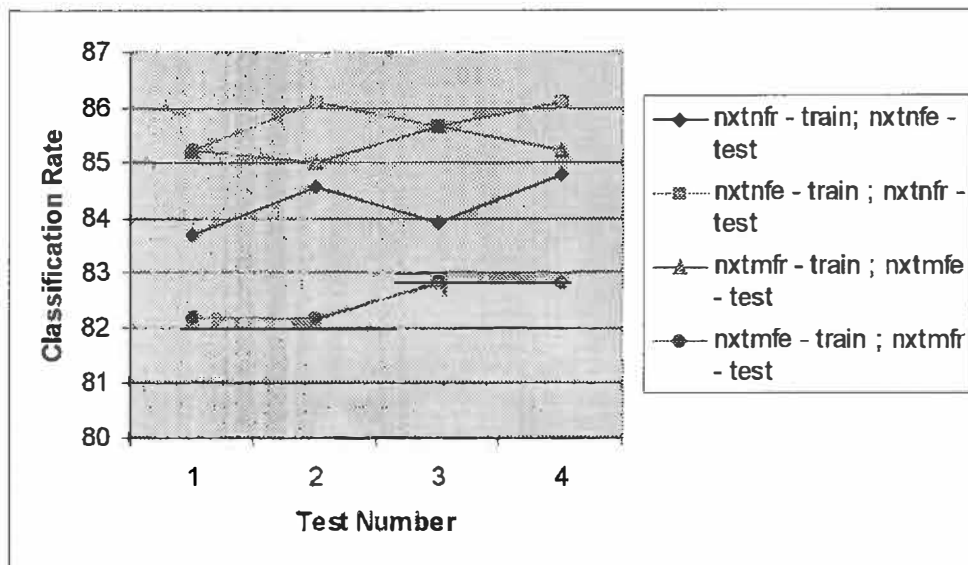
**Figure 6. 34:** Results of tests performed with n10snfr/e and n10smfr/e data (see Tables D.43-D.46)

The architecture of the network that was used to carry out the investigations in Figure 6.34 consisted of 99 inputs and 82 outputs (there are 82 distinct subjects in the training and testing sets). The DDA algorithm automatically inserts hidden units (see §5.10 on RBF with DDA in Chapter 5).



**Figure 6. 35:** Results of tests performed with n8snfr/e and n8smfr/e data (see Tables D.47-D.50)

The architecture of the network that was used to carry out the investigations in Figure 6.35 consisted of 63 inputs and 82 outputs (there are 82 distinct subjects in the training and testing sets). The DDA algorithm automatically inserts hidden units (see §5.10 on RBF with DDA in Chapter 5).



**Figure 6. 36:** Results of tests performed with nxtnfr/e and nxtnmfr/e data (see Table D.51-D.54)

The architecture of the network that was used to carry out the investigations in Figure 6.36 consisted of 54 inputs and 82 outputs (there are 82 distinct subjects in the training and testing sets). The DDA algorithm automatically inserts hidden units (see §5.10 on RBF with DDA in Chapter 5).

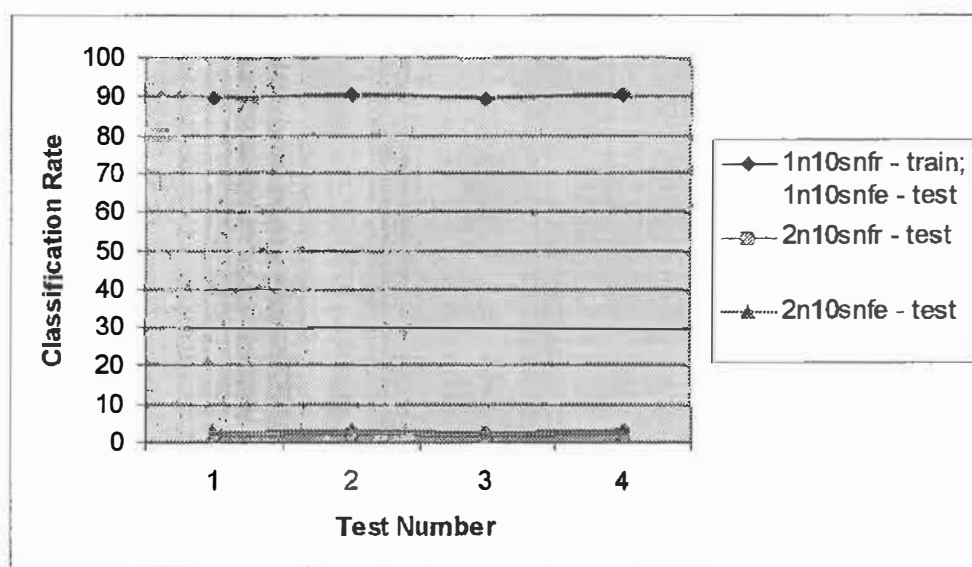
In Figure 6.34, the highest classification rate (true positive rate) obtained was 85.7% when the network was trained on n10smfr data and tested on n10smfe data. The highest true positive rate obtained in Figure 6.35 was 85.4% when the network was trained on n8smfr data and tested on n8smfe data. The highest classification rate obtained was 86.1% when the network was trained on nxtnfe data and tested on nxtnfr data (i.e. trained on pattern files containing remaining views and tested on  $\frac{3}{4}$  views).

There is a decrease in performance with all the normalised pattern files (compared to their unnormalised counterparts). The decrease in performance in the networks that



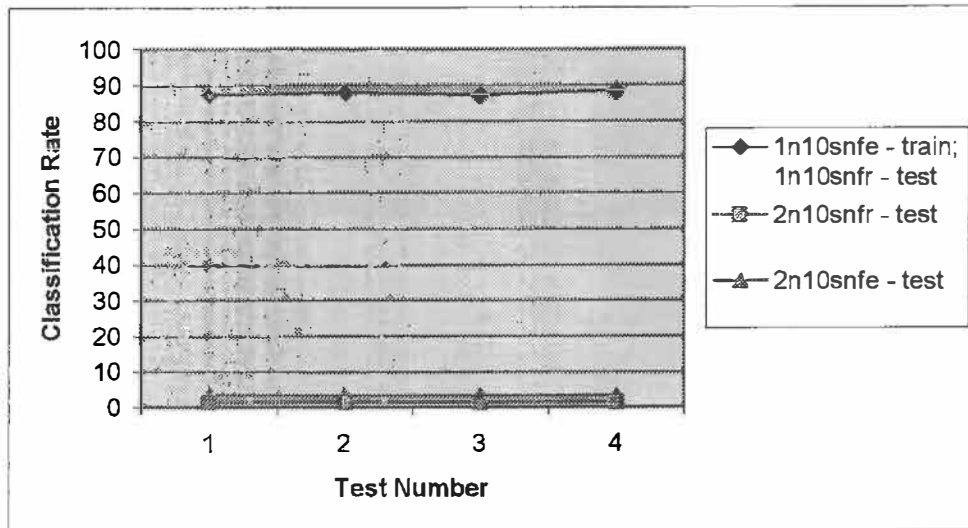
used non-mixed views data is far more significant than those that used mixed views data. Normalisation of a vector changes its magnitude, however preserves its direction. The results suggest that the magnitude of the unnormalised data in the non-mixed views pattern files was a feature that the neural network used in its discrimination of various patterns. The performance of the normalised pattern files that contained mixed-views data is not much better than their unnormalised counterparts (see §6.4.1). From Figures 6.34-6.36 it can be seen that networks that were trained on non-mixed views data (containing remaining views) and tested on the  $\frac{3}{4}$  views performed better than if they were trained with the latter pattern file and tested with the former.

#### 6.4.4 Tests with split databases and normalised data



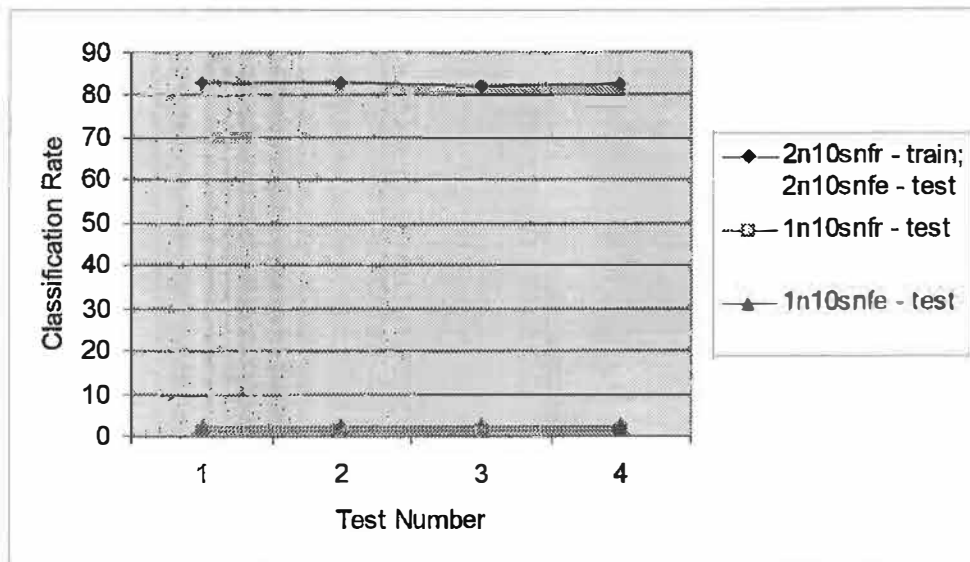
**Figure 6. 37:** Results of tests performed with 1n10snfr/e and 2n10snfr/e data (see Table D.55)

The highest classification rate in Figure 6.37 was 90.6% when the network was trained on 1n10snfr and tested on 1n10snfe. The highest false positive rate was 3.3%.



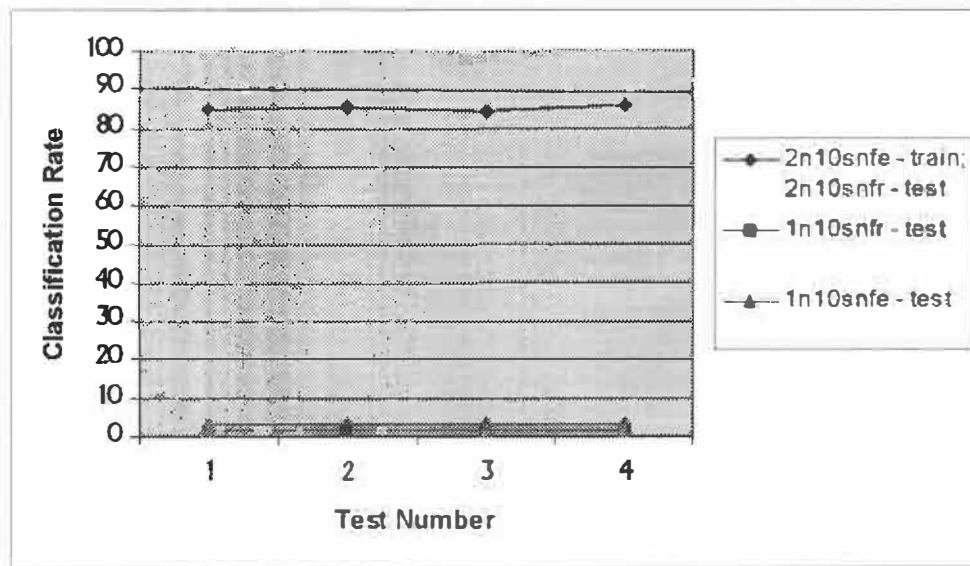
**Figure 6. 38:** Results of tests performed with 1n10snfr/e and 2n10snfr/e data (see Table D.56)

The highest classification rate in Figure 6.38 was 88.6% when the network was trained on 1n10snfe and tested on 1n10snfr. The highest false positive rate was 3.7%.



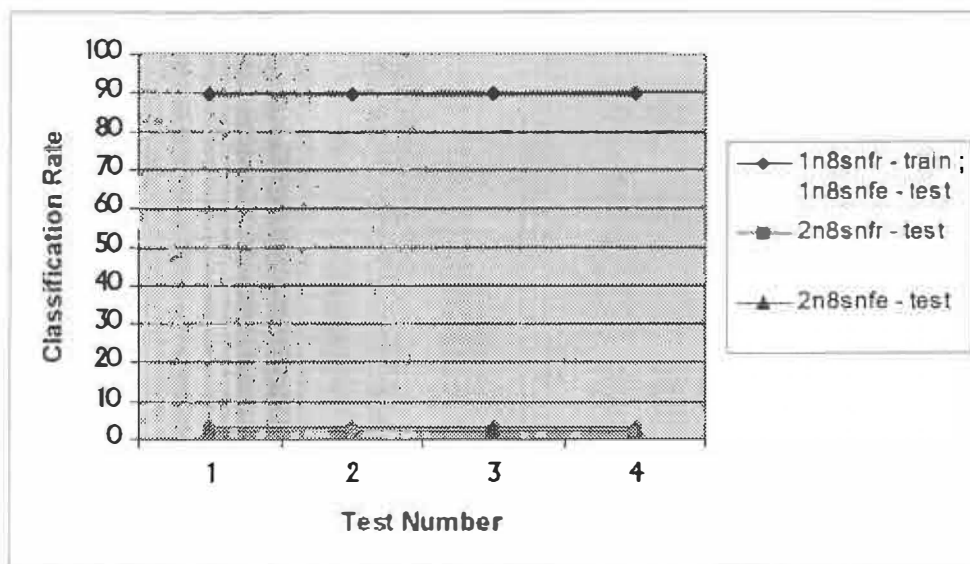
**Figure 6. 39:** Results of tests performed with 1n10snfr/e and 2n10snfr/e data (see Table D.57)

The highest classification rate in Figure 6.39 was 82.8% when the network was trained on 2n10snfr and tested on 2n10snfe. The highest false positive rate was 2.4%.



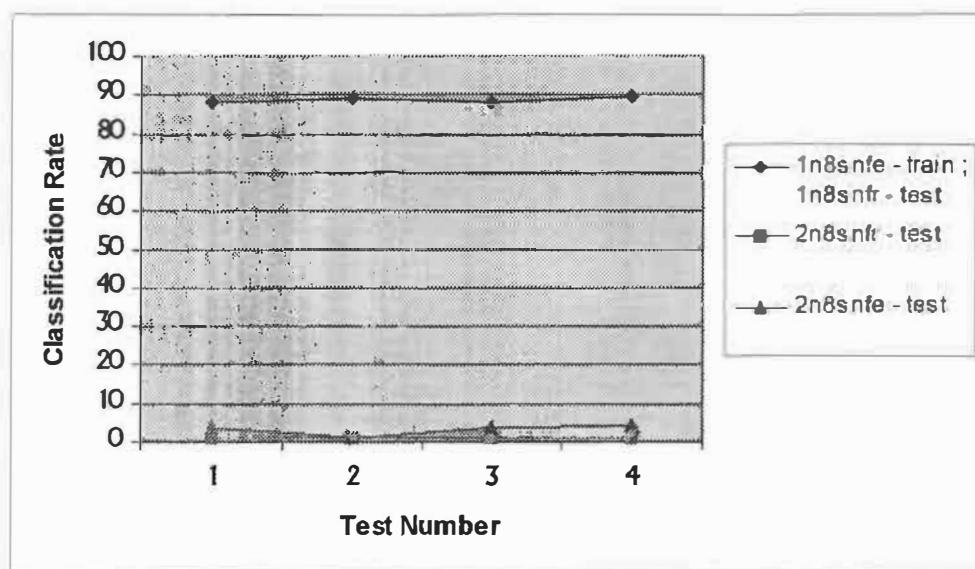
**Figure 6. 40:** Results of tests performed with 1n10snfr/e and 2n10snfr/e data (see Table D.58)

The highest classification rate in Figure 6.40 was 86.0% when the network was trained on 2n10snfe and tested on 2n10snfr. The highest false positive rate was 3.3%.



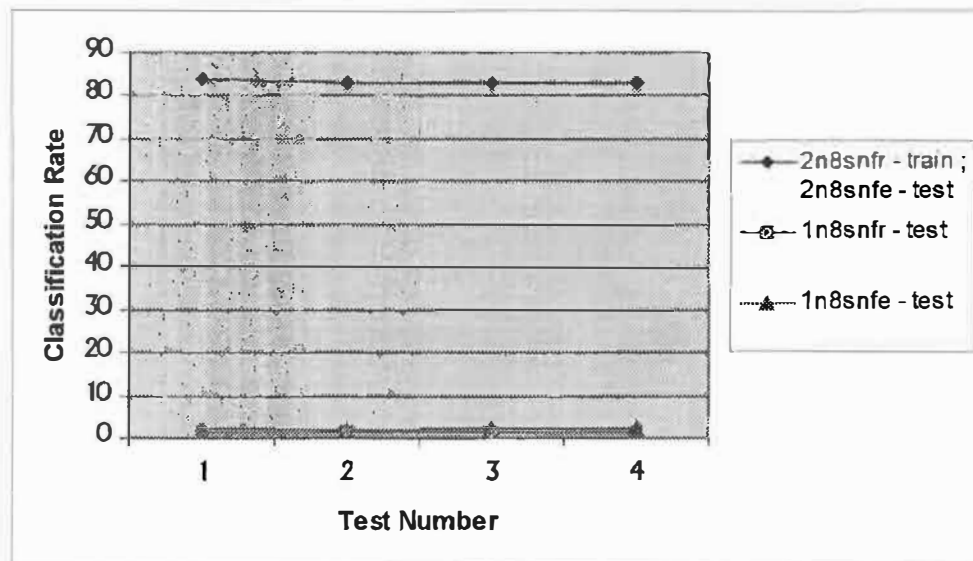
**Figure 6. 41:** Results of tests performed with 1n8snfr/e and 2n8snfr/e data (see Table D.59)

The highest classification rate in Figure 6.41 was 89.8% when the network was trained on 1n8snfr and tested on 1n8snfe. The highest false positive rate was 3.3%.



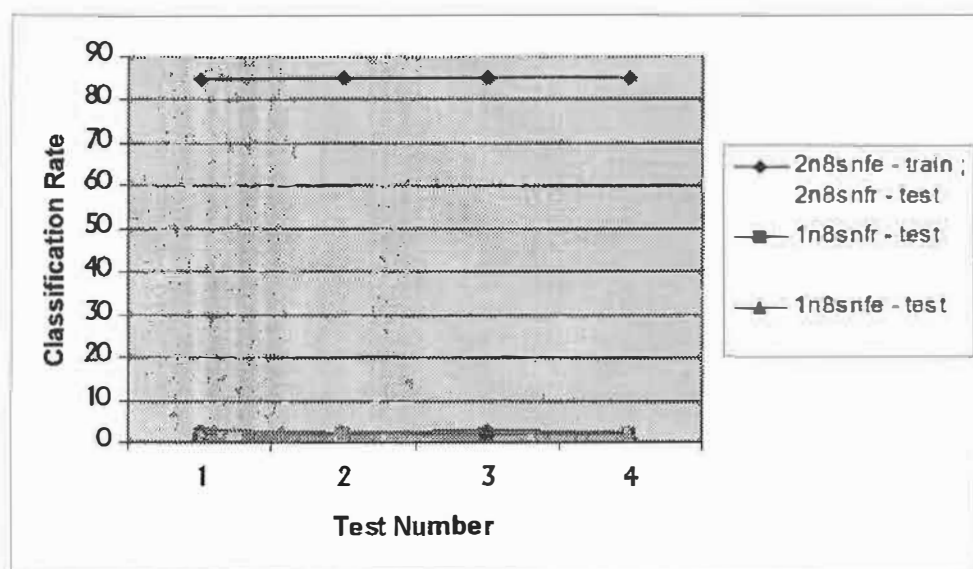
**Figure 6. 42:** Results of tests performed with 1n8snfr/e and 2n8snfr/e data (see Table D.60)

The highest classification rate in Figure 6.42 was 89.4% when the network was trained on 1n8snfe and tested on 1n8snfr. The highest false positive rate was 4.2%.

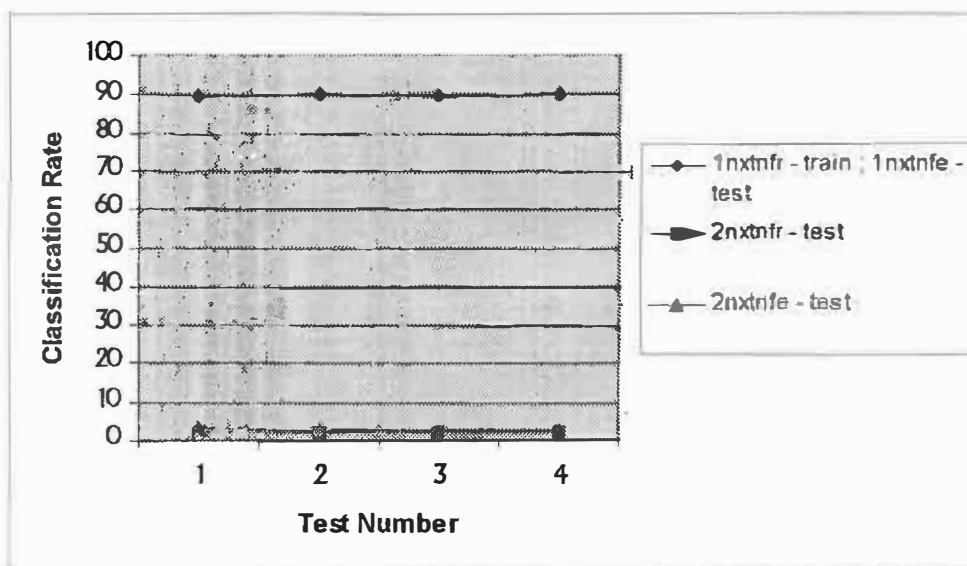


**Figure 6. 43:** Results of tests performed with 2n8snfr/e and 1n8snfr/e data (see Table D.61)

The highest classification rate in Figure 6.43 was 83.7% when the network was trained on 2n8snfr and tested on 2n8snfe. The highest false positive rate was 2.5%.

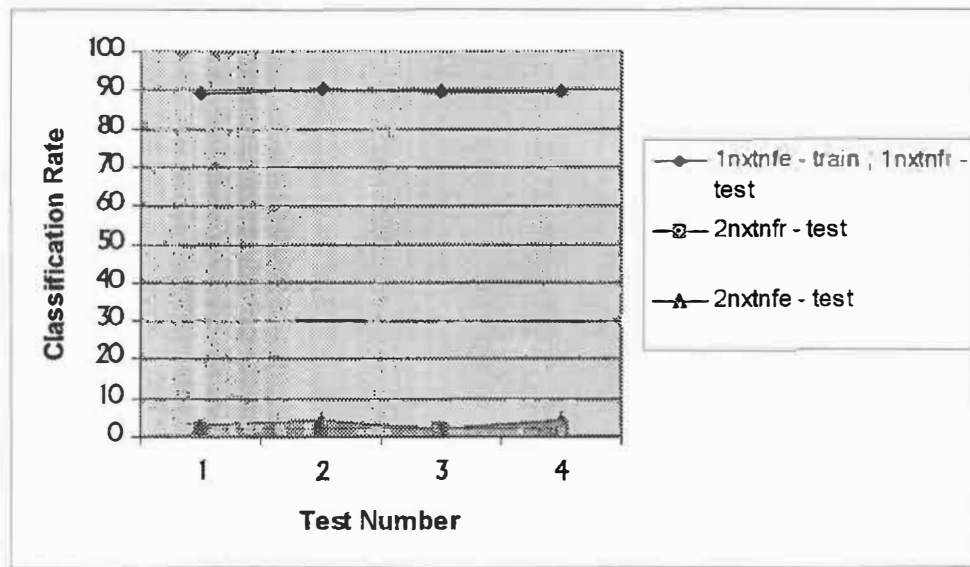


**Figure 6. 44:** Results of tests performed with 2n8snfr/e and 1n8snfr/e data (see Table D.62)  
The highest classification rate in Figure 6.44 was 85.1% when the network was trained on 2n8snfr and tested on 2n8snfe. The highest false positive rate was 2.4%.



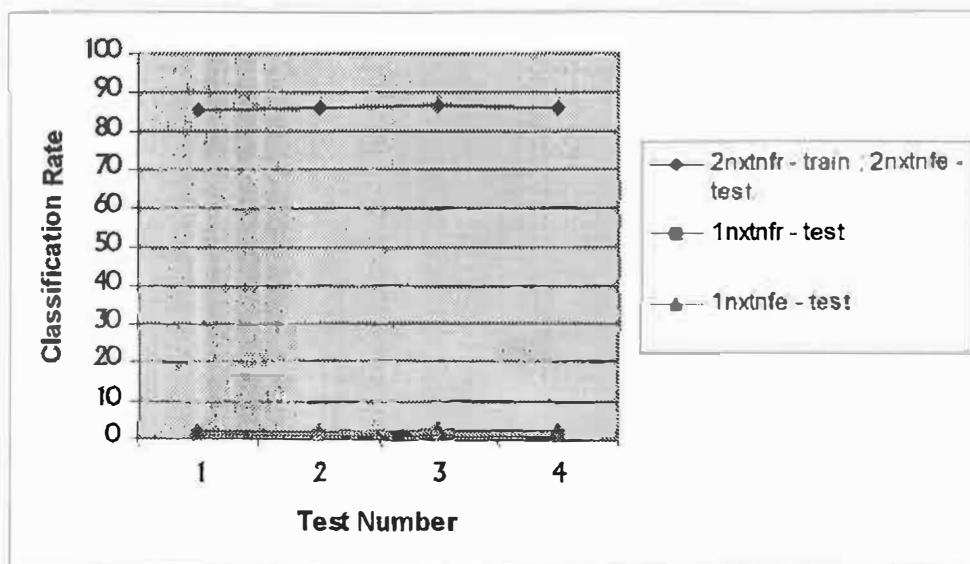
**Figure 6. 45 :** Results of tests performed with 1nxtnfr/e and 2nxtnfr/e data (see Table D.63)

The highest classification rate in Figure 6.45 was 90.2% when the network was trained on 1nxtnfr and tested on 1nxtnfe. The highest false positive rate was 3.3%.



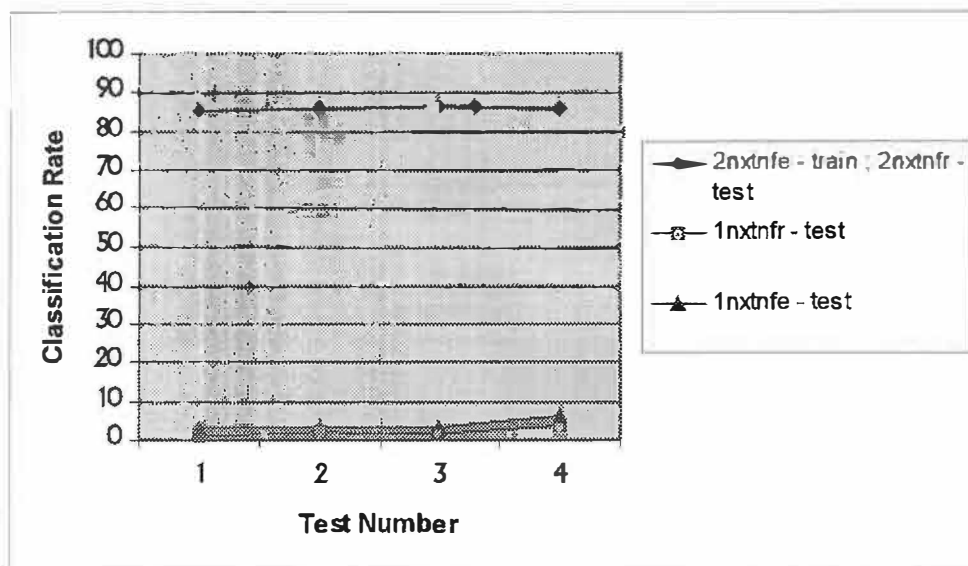
**Figure 6. 46:** Results of tests performed with 1nxtnfr/e and 2nxtnfr/e data (see Table D.64)

The highest classification rate in Figure 6.46 was 90.2% when the network was trained on 1nxtnfe and tested on 1nxtnfr. The highest false positive rate was 4.2%.



**Figure 6. 47:** Results of tests performed with 2nxtnfr/e and 1nxtnfr/e data (see Table D.65)

The highest classification rate in Figure 6.47 was 83.3% when the network was trained on 2nxtnfr and tested on 2nxtnfe. The highest false positive rate was 2.4%.



**Figure 6. 48:** Results of tests performed with 2nxtnfr/e and 1nxtnfr/e data (see Table D.66)  
The highest classification rate in Figure 6.48 was 86.5% when the network was trained on 2nxtnfr and tested on 2nxtnfe. The highest false positive rate was 6.0%.

**Table 6. 6:** Summary of results for tests performed with "split databases" and normalised data.

Pattern file network was trained on	Pattern file network was tested on	Highest true pos. rate (%)	Highest false pos. rate (%)	Figure Number
1n10snfr	1n10snfe, 2n10snfr, 2n10snfe	90.6	3.3	6.37
2n10snfr	2n10snfr, 1n10snfr, 1n10snfe	82.8	2.4	6.39
1n10snfe	1n10snfr, 2n10snfe, 2n10snfr	88.6	3.7	6.38
2n10snfe	2n10snfr, 1n10snfe, 1n10snfr	86.0	3.3	6.40
1n8snfr	1n8snfe, 2n8snfr, 2n8snfe	89.8	3.3	6.41
2n8snfr	2n8snfe, 1n8snfr, 1n8snfe	86.7	2.5	6.43
1n8snfe	1n8snfr, 2n8snfr, 2n8snfe	89.4	4.2	6.42
2n8snfe	2n8snfr, 1n8snfr, 1n8snfe	85.1	2.4	6.44
1nxtnfr	1nxtnfe, 2nxtnfr, 2nxtnfe	90.2	3.3	6.45
2nxtnfr	2nxtnfe, 1nxtnfr, 1nxtnfe	83.3	2.4	6.47
1nxtnfe	1nxtnfr, 2nxtnfr, 2nxtnfe	90.2	4.2	6.46
2nxtnfe	2nxtnfr, 1nxtnfr, 1nxtnfe	86.5	6.0	6.48

The highest recognition rate obtained was 90.6% when the network was trained with 1n10snfr and tested with 1n10snfe. The corresponding database 2's performance was 82.8% and 2.4%. When considering results from split databases, it is important to consider the results of both databases (i.e. when considering the results of 1u10snfr, the results of 2u10snfr should be examined as well). Closer examination of the results show that the networks that were trained on 1nxtnfe and 2nxtnfe yielded true positive recognition rates of 90.2% and 86.5% respectively and true negative recognition rates of 4.2% and 6.0% respectively. These figures suggest a better performance compared to networks trained with just 1n10snfr and 2n10snfr. The normalised data did not perform as well as the unnormalised data with split databases. As in §6.4.2, data from the triangular region performed the best.

It is clear that the results obtained with split databases are better than those obtained from full databases. There are fewer pattern classes with split databases and subsequently clearer pattern boundaries. Hence, the network should be able to discriminate between the different patterns more efficiently. However, this theory applies only to database 1, as reflected in the results. Database 2's results were consistently lower than those of database 1. Although the number of patterns decreased, it is possible that the pattern boundaries for the data-set lay too close to each other, thus making it difficult for the network to discriminate between the different patterns.

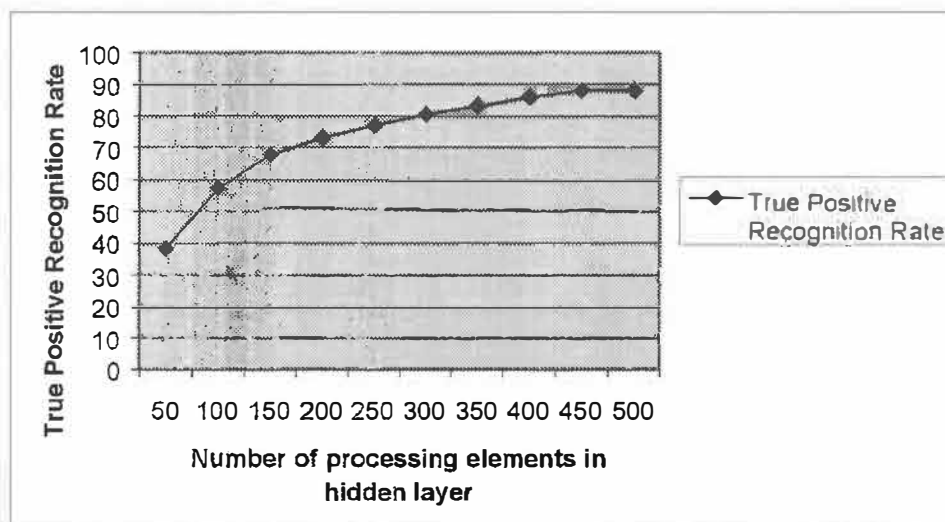
It was also found that results (see Figures 6.10-6.12 and Table 6.5) obtained with mixed views data (characterised by a letter "m" in the file name) showed that mixed views didn't enhance the generalisation ability of the network. Tests performed with mixed views cannot be used to ascertain whether ¾ views are better suited to the training set or testing set. From Figures 6.10-6.12 and Table 6.5, it is clear that these views are better suited to the testing set, rather than the training set.

Observation of Table 6.5 and 6.6 led to only certain data-sets being used in the tests with the counter-propagation network. These datasets are reflected in Tables 6.7 and 6.8.



## 6.5 Counter-propagation Network

The counter-propagation network of *Hecht-Nielsen*[11] was also investigated in an attempt to improve the classification rates obtained from the RBF-DDA networks. Unlike the RBF-DDA algorithm, the counter-propagation network requires a definite number of processing elements in the hidden layer to begin learning. This was the first aspect of the network that was investigated. The results of the tests are documented in Appendix F.



**Figure 6. 49:** Results from investigation into optimum number of processing elements of the hidden layer of the counter-propagation network.

The u10snfe/r file was used for training and testing the network. Each legend represents a new network that was initialised and trained for 100 cycles with parameters of  $\alpha=0.1$  and  $\beta=0.1$  (Preliminary tests showed that these parameters yielded optimum results.). Each test was performed three times and the best of three results is depicted in the graph. It is clear from Figure 6.49 that the true positive recognition rate is “proportional” to the number of elements in the hidden layer. This result confirms *Kulkarni*’s[27] observation that it is necessary for the number of elements in the hidden layer be the same as the number of patterns presented to it, for the counter-propagation

network to perform efficiently. Further investigations were undertaken with datasets that yielded favourable results in the previous section..

**Table 6. 7:** Results of tests performed on normalised and unnormalised data for counter-propagation network.

Training set	Testing set	Number of cycles	$\alpha$	$\beta$	Rec. rate	Min. O/P	Training Time
u10snfe	u10snfr	200	0.1	0.1	55 - 88.043	0.94743	64 mins.
u8snfe	u8snfr	200	0.1	0.1	55 - 88.043	0.95333	40 mins.
uxtne	uxtnefr	200	0.1	0.1	53 - 88.478	0.95333	58 mins.
n10snfe	n10snfr	200	0.1	0.1	55 - 88.043	0.95407	62 mins.
n8snfe	n8snfr	200	0.1	0.1	55 - 88.043	0.95407	41 mins.
nxtnfe	Nxtnfr	200	0.1	0.1	53 - 88.478	0.95407	54 mins.

For tests performed on normalised and unnormalised data with the full dataset, best results were obtained when the remaining views were used for training and the  $\frac{3}{4}$  views were used for testing. Similar results were obtained for normalised and unnormalised data.

**Table 6. 8:** Results of tests performed on normalised and unnormalised data using split databases for counter-propagation network.

Training set	Testing set	Number of cycles	$\alpha$	$\beta$	Rec. rate	Min. O/P	Training Time
1uxtne	1uxtnefr	200	0.1	0.1	14 - 94.3	0.91678	17 mins.
	2uxtne				206 - 4.2	0.91678	
	2uxtnefr				207 - 3.7	0.91678	
1uxtnefr	1uxtne	200	0.1	0.1	14 - 94.3	0.91678	18 mins.
	2uxtne				208 - 3.3	0.91678	
	2uxtnefr				210 - 2.3	0.91678	
2uxtne	2uxtnefr	200	0.1	0.1	26 - 87.9	0.91902	15 mins.
	1uxtne				235 - 4.1	0.91902	
	1uxtnefr				235 - 4.1	0.91902	

2uxtnfr	2uxtnfe	200	0.1	0.1	21 – 90.2	0.91902	16 mins.
	luxtnfe				238 – 2.857	0.91902	
	luxtnfr				240 – 2.041	0.91902	

Training sets of luxtnfr and 2uxtnfr yielded recognition results of 94.3% and 90.2%. The results yielded by the radial basis function network were 94.7% and 83.7% respectively (see Table 6.5).

The minimum output that was classified as a true positive, is higher for networks trained with the counter-propagation algorithm. The results of the counter-propagation and radial basis function network will be compared in the following chapter.

It is noticed that results obtained from networks that were trained on remaining views and tested on three-quarter views performed consistently better than those trained on three-quarter views and tested on remaining views. The reason for this, from a psychophysical perspective, is discussed Chapter 2. From a neural network point of view, the remaining views allow for greater generalisation during the training phase. This allows the three-quarter view (the view that allows better encoding and recognition[24]) to be recognised faster and with greater accuracy.

## 6.6 Network per person approach (NPPA)

A three-layer back-propagation network of SNNS [13] was used in the network per person approach. These tests were performed to examine the viability of this approach compared to the database approach. Experiments were performed using single and double hidden layers in the network. It was found that although the network with single layers was able to train faster, better performance was obtained with a network that contained two hidden layers. Hence a network with two hidden layers was chosen. The network consists of 99 inputs, two hidden layers that contain 10x10 processing elements each and a single output unit.

A learning rate of  $\alpha=0.1$  and a momentum of  $\mu=0.5$  was used (these values are suggested by the authors of SNNS). The flat spot elimination was set to 0.1, while the maximum tolerable error was set to zero. 81 networks were trained separately using this algorithm. Each network was trained for 50 cycles. Earlier experimental work showed that this number of training cycles prevents over-training (for these particular data sets), as well as allows for optimum performance of the network. All the networks were trained on a Pentium™ 133MHz with 32MB of RAM, on SNNS, which used a Linux platform. Training time was typically 20 minutes per network.

The training and testing files were generated as follows:

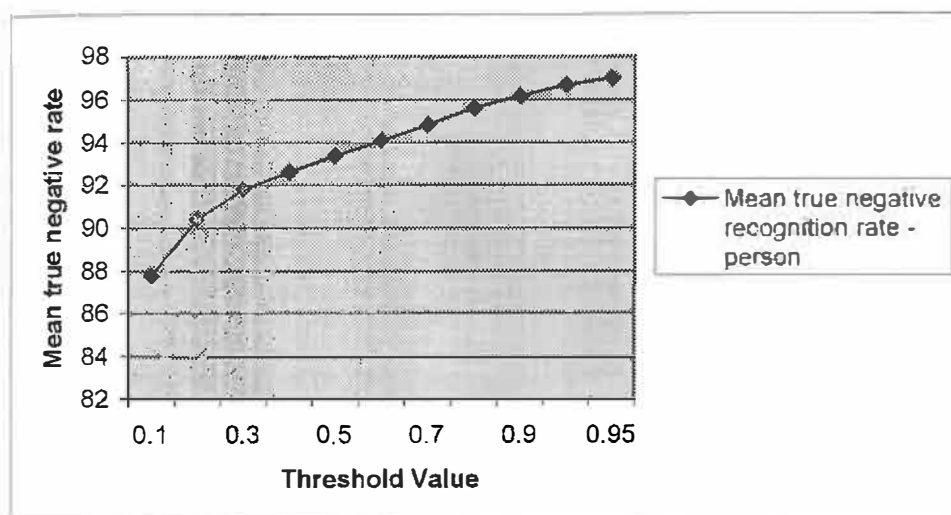
The training file consisted of 22 subjects (of 83 – although only 81 networks were trained for 81 distinct subjects, the testing file contained 83 distinct subjects), of which 1 of the 22 was a true positive, and the rest were true negatives. In the training file, for each of the 22 subjects, 5 views of the ten captured were used. *Lawrence*[41] used the same number of faces (5) in his training and testing sets. The testing file contained all 83 subjects with the other 5 views that were not contained in the training set. Hence, there was no overlap between the training set and the testing set. Also, there was unseen data (61 distinct subjects) which was used to determine the robustness of the network.

Some subjects wore glasses, while others wore caps (see Appendices B and E). Images of the relevant valid subjects with and without spectacles and/or caps were used in the training set.

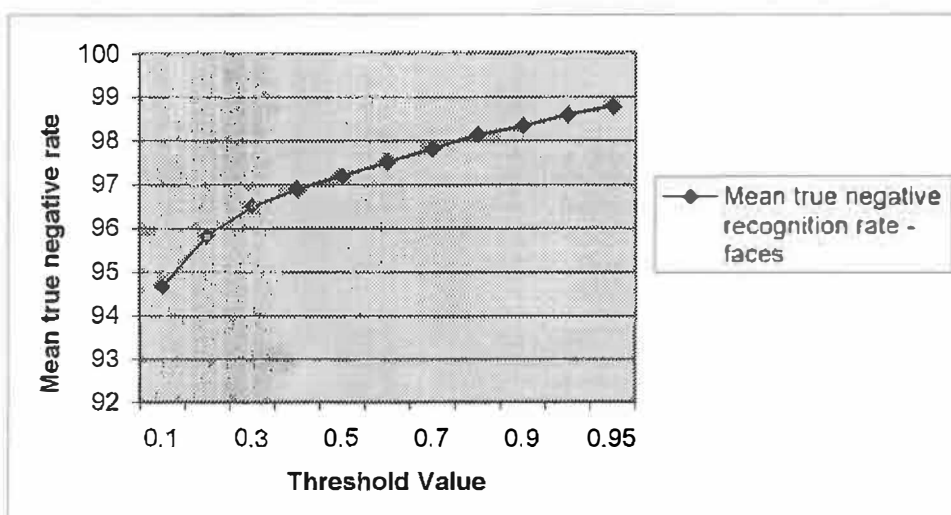
The choice of the 21 true negative subjects was not based on the variance distribution of the 2-D DCT images of the subjects. They were chosen at random from the database. These 21 subjects were included as true negatives in every one of the 81 training files that were generated, unless they were a true positive (i.e. the network was being trained for them). It was also arbitrarily decided that approximately 25% of the database would be used in the training sets. To correctly determine the optimum number of true negatives in the training set would require the number of true negative to be varied from 1 to, say 80. In effect,  $81^2$  networks would have to be trained. In addition to this, there are other variables to consider: The training parameters that work well under a particular topology and training set may not yield optimum performance when trained using another training set. It was decided not to perform such an analysis, due to its time consuming nature. It is felt that the choice of 25% of the database is not too conservative, neither is it over-zealous with respect to the distinct number of subjects being presented to the neural network during the training phase.

The output of the neural network ranged from 0 to 1. Analysis of the data was done by setting various thresholds of 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 0.93 and 0.95 to the output neuron. Appendix E contains tables detailing the results for thresholds of 0.8 to 0.95 for 81 networks.

Appendix E also discusses the formula that was used to calculate the true positive and true negative recognition rates.



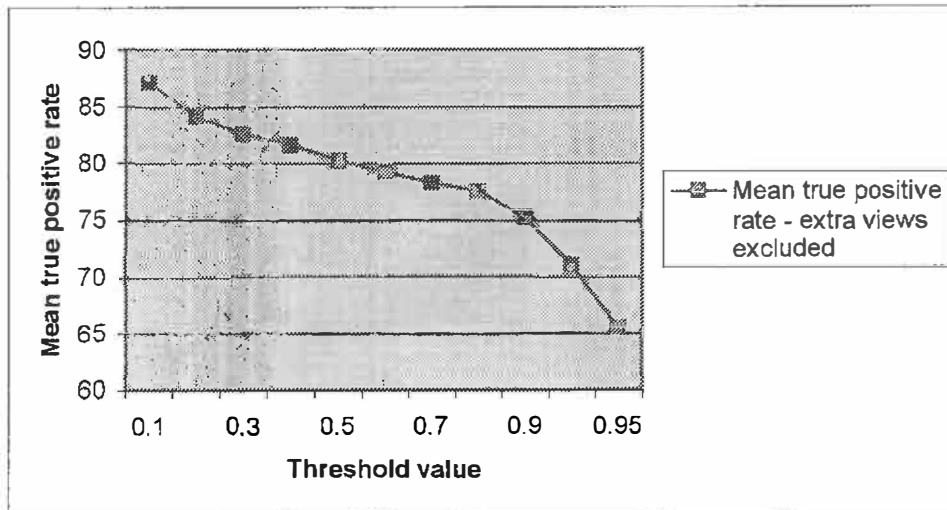
(a)



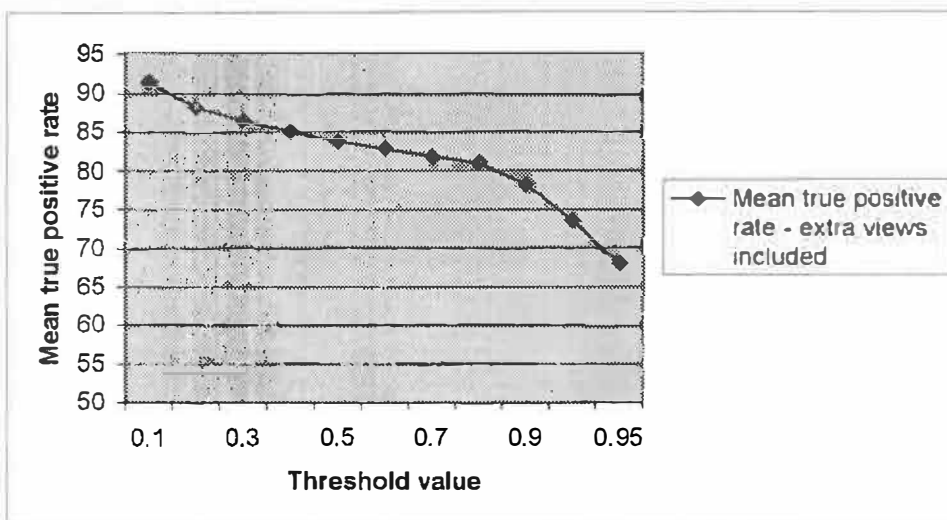
(b)

**Figure 6. 50:** (a) Mean true negative rate for different thresholds, calculated using "person statistics" (See Appendix E)

(b) Mean true negative rate for different thresholds, calculated using "face statistics" (See Appendix E)



(a)



(b)

**Figure 6. 51:** (a) Mean true positive rate for different thresholds, excluding extra views.  
(See Appendix E)

(b) Mean true negative rate for different thresholds, including extra views.  
(See Appendix E)

Figure 6.50 reports on two different mean true negative recognition rates (MTNRR): “person” and “face”. It is important to define each of these. For “person”, analysis is

performed on the output of the neural network, for all the subjects (other than the valid subject i.e. the one for whom the network was trained) presented to the input of the neural network during the testing phase. If the output for any one of the presented subject's views lies above a particular threshold, it is regarded as if all five of the subject's views were incorrectly classified as a true positive. This could be considered harsh, however it is felt that it is a better reflection of a real world scenario, and at the same time gives the correct number of distinct subjects incorrectly classified.

For "faces" (i.e. Figure 6.50 (b)), the calculation is based on the total number of faces (465 -- in the testing set) in the database, and not on distinct subjects. The threshold technique described above was applied here as well. It is expected that the MTNRR for faces is higher than for persons, as shown in Figure 6.50.

From Figure 6.50 (a), it can be seen that high true negative recognition rates are obtained for low thresholds. The high values associated with the graph of Figure 6.50 (a) suggests that the neural networks were able to generalise well (61 distinct subjects i.e. 305 views were unseen and contained in the testing set). It also suggests that an appropriate number of true negatives were included in the training set.

Figure 6.51 (a) and (b) also report on two types of mean true positive recognition rates (MTPRR). The first is based on a calculation that excludes "extra views" i.e. views of subjects with glasses etc, and Figure 6.51 (b) is based on a calculation that includes these views. The second approach could be interpreted as using a "skew" test set analysis (since not all the subjects wear glasses etc). The author does not share this view and both figures are included for completeness. The MTPRR is clearly inversely proportional to the MTNRR and the two figures can be used to choose an appropriate threshold value.

The view held by *Bouwer*[6] is that in security applications, it is preferred to produce a greater ratio of false negatives to false positives, than vice versa. False negatives,



although annoying to valid subjects, are safer than false positives, where an invalid subject may be admitted. Most face recognition research reports on only the true positive recognition rate that is obtained [1], [8], [11]. Equally important however is the true negative recognition rate (i.e. how well a network can reject an imposter).

The  $\text{mean}_1$  of the output neuron was 0.84 ( $\text{std}_1=0.30$ ) for true positives that excluded extra views and 0.80 ( $\text{std}_2 = 0.34$ ) for true positives that included extra views.

**Table 6. 9:** Table showing possible choices for thresholds.

Threshold	True Positive Excluding Extra Views	True Positive Including Extra Views	True Negative – Persons	True Negative – Faces
0.5 ( $\text{mean}_1 - \text{std}_1$ )	80.3	83.7	93.4	97.2
0.95 ( $\text{mean}_1 + \text{std}_1$ )	65.5	68.0	97.0	98.8
0.4 ( $\text{mean}_2 - \text{std}_2$ )	81.6	85.1	92.6	96.9
0.95 ( $\text{mean}_2 + \text{std}_2$ )	65.5	68.0	97.0	98.8

A plausible solution would be to use the first option in Table 6.9.

The networks are able to generalise quite well, as is evident from the high true negative recognition rates.

The disadvantages of the NPPA however, are that

- A network has to be trained for every person.
- Changes in subject's data (scars on face etc) implies that networks will have to be retrained.

The advantages of the NPPA are that

- High true negative recognition rates are obtained
- Relatively high true positive rates are obtained

## Chapter 7

# CONCLUSION AND RECOMMENDATIONS

---

### 7.1 Conclusion

This thesis examined the implementation of a two-dimensional discrete cosine transform and various different neural networks in an automatic face recognition system. The two-dimensional transform was implemented using the fast Fourier transform. It was used because of its ability to de-correlate the data from the images and easily facilitate feature extraction. The neural networks were used for the classification of the faces. Although neural networks have been used in a classification mode, they are not restricted to this scope. They have been used in problems as diverse as credit scoring to prediction[10].

Chapter 2 shows that it is indeed difficult to compare various research efforts since different researchers use different face databases. Consequently, an algorithm that performed well on a particular database is not guaranteed to have the same effect on another database. This unfortunate result was re-iterated by the work of *Zhang*[40].

Psychophysical research into the mechanisms of human face recognition has influenced much work in the field of machine recognition of faces [8], [30]. A finding that directly influenced this study was the discovery of the  $\frac{3}{4}$  view advantage by *Perret*[32], which was confirmed, using psychophysical methods by *Bruce*[20] and *Schyns*[24]. They reported that the  $\frac{3}{4}$  view holds an advantage in face recognition to previously unseen views. It was with this in mind that a face capturing utility (a system component of the

face recognition algorithm) was developed, for the building of a database that exploited the  $\frac{3}{4}$  view advantage. Ten views per subject were captured at different head orientations. The entire database consists of 820 facial images (i.e. 82 distinct subjects).

A general symmetry transform (GST) was implemented as a means to automatic face location in images containing a face with a homogenous background. The transform, as described by *Reisfeld*[15] (and used with great effect in his PhD thesis) is able to locate areas of high symmetry in an image (isotropic, as well as radial symmetry). This is done by assigning to each pixel a symmetry magnitude (see equation 4-7). Images of size 215x215 were presented to the transform. The transform was found to be accurate in assigning high symmetry to the eye and eyebrow regions. (See Figure 4.5 and Table 4.1). However, the inherent problem with the GST is the computational intensity demanded due the nature of its operation (i.e. assigning a magnitude of symmetry to each pixel). Images of size 215x215 took approximately 113 minutes to complete on a Pentium 133MHz computer running the Linux operating system. A speed enhancement study was performed on the GST by first sub-sampling the images to sizes of 100x100 and 50x50 pixels respectively. The transform took approximately 70 and 5 minutes respectively. Although the transform performed faster, degradation of the transformed images was clearly noted. Observation of the GST showed (see equation 4.6) that the transform can be speeded up by thresholding the input. An image of size 100x100 pixels took 30 minutes to be transformed. In view of the complexity of the GST, manual normalisation was used and was implemented much faster than the GST.

The two- dimensional discrete cosine transform that was used employed a fast algorithm by first transforming the rows of the image, (thus forming a semi-transformed matrix), and then the columns of the semi-transformed matrix. It was shown in Figures 4.9 – 4.15 that data with high energy in the transformed matrix was concentrated in the upper left hand side of the transformed matrix near the dc or mean grey level of the image. A variance analysis of each coefficient of the transformed image was performed for each of the subjects. This investigation revealed intra-class variance clusters (see Figures 4.22 – 4.24). It was shown that these clusters were

grouped approximately in a 10x10 square region. However, since in a study such as this one, inter-class variances are more important, a variance analysis was performed on each coefficient of every transformed image in the database. Visual inspection revealed that the data with the most variance was situated in a 10x10 region (see Figure 4.25). This was used as a starting point for data extraction. A performance analysis of different data sets (i.e. different sizes of data as well as different regions (square and triangular – see Figures 6-1 and 6.2 from which the data was extracted) was performed using the radial basis function network with dynamic decay adjustment. The results of the investigation revealed that the data that was extracted from a 10x10 square region of the transformed images yielded the best performance (88%), while the 8x8 and 12x12 regions followed with recognition rates of 87.4% and 87.4% respectively. For data that was extracted from a triangular region, the highest recognition rate was obtained when the length of one side of the isosceles triangle was 10 coefficients long. These results were used to choose data (i.e. size of data and region) for further investigations. The 8x8 and 10x10 square region data was used, as well as 10x10 triangular region data.

For the radial basis function network, particular values of  $\theta^+$  and  $\theta^-$  were used during training that yielded best performance. These values can be found in Table 6.4. Full databases as well as split databases were investigated. In both types of investigation, networks that were trained on remaining views and tested on  $\frac{3}{4}$  views yielded high recognition rates. Networks generally didn't perform better when "mixed views" were used in the data set. Hence, the generalisation ability of the network was not improved. For full databases, it was found that triangular region data performed almost as well as the square region data, however, 10x10, square region data produced a recognition rate of 88.3% (trained on u10snfe and tested on u10snfr). Perhaps one of the most interesting aspects of the results obtained with split databases is the low false positive rate obtained. This demonstrates the networks ability to reject impostors.

The results from tests performed with normalised full databases were not as good as those performed with unnormalised data. The highest classification rate obtained was

86.7% when the network was trained on normalised data extracted from a triangular region containing remaining views and tested on  $\frac{3}{4}$  views. Tests done on split databases and normalised data revealed that data extracted from the triangular region performed the best (i.e. trained on files 1nxtnfe and 2nxtnfe yielded true positive recognition rates of 90.2% and 86.5% respectively).

Datasets that produced high recognition rates were used in investigations with the counter-propagation network. Comparative results for the radial basis function network and counter-propagation network (for tests done with full databases i.e. 82 subjects) can be tabulated as follows:

**Table 7. 1:** Comparative results of Counter-propagation network and RBFN with DDA.

Train	Test	Counter-propagation Classification rate (%)	RBFN –DDA Classification rate (%)
u10snfe	u10snfr	88.0	88.3
u8snfe	u8snfr	88.0	87.6
uxtne	uxtnfr	88.5	87.8
n10snfe	n10snfr	88.0	85.0
n8snfe	n8snfr	88.5	85.0
nxtne	nxtnfr	88.5	86.7

Counter-propagation networks take longer to learn compared to radial basis function networks trained with the dynamic decay adjustment algorithm. Radial basis function networks take typically < 5 epochs to train (i.e. < 3 minutes) as opposed to counter-propagation networks that could take as long as 64 minutes to train. Unnormalised data produced better results than normalised data in investigations with the counter-propagation network as well.

**Table 7. 2:** Comparative results of Counter-propagation Network and Radial Basis Function Network trained with DDA, for split databases (i.e. 41 subjects).

Train	Test	Counter- propagation Classification rate (%)	RBFN Classification rate (%)
1nxtnfe	1uxtfnr	94.3	93.5
	2uxtfnr	3.7	3.7
	2nxtnfe	4.2	4.2
<b>1nxtfnr</b>	<b>1nxtnfe</b>	<b>94.3</b>	<b>94.7</b>
	<b>2nxtfnr</b>	<b>3.3</b>	<b>3.7</b>
	<b>2nxtnfe</b>	<b>2.3</b>	<b>2.8</b>
2nxtnfe	2nxtfnr	88.0	87.0
	1nxtnfe	4.1	2.4
	1nxtfnr	4.1	2.9
<b>2nxtfnr</b>	<b>2nxtnfe</b>	<b>90.2</b>	<b>83.7</b>
	<b>1nxtnfe</b>	<b>2.9</b>	<b>1.2</b>
	<b>1nxtfnr</b>	<b>2.0</b>	<b>2.4</b>

It is clear from Table 7.1 and 7.2 that the smaller databases yield higher true positive and true negative recognition rates. The optimum size for the “small” database was not determined, as this constitutes a complete study on its own. This could be a source for further work. For split databases the overall better performance was obtained from the counter-propagation network (see **bold** values of Table 7.2). The number of processing elements required by the RBFN is similar to the amount used by the counter-propagation network.

The back-propagation network that was used in a network per person approach (NPPA) produced high true negative and true positive recognition rates, for different thresholds. The results of the NPPA cannot be directly compared to those of the database approach (DA) (i.e. results of the counter-propagation and radial basis function networks) due to the diverse approaches used. Qualitatively, however, the NPPA compared to the DA requires more work. However, batch training can be used to overcome this. With the DA, if a subject changes, then the whole database has to be re-trained, as opposed to the NPPA where only 1 network has to be trained. The

to overcome this. With the DA, if a subject changes, then the whole database has to be re-trained, as opposed to the NPPA where only 1 network has to be trained. The NPPA would work well in high security environment with a set number of distinct subjects. This recommendation is made on the basis of the high true negative and true positive recognition rates. The DA would work well with a vast number of valid subjects.

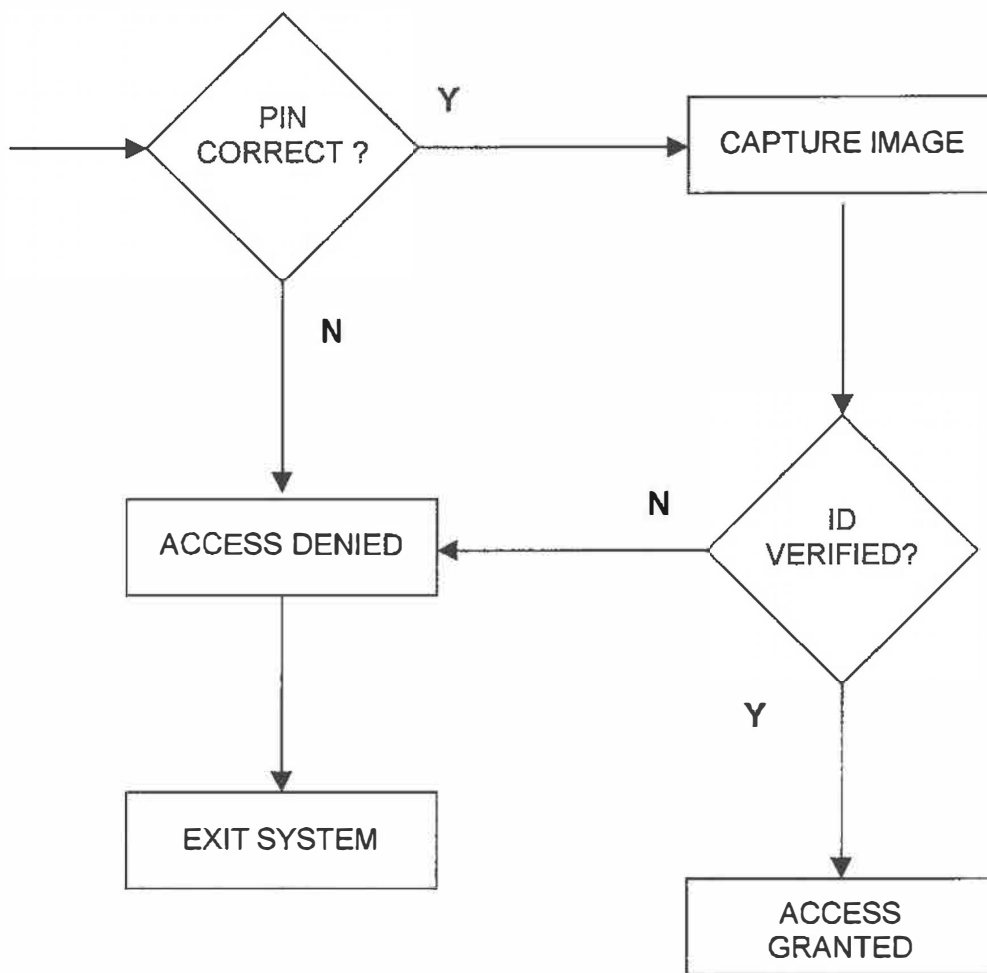
## 7.2 Recommendations

The face recognition algorithm (FRA) described in the main body of this thesis would work well in an environment where a subject's identity needs to be verified. Assume that the algorithm will be implemented in a security system where the first level of access would be the subject's PIN (e.g. ATM application). Hence, in such a system, the subject's PIN would be used in addition to the FRA for subject verification. The enhanced security system could be described in flow diagram form as depicted in Figure 7.1.

Before describing how the FRA would be implemented into the ATM security system, it is necessary to define the type of card that the subject would use, as well as its implications. The FRA uses neural networks as the core verification principle, and hence it is necessary to store the artificial neural network (ANN) weights of the valid subjects somewhere.

One option is to store the weights on a smart card/ATM card. Thus, the weights could be read, passed to a neural network (preferably processing takes place at the ATM), and verification can take place. This type of implementation would utilise the "neural network per person" approach. The neural network is trained to verify the valid subject (i.e. the owner of the ATM card) and reject all impostors. Although the high true negative and high true positive recognition rates associated with this approach are quite appealing, the disadvantage is that a separate network has to be trained for each subject. This may not be viable if the number of valid subjects is large. However, it could be accomplished if "batch training" (a facility offered by SNNS) was performed. The flow diagram, which describes this approach, is depicted in Figure 7.2.





**Figure 7. 1:** High-level flow diagram of FRA integrated with level-one PIN access security system.

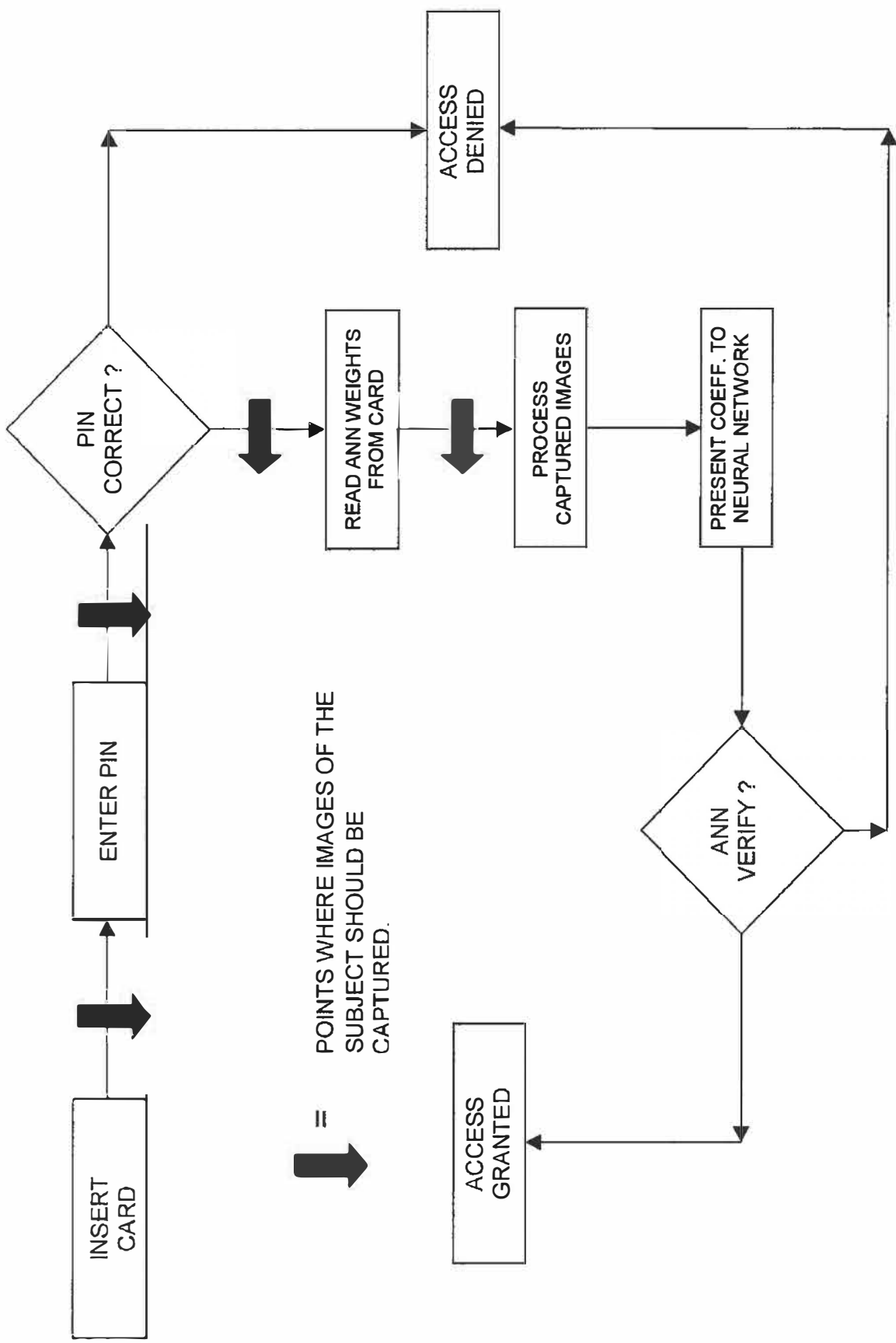
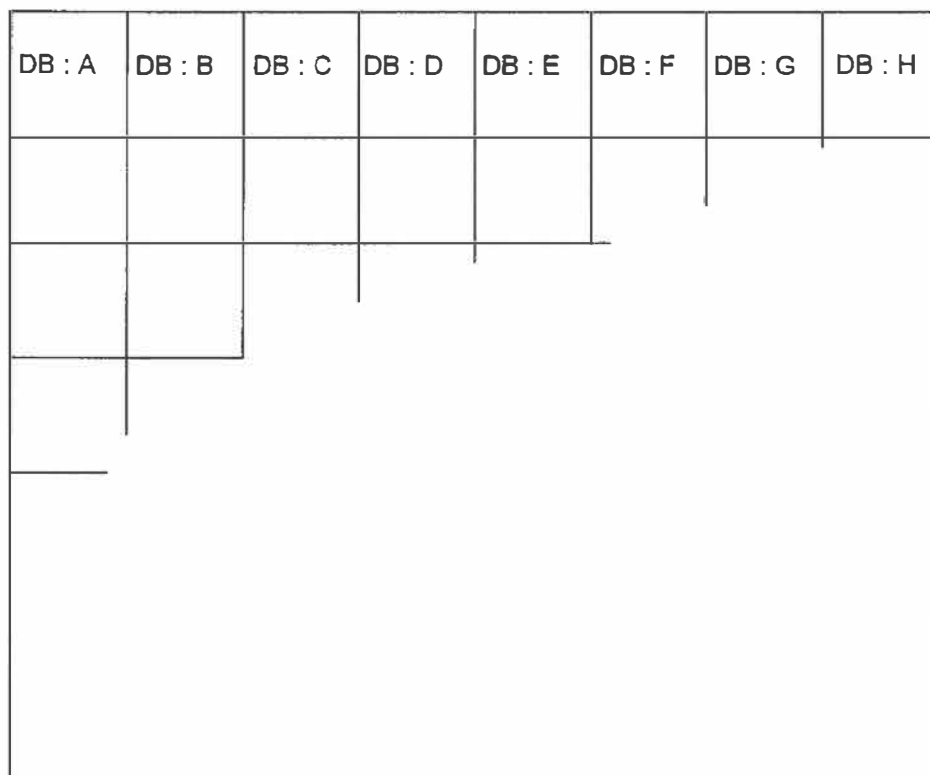


Figure 7. 2: High-level flow diagram of security system using smart card to store weights of the ANN.

The second option is to store the subject's weights in a database of weights, on some remote server. The valid subject's ANN weights can be accessed from the database via the subject's PIN. The flow diagram that describes this approach is much like that of Figure 7.2.

The large number of valid subjects to be trained can be exploited to the advantage of the security system. It was observed for split databases in Chapter 6 (i.e. training on part of database 1 and testing on the remainder of database 1 and all of database 2), that the “unseen” database produced high true negative recognition rates.

The large training database could be split into smaller databases as follows:



**Figure 7. 3: Depiction of small databases within one large database.**

The valid subjects from each database can only be recognised in their database i.e. if subject  $n$  belongs to database A, then subject  $n$  would be verified correctly in database A

and not (within some acceptable error) in any of the other databases. However, the optimum number of subjects in each database requires some investigation, and was not completely studied in this research due to subject availability. This method of training can be likened to the scope of variables in a function. The variables can be seen within the function, but not outside the function.

The general symmetry transform was proposed as a possible solution for automatic face location, but it was found that it was computationally intensive. This provides a source for further work to be done in this particular field. Face location deserves a study dedicated to it alone. It is easy for humans to be able to take into account varying light levels, rotation, various backgrounds etc, and still perform effective face recognition. However it is difficult to reproduce this process on a computer.

This thesis marks an initial study into automatic face recognition. It was shown the two dimensional discrete cosine transforms and neural networks do form a viable solution to the problem of automatic face recognition. However, it is not the optimal solution. With the advent of faster and more powerful processors techniques such as elastic graph matching and hidden markov models a more realistic automatic face recognition algorithm could be implemented.

## References

---

- [1] M.K. Fleming and G.W. Cottrell, "Categorisation of Faces Using Unsupervised Feature Extraction", *Proceedings of International Joint Conference on Neural Networks*, Vol. II, pgs. 65-70, 1990.
- [2] R. Brunelli and T. Poggio, "Face Recognition: Features versus Templates", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15 Part 10, pgs. 1042-1062, 1993.
- [3] R. Brunelli and T. Poggio, "Face Recognition Through Geometrical Features", *Proceedings of 2<sup>nd</sup> European Conference on Computer Vision*, pgs. 972-800, Santa Margherita Ligure, Italy, May 1992.
- [4] N. Intrator, D. Reisfeld and Y. Yeshurun, "Face Recognition using a Hybrid Supervised/Unsupervised Neural Network", *Department of Computer Science*, Tel Aviv University, June 22, 1995.
- [5] A. Samal and P. A. Iyengar, "Automatic Recognition and Analysis of Human Faces and Facial Expressions: A Survey", *Pattern Recognition*, Volume 25 Part 1, pgs. 65-77, 1992.
- [6] P. Bouwer and A.D. Broadhurst, "An Approach to Face Recognition Using Gray-Scale Transforms and Neural Networks", *Proceedings of COMSIG 95*, Pretoria, 1995.
- [7] D. Resifeld, "Generalised Symmetry Transforms: Attentional Mechanisms and Face Recognition", *PhD Thesis*, Tel Aviv University, January 1994.

- [8] F. Samaria, "Face Recognition using Hidden Markov Models", *PhD Thesis*, University of Cambridge, 1994.
- [9] J. Wilder, "Face Recognition Using Transform Coding of Gray Scale Projections and the Neural Tree Network", *Artificial Neural Networks for speech and Vision – Edited by R.J. Mammone*, Chapman and Hall Neural Computing Services, London, 1993.
- [10] R. Hecht-Nielsen, "Neurocomputing", *Addison-Wesley Publishing Company*, Reading, Massachusetts, 1987.
- [11] R. Hecht-Nielsen, "Applications of Counterpropagation Networks", *Neural Networks*, Vol. 1 Part 2, 1988.
- [12] A. Zell et al, "SNNS User Manual Version 4.1", *Institute for Parallel and Distributed High-Performance Systems – University of Stuttgart*, Version 4.1, 1996.
- [13] T. Kohonen, "The Self-Organising Map", *Proceedings of the IEEE*, Vol. Part 9, pgs. 1460-1484, September 1990.
- [14] N. Intrator, D. Reisfeld, Y. Yeshurun, "Extraction of Facial Features for Recognition using Neural Networks", *Department of Computer Science*, Tel Aviv University, June 1995.
- [15] D. Reisfeld, H. Wolfson and Y. Yeshurun, "Context Free Attentional Operators: The Generalised Symmetry Transform", *Department of Computer Science*, Tel Aviv University, March 1995.
- [16] N. Intrator, D. Reisfeld, Y. Yeshurun, "Face Recognition Using a Hybrid Supervised/Unsupervised Neural Network", *Department of Computer Science*, Tel Aviv University, June 22 1995.

- [17] K. R. Rao and P. Yip, "Discrete Cosine Transform : Algorithm, Advantages, Applications", *Academic Press Inc. Oval Road London*, 1990
- [18] R.J. Schalkoff, "Digital Image Processing and Computer Vision", *John Wiley and Sons*, New York, 1989.
- [19] P.M. Embree and B. Kimble, "C Language Algorithms for Digital Signal Processing", *Prentice Hall Ptr, Up Saddle River, New Jersey*, 1991.
- [20] V. Bruce, T. Valentine and A. Baddeley, "The Basis of the  $\frac{3}{4}$  View Advantage in Face Recognition", *Applied Cognitive Psychology*, Vol. 1, pgs 109-120, 1987.
- [21] C. G. Looney, "Pattern Recognition using Neural Networks – theory and algorithms for scientists and engineers", *Oxford University Press*, Oxford, 1997.
- [22] N. Ahmed, T. Natarajan, and K. Rao, "Discrete Cosine Transform", *IEEE Transactions on Computers*, vol. 1, pgs 90-93, 1974.
- [23] J. Canny, "A Computation Approach to Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8 Part 6, November 1986.
- [24] P. G. Schyns and H.H. Bulthoff, "Conditions for Viewpoint Dependent Face Recognition", *AI Memo No.1432, CBCL Paper No. 81*, Massachusetts Institute of Technology – Artificial Intelligence Laboratory and Centre for Biological and Computational Learning, August 1993.
- [25] D. I. Perret et al, "Visual cells in the Temporal Cortex Sensitive to Face View and Gaze Direction", *Proceedings of the Royal Society of London*, pgs. 293-317, 1985.
- [26] A. K. Jain, "Fundamentals of digital image processing", Englewood Cliffs, N.J., Prentice-Hall, 1989.

- [27] A. D. Kulkarni, "Artificial neural networks for image understanding", New York, Van Nostrand Reinhold, 1994.
- [28] B. Krose and P. Van Der Smagt, "An Introduction to Neural Networks", 8<sup>th</sup> Edition, University of Amsterdam, Kruislaan 403, NL-1098 SJ Amsterdam, November 1996.
- [29] M. Golfarelli, D. Mario and D. Maltoni, "On the Error-Reject Trade-Off in Biometric Verification Systems", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19 Part 7, July 1977.
- [30] R. Chellapa et al, "Human and Machine Recognition of Faces: A Survey", Computer Vision Laboratory-Centre for Automation Research, University of Maryland – College Park, MD 20742-3275, May 1995.
- [31] L. D. Harmon, "The Recognition of Faces", *Scientific American*, Vol. 229 Part 5, pgs 71-82, 1973.
- [32] D. I. Perret, A.J. Milstin, and A.J. Chitty, "Visual Neurones Responsive to Faces.", *Trends in Neurosciences*, Vol. 10, pgs. 358-364, 1989.
- [33] K.K. Sung and T. Poggio, "Example-based Learning for View-based Human Face Detection", *AI Memo No.1512, CBCL Paper No. 112*, Massachussets Institute of Technology – Artificial Intelligence Laboratory and Centre for Biological and Computational Learning, December 1994.
- [34] I. Craw, H. Ellis and J. Lishman, "Automatic Extraction of Face Features", *Pattern Recognition Letters*, Vol. 5, pgs. 183-187, 1987.



- [35] V. Govindaraju, S.N Srihari and D.B. Sher, "A Computational Model for Face Location", *Proceedings, Third International Conference on Computer Vision*, pgs 718-721, 1990.
- [36] S. Baluja, H.A. Rowley and T. Kanade, "Human Face Detection in Visual Scenes", *CMU-CS-95-158*, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213.
- [37] M. Kokuer, "Automatic Facial Feature Recognition and Location", Anadolu University , Eskisehir, Turkey, November 1992.
- [38] T. Kanade, "Picture Processing by Computer Complex and Recognition of Human Faces", *Technica report*, Kyoto University, Department of Information Science, 1973.
- [39] W. Konen and E. Schulze-Kruger, "ZN-Face: A system for Access Control Using Automated Face Recognition", *International Workshop on Automatic Face and Gesture Recognition*, University of Zurich, 1995.
- [40] J. Zhang, Y. Yan and M. Lades, "Face Recognition: Eigenface, Elastic Matching and Neural Nets", *Proceedings of the IEEE – Special Issue on Biometrics*, Vol. 85 Part 9, September 1997.
- [41] S. Lawrence et al, "Face Recognition: A Convolutional Neural Network Approach", *IEEE Transactions on Neural Networks – Special Issue on Neural Networks and Pattern Recognition*, 1997  
<http://www.neci.nj.nec.com/homepages/lawrence/papers/face-tnn97/l2h/index.htm>
- [42] M.R. Berthold and J. Diamond, "Boosting the performance of RBF networks with dynamic decay adjustment", *Advances in Neural Information Processing Systems*, Vol. 7, 1995.

- [43] S.C. Debipersad and A.D. Broadhurst , "Face Recognition using Neural Networks", *Proceedings of Comsig '97*, Grahamstown, 1997.

## APPENDIX A

---

```
/*

    SANJEEV C DEBIPERSAD

    FACE CAPTURING UTILITY FOR IMAGES OF SIZE 215X255
    VER. 2

*/

/* System include files */
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys\types.h>
#include <sys\stat.h>
#include <io.h>

/* DT2867 related include files */
#include "c:\dti\include\ioctl.h"
#include "c:\dti\include\dt2867\lib67.h"
#include "c:\dti\dt2867\examples\exlib.h"
#include "c:\dti\include\ioctext.h"
#include "c:\dti\include\dtityp.h"

/***** Type definitions *****/

/* This structure contains information that varies depending */
/* upon the field rate of the incoming video signal [NTSC */
/* (60 Hz) vs. PAL (50Hz)]. */
typedef struct buffer_size_struct
{
    int width;           /* displayable image width */
    int height;          /* displayable image height */
} Buf_size;

/***** Macro Definitions *****/

/* define the image width and height for 50/60Hz images */
```

```

#define NTSC_WID  640
#define NTSC_HGT  480
#define PAL_WID   768
#define PAL_HGT   512

#define LUT_SIZE  256
#define FILL_VAL  128
#define WIN_COLOR 15

#define ENTER(s)  printf("Hit <Enter> %s:  ", (s)),  gets(answer)

/***** Static Variables *****/

static char *usage = "USAGE:  exlc  <device_name>\n\n"
                    "\twhere <device_name> is the name of the device to
be opened\n"
                    "\t(eg. dt2867$0).\n\n";

static char answer[133];          /* buffer used to read  */
static int status = E_NORMAL;     /* user responses       */
static u_long lut[256] = {0};     /* status variable used */
                                /* by error handler    */
static u_long lut[256] = {0};     /* used for lookup      */
static Buf_size ntsc_data = { NTSC_WID, NTSC_HGT };
static Buf_size pal_data = { PAL_WID, PAL_HGT };

/***** LIB67 Error Handler *****/

static void my_handler(const int err, char const * const func, char
const * const msg)
{
    /* Save the error in the static status variable */
    status = err;

    /* Now, simply print the error and return */
    printf("\aError %d:  %s\n\tin function %s\n\n", err, msg, func);
    return;
}

/***** Memory Cleanup Function *****/

/* Define a simple static function to perform memory cleanup */
static void cleanup(u_long *p1, u_long *p2)
{
    if (p1)          free (p1);
    if (p2)          free (p2);

    return;
}

/***** Main Program *****/

```

```

int main (int argc, char **argv)
{
    register int i = 0;           /* fast index          */
    int fd = -1;                 /* device file         */
                                /* descriptor          */
    DT67_input input_buf;        /* LIB67 input         */
                                /* structure           */
    Config_struct config;         /* used to get DT2867  */
                                /* base configuration  */
    Buf_size *buf_size = &ntsc_data; /* pointer to signal   */
                                /* dependent data      */
    u_long red = 0, green = 0, blue = 0; /* saved rgb          */
                                /* values for          */
                                /* output and          */
                                /* overlay LUTs        */
    u_long r = 0, g = 0, b = 0; /* rgb values for      */
                                /* output and overlay  */
                                /* LUTs                */
    u_long *lut_save = calloc(LUT_SIZE, sizeof(u_long));
                                /* pointer to saved input lookup table values */

    u_long *lut_array = calloc(LUT_SIZE, sizeof(u_long));
                                /* pointer to input lookup table values for fill */
                                /* function */

    XY_rgn window = {0};
    XY_rgn_buf rgn;
    long count;
    unsigned int len;
    unsigned char far *fptr;
    FILE *in;

    /* verify that the buffers were successfully allocated */
    if (!lut_save || !lut_array)
    {
        cleanup (lut_save, lut_array);
        return E_NOMEM;
    }

    /* Check for correct number of command line arguments */
    if (argc != 2)
    {
        printf(usage);
        cleanup (lut_save, lut_array);
        return 1;
    }

    /* Open channel to the device specified by the user */
    if ( (fd = open("dt2867$0", O_RDWR)) == -1 )
    {
        printf("\aUnable to open device:  %s\n", argv[1]);
        cleanup (lut_save, lut_array);
        return 1;
    }
}

```

```

/* Figure out if we are running a 50H or 60H device */
if ( status = ioctl(fd, DRV_CONFIG, &config) )
{
printf("\aUnable to get device configuration\n\tstatus = d\n\n",
status);
cleanup (lut_save, lut_array);
return status;
}
if ( config.hz == 50 )
buf_size = &pal_data;

/* Tell LIB67 to use our error handler */
(void) dt67_set_handler(my_handler);

/* Reset the DT2867 to the default values. This call */
/* shows a method */
/* for retrieving, and testing, the status from */
/* LIB67 functions. */

if ( (status = dt67_reset(fd)) != E_NORMAL )
{
cleanup (lut_save, lut_array);
return status;
}

/* Load input LUT 0 (8-bit), processing LUT 0 (16-bit), and */
/* the output LUT with the identity function (this shows */
/* another legal calling format). */

for (i = 0; i < 256; i++)
lut[i] = i;
if ( status = dt67_inp_lut(fd, DT67_WRITE, DT67_ILUT, DT67_LUT0,
0, 256, lut) )
{
cleanup (lut_save, lut_array);
return status;
}
if ( status = dt67_inp_lut(fd, DT67_WRITE, DT67_PLUT, DT67_LUT0,
0, 256, lut) )
{
cleanup (lut_save, lut_array);
return status;
}
if ( status = dt67_disp_lut(fd, DT67_WRITE, DT67_OLUT, 0, 256,
lut, lut, lut) )
{
cleanup (lut_save, lut_array);
return status;
}

/*===== Window A/D =====*/

```

```

/* Now turn on the display for buffer 0. */
if ( dt67_disp_sel(fd, DT67_A2D_OUTPUT) )
{
    cleanup (lut_save, lut_array);
    return status;
}
if ( dt67_disp(fd, DT67_ON) )
{
    cleanup (lut_save, lut_array);
    return status;
}
/* set overlay lookup table to blue for index specified by
WIN_COLOR */

/* read the contents of the overlay lookup table at index
WIN_COLOR */

if (dt67_disp_lut(fd, DT67_READ, DT67_OVLUT, WIN_COLOR, 1, &red,
&green, &blue))
{
    cleanup (lut_save, lut_array);
    return status;
}

/* write the contents of the overlay lookup table at index
WIN_COLOR to BLUE */
b = 255L;
r = g = 0L;
if (dt67_disp_lut(fd, DT67_WRITE, DT67_OVLUT, WIN_COLOR, 1, &r,
&g, &b))
{
    cleanup (lut_save, lut_array);
    return status;
}

/* perform window A/D (note: EXLIB has no error handling so */
/* we need to get the status via the return value). */

/* Define the window */
window.y = 150;
window.x = 250;
window.height = 215;
window.width = 255;
/* Perform the window A/D */
if ( status = exlc_window_a2d(fd, DT67_BUF1, WIN_COLOR, &window)

{
    printf("\aWindow A/D failed\n\tstatus = %d\n\t%s\n\n",
status, exlc_error_str(status));
    cleanup (lut_save, lut_array);
    return status;
}

/* Wait for the user to tell us to exit */
ENTER("to exit");

```

```

/*      dt67_reset(fd); */

/* Wait for the user to tell us to clear the frame */
ENTER("to clear frame buffer 0");

/* clear buffer 0 (note: EXLIB has no error handling so we need */
/*      to get the status via the return value). */

    if ( dt67_disp(fd, DT67_BUF0) )
    {
        cleanup (lut_save, lut_array);
        return status;
    }

    if ( status = exlc_frame_clear(fd, DT67_ACQ0) )
    {
        printf("\aFrame clear failed\n\tstatus = %d\n\t%s\n\n",
status, exlc_error_str(status));
        cleanup (lut_save, lut_array);
        return status;
    }
    if ( dt67_disp(fd, DT67_BUF0) )
    {
        cleanup (lut_save, lut_array);
        return status;
    }

    if ( dt67_report_input(fd, &input_buf) )
        return status;

/* First, set up to acquire a frame to buffer 0 */
input_buf.timing = DT67_EXTERNAL;
input_buf.acqenb = DT67_ACQ0;
if ( dt67_setup_input(fd, &input_buf) )
{
    cleanup (lut_save, lut_array);
    return status;
}

/* Acquire the frame */
if ( dt67_input(fd) )
{
    cleanup (lut_save, lut_array);
    return status;
}

/* Now turn on the display for buffer 0. */
if ( dt67_disp_sel(fd, DT67_BUF0) )
{
    cleanup (lut_save, lut_array);

```



```

        return status;
    }
    if ( dt67_disp(fd, DT67_ON) )
    {
        cleanup (lut_save, lut_array);
        return status;
    }

    dt67_io_sel(fd,DT67_BUF0);

    rgn.region.x=250;
    rgn.region.y=150;
    rgn.region.width=215;
    rgn.region.height=255;

    rgn.size=1;

    if((rgn.buf=calloc(rgn.region.width*rgn.region.height,1))==NULL)
    {
        printf("1. Could not allocate memory, exiting");
        exit(0);
    }

    if ( (status=ioctl(fd,GET_XY_RGN,&rgn)) != E_NORMAL) {
        exit(0);
    }

    in = fopen(argv[1],"wb");
    fptr=(char far *)rgn.buf;
    count = (long)rgn.region.width*rgn.region.height;

    while(count!=0){
        len = fwrite(fptr,1,count>32768?32768:count,in);
        count-=len;
        fptr+=len;
    }
    farfree(rgn.buf);
    fclose(in);

    /* All done, close the device and return normal status */
    close(fd);
    cleanup (lut_save, lut_array);
    return 0;
}

```

## APPENDIX B

This appendix is a graphical description of the database that was used in this study. Categorisation into groups is meant only for statistics.

Table B. 1 : Statistical description of database collected, and used.

	Indian Male	Indian Female	African Male	African Female	European Male	European Female	Coloured Male	Coloured Female
No. Of Subjects	16	29	6	6	9	10	2	4
Subjects with glasses	2	1	0	0	0	2	0	0
Subjects with hats	0	0	1	2	0	0	0	0
Occluded views	0	1	0	0	0	0	0	0
Alice Bands	0	0	0	0	0	1	0	0

Table B. 2 : Database of subjects 1-3















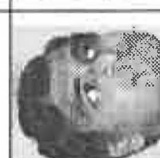
















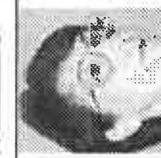
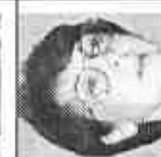

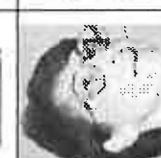





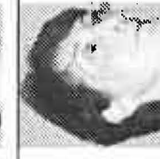



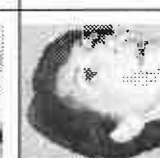





Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
1										
2										
2										
3										
3										

Table B. 3 : Database of subjects 4-8











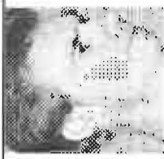
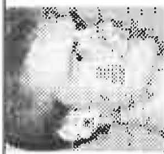
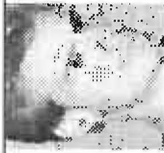





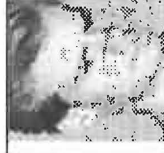




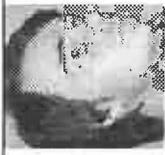









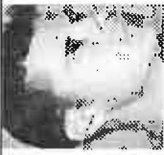








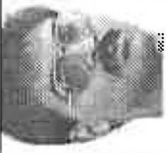







Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
4										
5										
6										
7										
8										

Table B. 4 : Database of subjects 8-10





Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
8										
9										
9										
10										
10										



Table B. 5 : Database of subjects 10-14



















































Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
10										
11										
12										
13										
14										

Table B. 6 : Database of subjects 15-18

















































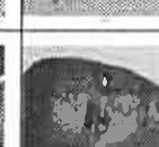

Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
15										
16										
17										
17										
18										

Table B. 7 : Database of subjects 19-23



















































Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
19										
20										
21										
22										
23										



Table B. 8 : Database of subjects 24-28



















































Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
24										
25										
26										
27										
28										

Table B. 9 : Database of subjects 29-33































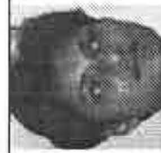
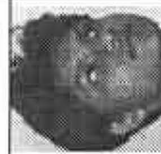








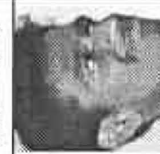

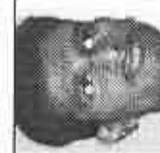




Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
29										
30										
31										
32										
33										

Table B. 10 : Database of subjects 34-37



















































Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
34										
35										
35										
36										
37										

Table B. 11 : Database of subjects 38-42





































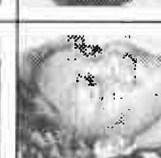













Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
38										
39										
40										
41										
42										



Table B. 12 : Database of subjects 43-47











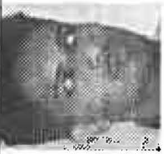
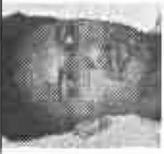






































Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
43										
44										
45										
46										
47										

Table B. 13 : Database of subjects 48-51













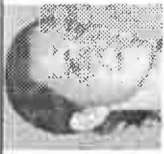




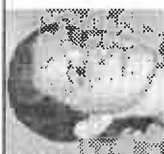
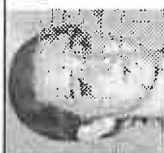































Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
48										
49										
50										
51										
52										

Table B. 14 : Database of subjects 53-56

































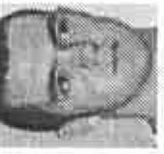

















Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
53										
53										
54										
55										
56										

Table B. 15 : Database of subjects 57-61













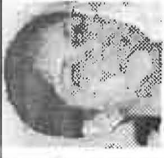
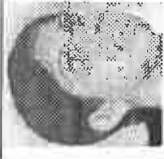




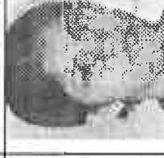































Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
57										
58										
59										
60										
61										



Table B. 16 : Database of subjects 62-65
































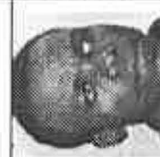






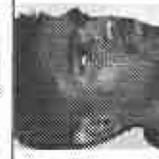







Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
62										
63										
64										
64										
65										

Table B. 17 : Database of subjects 66-70






Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
66										
67										
68										
69										
70										

Table B. 18 : Database of subjects 71-75










































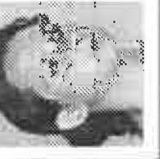








Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
71										
72										
73										
74										
75										



Table B. 19 : Database of subjects 76-80








































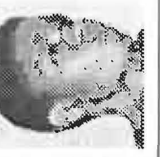
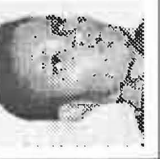
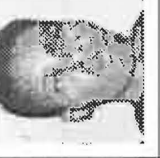
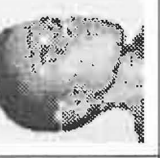

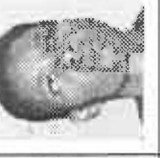














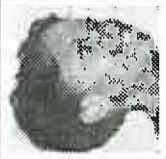









Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
76										
77										
78										
79										
80										

Table B. 20 : Database of subjects 81-82

Subject Number	View # 1	View # 2	View # 3	View # 4	View # 5	View # 6	View # 7	View # 8	View # 9	View # 10
81										
82										

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>


#define DIM 100
#define pi 3.14159265359
#define sigma 10


void main(void)
{
    FILE *fname,*out;
    unsigned char *fbuf,*h;
    int argc,px,py,count;
    int a,b,c,d,nabx,naby,maxa,maxb,len,max=0;
    float *r,*theta,*M;
    float dij,aij,pij,cij,t,y;
    char ch;



dij = 0;
aij = 0;
pij = 0;
cij = 0;


argc=2;
if (argc!=2){
printf("\n\n\n\n\n\n\n\n\n\nIncorrect Usage\n");
printf("correl FILENAME TEMPLATENAME");
exit(0);
}




if ((fname = fopen("face.raw","rb"))==NULL){
printf("Error, could not open input file");
exit(0);
}





if ((out = fopen("out","wb"))==NULL){
printf("Error, could not open output file");
exit(0);
}






fbuff = (unsigned char *)calloc((long)(DIM*DIM),1);
if (fbuff==NULL){
```

```

    printf("1. Error allocating memory, exiting");
    exit(0);
}

r = (float *) malloc(((DIM)*(DIM))*sizeof(float));
if (r==NULL){
    printf("3. Error allocating memory, exiting");
    exit(0);
}

theta = (float *) malloc(((DIM)*(DIM))*sizeof(float));
if (theta==NULL){
    printf("3. Error allocating memory, exiting");
    exit(0);
}

M = (float *) malloc(((DIM)*(DIM))*sizeof(float));
if (M==NULL){
    printf("3. Error allocating memory, exiting");
    exit(0);
}

h = (unsigned char *) malloc(((DIM)*(DIM))*sizeof(unsigned
char));
if (h==NULL){
    printf("3. Error allocating memory, exiting");
    exit(0);
}

    for(a=0;a<=(DIM-1);a++){
    for(b=0;b<=(DIM-1);b++){
        *(M+(a*(DIM))+b)=0;
        *(r+(a*(DIM))+b)=0;
        *(theta+(a*(DIM))+b)=0;
        *(h+(a*(DIM))+b)=0;
    }
}

count = DIM*DIM;
fread(fbuf,sizeof(char),count,fname);

for(a=1;a<=(DIM-2);a++){
    for(b=1;b<=(DIM-2);b++){
        nabx = (int) (*(fbuf+(a*(DIM))+b)) - (int) (*(fbuf+((a-1)*(DIM))+b));
        naby = (int) (*(fbuf+(a*(DIM))+b)) - (int) (*(fbuf+(a*(DIM))+ (b+1)));
        *(r+(a*DIM)+b) = log(1+sqrt(pow(nabx,2)+pow(naby,2)));
        if (naby == 0) *(theta+(a*DIM)+b) = 0;
        if (nabx == 0) *(theta+(a*DIM)+b) = pi/2;
        if ((nabx !=0) && (naby !=0)) *(theta+(a*DIM)+b) =
            atan((float)naby/nabx);
    }
}

for(a=1;a<=(DIM-2);a++){
    for(b=1;b<=(DIM-2);b++){
        if (*(r+(a*DIM)+b) ==0 ) continue;

```

```

for (c=1; c<=(DIM-2); c++) {
    for (d=1; d<=(DIM-2); d++) {
        if ((* (r+(a*DIM)+b) ==0 || (* (r+(c*DIM)+d) ==0)) continue;
        px = ceil((a+c)/2);
        py = ceil((b+d)/2)+1;
        if (d-b==0) aij=0;
        if (c-a==0) aij=pi/2;
        if (((d-b)!=0) && ((c-a)!=0)) aij = atan((float)(d-b)/(c-a));

        dij = 1/(sqrt(2*pi)*sigma)*exp(-1*(sqrt(pow(a-c,2)+pow(b-
            d,2))/(2*sigma)));

        pij = (1-cos(*(theta+(a*DIM)+b)+ *(theta+(c*DIM)+d)-2*aij))* (1-
            cos(*(theta+(a*DIM)+b)- *(theta+(c*DIM)+d)));

        cij = (dij*pi*(* (r+(a*DIM)+b))*(* (r+(c*DIM)+d)));

        * (M+(px*DIM)+py) = (* (M+(px*DIM)+py)) +cij;

        }
        cij=0;
        pij=0;
        dij=0;
        px=0;
        py=0;
    }
}
printf ("%d %d %d %d %.3f\n", a,b,c,d, * (M+(a*DIM)+b));
}

for (a=0; a<=(DIM-1); a++) {
    for (b=0; b<=(DIM-1); b++) {
        count = (unsigned int) (ceil(* (M+(a*DIM)+b)));
        * (h+(a*DIM)+b) = (unsigned char) count;
        if (((int) * (fbuf+(a*DIM)+b) != 0)) {
            * (h+(a*DIM)+b) = (unsigned char) 0;
        }
    }
}

count = DIM*DIM;
fwrite(h, sizeof(unsigned char), count, out);

for (a=0; a<=(DIM-1); a++) {
    for (b=0; b<=(DIM-1); b++) {
        if ((int) (* (h+(a*DIM)+b)) > max)
        {
            max = (int) * (h+(a*DIM)+b);
            maxa = a;
            maxb = b;
        }
    }
}

```



```
printf ("\nMax occurs at %d %d which is %d\n",maxb,maxa,max);

fclose(fname);
fclose(out);
free(fbuf);
free(r);
free(theta);
free(M);
free(h);
}
```

## APPENDIX D

This appendix describes the results of the tests performed with the RBF network. The results are presented in the following tabular format:

- Test No. - This is the number given to the particular test that was performed with the parameters and pattern set specified in the table.
- $\theta^+$  - RBF parameter. (See chapter 5)
- $\theta^-$  - RBF parameter. (See chapter 5)
- Min. O/P - This is the minimum output value that was classified correctly as a true positive.
- Training cycles - Number of cycles during which training of the network was performed.
- Hidden Units - The number of hidden units that were inserted by the DDA for each test.
- Class. Rate - This is the classification rate. The column consists of two numbers : the first number is the number of faces that were classified incorrectly, while the second is a percentage expression of the faces that were classified correctly. The true positive percentage is calculated as follows :

$$100 - 100 * \left( \frac{x}{y} \right) \quad \text{D- 1}$$

where x is the number of faces that were classified incorrectly and y is the total number of faces in the pattern set. The results are reported differently where tests were performed with “split databases” (characterised by a suffix of “r”, “e”, “a” or “b” in the test number – e.g. Table D.20). The second number in the classification

rate is percentage of faces that were classified incorrectly (false positives), while the first number is the true negatives (impostors that were classified as impostors).

- SSE - This is the sum of the squared errors of the network.
- MSE - This is the mean squared error of the network.

The SSE is computed with the following formula[13] :

$$SSE = \sum_{p \in \text{patterns}} \sum_{j \in \text{output}} (t_{pj} - o_{pj})^2 \quad \text{D-2}$$

where  $t_{pj}$  is the desired output of output neuron  $j$  on pattern  $p$  and  $o_{pj}$  is the actual output. The MSE is defined as the SSE divided by the number of patterns.

Test numbers rp1 to rp49 were preliminary tests that were implemented to observe the effect of varying different network parameters.

**Table D. 1** : Results from preliminary tests performed using u10snfr for training and u10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  is varied from 0.4 to 0.9, while  $\theta^-$  is kept constant at 0.2.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rp1	0.4	0.2	0.12254	3	319	100 – 78.261	1657.1408	3.60248
Rp2	0.5	0.2	0.12254	3	361	96 – 79.130	1513.1396	3.28943
Rp3	0.6	0.2	0.12254	3	383	92 – 80.00	1500.8227	3.26266
Rp4	0.7	0.2	0.12254	3	420	90 – 80.435	1500.4271	3.26180
Rp5	0.8	0.2	0.20207	3	439	90 – 80.435	1484.6414	3.22748
Rp6	0.9	0.2	0.20207	3	456	91 – 80.217	1475.7034	3.20805

**Table D. 2 :** Results from preliminary tests performed using u10snfr for training and u10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  is varied from 0.4 to 0.9, while  $\theta^-$  is kept constant at 0.02.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rp7	0.4	0.02	0.00117	3	418	62 – 86.522	155.39569	0.33782
Rp8	0.5	0.02	0.00508	3	432	65 – 85.870	145.47714	0.31625
Rp9	0.6	0.02	0.00508	3	441	66 – 85.652	138.63564	0.30138
Rp10	0.7	0.02	0.00508	3	451	66 – 85.652	133.5090	0.29024
Rp11	0.8	0.02	0.00417	3	450	66 – 85.652	131.82143	0.28657
Rp12	0.9	0.02	0.00417	3	460	66 – 85.652	130.45839	0.28361
Rp13	0.3	0.02	0.00117	3	396	65 – 85.870	184.35263	0.40077

**Table D. 3 :** Results from preliminary tests performed using u10snfr for training and u10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  is varied from 0.4 to 0.9, while  $\theta^-$  is kept constant at 0.002.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rp14	0.4	0.002	0.00010	3	396	62 – 86.522	64.38924	0.13998
Rp15	0.5	0.002	0.00010	3	444	61 – 86.739	57.00109	0.12393
Rp16	0.6	0.002	0.00009	3	453	61 – 86.739	50.01250	0.10872
Rp17	0.7	0.002	0.00009	3	456	61 – 86.739	48.88790	0.10628
Rp18	0.8	0.002	0.00009	3	460	60 – 86.957	47.46719	0.10319
Rp19	0.9	0.002	0.00009	3	460	60 – 86.957	47.46719	0.10319

**Table D. 4 :** Results from preliminary tests performed using u10snfr for training and u10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  is varied from 0.4 to 0.9, while  $\theta^-$  is kept constant at 0.0002.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rp20	0.4	0.0002	< 0.0001	3	446	62 – 86.522	34.27124	0.07450
Rp21	0.5	0.0002	< 0.0001	3	453	62 – 86.522	26.5960	0.05782
Rp22	0.6	0.0002	< 0.0001	3	456	62 – 86.522	25.09672	0.05456
Rp23	0.7	0.0002	< 0.0001	3	460	61 – 86.739	23.49164	0.05107
Rp24	0.8	0.0002	< 0.0001	3	460	61 – 86.739	23.49164	0.05107
Rp25	0.9	0.0002	< 0.0001	3	460	61 – 86.739	23.49164	0.05107

**Table D. 5 :** Results from preliminary tests performed using u10snfr for training and u10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  and  $\theta^-$  are both set to the same values that range from 0.1 to 0.9.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rp26	0.1	0.1	0.05254	3	226	92 – 80.000	900.47249	1.95745
Rp27	0.2	0.2	0.17147	3	226	108 – 76.522	1817.6272	3.9536
Rp28	0.3	0.3	0.37043	3	226	128 – 72.174	4363.3065	9.55067
Rp29	0.4	0.4	0.68661	3	226	159 – 65.435	10570.382	22.9790
Rp30	0.5	0.5	1.17583	3	226	269 – 54.565	24282.417	52.7878
Rp31	0.6	0.6	1.79402	3	226	276 – 40.000	53.607.853	116.538
Rp32	0.7	0.7	2.84040	3	226	342 – 25.652	115740.20	251.609
Rp33	0.8	0.8	4.22605	3	226	377 – 18.043	249495.34	542.381
Rp34	0.9	0.9	6.71538	3	226	432 – 6.087	551334.62	1198.55

**Table D. 6 :** Results from preliminary tests performed using u10snfr for training and u10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  and  $\theta^-$  are both set to the same values that range from 0.01 to 0.09.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rp35	0.01	0.01	0.00131	3	226	81 – 82.391	718.29431	1.56151
Rp36	0.02	0.02	0.00395	3	226	83 – 81.957	716.44800	1.55750
Rp37	0.03	0.03	0.00754	3	226	83 – 81.957	721.55493	1.56860
Rp38	0.04	0.04	0.01195	3	226	83 – 81.957	731.85840	1.59100
Rp39	0.05	0.05	0.01708	3	226	83 – 81.957	746.95850	1.62382
Rp40	0.06	0.06	0.02990	3	226	84 – 81.739	766.8673	1.66710
Rp41	0.07	0.07	0.02937	3	226	85 – 81.522	791.79700	1.72130
Rp42	0.08	0.08	0.03648	3	226	89 – 80.652	822.0708	1.78713
Rp43	0.09	0.09	0.04420	3	226	92 – 80.000	858.12762	1.86549

**Table D. 7 :** Results from preliminary tests performed using u10snfr for training and u10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  is varied from 0.4 to 0.9, while  $\theta^-$  is kept constant at 0.001.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rp44	0.4	0.001	0.0002	3	444	54 – 88.261	45.9228	0.09883
Rp45	0.5	0.001	0.0002	3	454	54 – 88.261	36.96737	0.08036
Rp46	0.6	0.001	0.0002	3	457	54 – 88.261	33.24289	0.07227
Rp47	0.7	0.001	0.0002	3	457	54 – 88.261	33.24289	0.07227
Rp48	0.8	0.001	0.0002	3	454	54 – 88.261	31.6600	0.06883
Rp49	0.9	0.001	0.0002	3	454	54 – 88.261	31.6600	0.06883

**Table D. 8 :** Results from tests performed using u10snfr for training and u10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
1	0.4	0.001	0.00003	3	441	62 – 86.522	50.11327	0.10894
2	0.4	0.002	0.0001	3	436	62 – 86.522	64.38924	0.13998
3	0.4	0.0002	0.00001	3	446	62 – 86.522	34.27124	0.07450
4	0.8	0.002	0.00009	3	460	60 – 86.957	47.6719	0.10319
5	0.8	0.0002	0.00009	3	460	60 – 86.957	23.49164	0.05107

**Table D. 9 :** Results from tests performed using u10snfe for training and u10snfr for testing. The neural network architecture consists of 99 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
6	0.4	0.001	0.00002	3	444	54 – 88.261	45.92238	0.09983
7	0.4	0.002	0.00007	3	441	54 – 88.261	55.94427	0.12162
8	0.4	0.0002	0.00001	3	450	57 – 87.609	30.04824	0.06532
9	0.8	0.002	0.00007	3	459	54 – 88.261	41.08091	0.08931
10	0.8	0.0002	0.00001	3	460	57 – 87.609	19.39404	0.04216

**Table D. 10 :** Results from tests performed using u10smfr for training and u10smfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
11	0.4	0.001	0.00003	3	441	61 – 86.739	45.98875	0.09998
12	0.4	0.002	0.0001	3	435	60 – 86.957	58.46727	0.12710
13	0.4	0.0002	0.00001	3	446	60 – 86.957	31.79949	0.06913
14	0.8	0.002	0.00008	3	4660	61 – 86.739	43.40290	0.09435
15	0.8	0.0002	0.00001	3	460	60 – 86.957	21.79029	0.04737

**Table D. 11 :** Results from tests performed using u10smfe for training and u10smfr for testing. The neural network architecture consists of 99 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
16	0.4	0.001	0.00007	3	447	70 – 84.783	47.29271	0.10281
17	0.4	0.002	0.000017	3	445	70 – 84.783	59.30832	0.12893
18	0.4	0.0002	0.00002	3	449	72 – 84.348	32.94275	0.07161
19	0.8	0.002	0.00007	3	459	70 – 84.783	45.08393	0.09801
20	0.8	0.0002	0.00002	3	460	72 – 84.348	21.03061	0.04572

**Table D. 12 :** Results from tests performed using u8snfr for training and u8snfe for testing. The neural network architecture consists of 63 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
21	0.4	0.001	0.00002	3	437	60 – 86.957	61.72296	0.13418
22	0.4	0.002	0.00007	3	437	60 – 86.957	72.06032	0.15665
23	0.4	0.0002	0.00003	3	443	61 – 86.739	41.26799	0.08971
24	0.8	0.002	0.00005	3	460	59 – 87.174	57.65048	0.12533
25	0.8	0.0002	0.00003	3	460	60 – 86.957	29.35118	0.06381

**Table D. 13 :** Results from tests performed using u8snfe for training and u8snfr for testing. The neural network architecture consists of 63 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
26	0.4	0.001	0.00002	3	443	57 – 87.609	53.28607	0.11584
27	0.4	0.002	0.00005	3	440	57 – 87.609	64.55523	0.14034
28	0.4	0.0002	0.00001	3	449	58 – 87.391	35.42840	0.07702
29	0.8	0.002	0.00005	3	459	57 – 87.609	49.87717	0.10843
30	0.8	0.0002	0.00001	3	460	58 – 87.391	24.7876	0.05389

**Table D. 14 :** Results from tests performed using u8smfr for training and u8smfe for testing. The neural network architecture consists of 63 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
31	0.4	0.001	0.00003	3	399	62 – 86.522	54.85690	0.11924
32	0.4	0.002	0.00008	3	399	63 – 86.304	64.87965	0.14104
33	0.4	0.0002	0.00001	3	443	62 – 86.522	37.52636	0.08158
34	0.8	0.002	0.00008	3	459	62 – 86.522	51.73994	0.11248
35	0.8	0.0002	0.00001	3	460	61 – 86.739	26.97575	0.05864

**Table D. 15 :** Results from tests performed using u8smfe for training and u8smfr for testing. The neural network architecture consists of 63 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
36	0.4	0.001	0.00001	3	444	70 – 84.783	60.10002	0.13065
37	0.4	0.002	0.00002	3	441	69 – 85.000	71.75588	0.15599
38	0.4	0.0002	0.00001	3	449	73 – 84.130	39.08024	0.08496
39	0.8	0.002	0.00002	3	459	69 – 85.000	56.49219	0.12281
40	0.8	0.0002	0.00001	3	460	73 – 84.130	27.67075	0.06015



**Table D. 16 :** Results from tests performed using uxtnfr for training and uxtnfe for testing. The neural network architecture consists of 54 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
41	0.4	0.001	0.00002	3	436	62 – 86.522	66.42831	0.14441
42	0.4	0.002	0.00005	3	433	61 – 86.739	79.3662	0.17254
43	0.4	0.0002	0.00003	3	443	63 – 86.304	43.78766	0.09519
44	0.8	0.002	0.00005	3	460	60 – 86.957	62.46791	0.13580
45	0.8	0.0002	0.00003	3	460	62 – 86.522	32.25453	0.07012

**Table D. 17 :** Results from tests performed using uxtnfe for training and uxtnfr for testing. The neural network architecture consists of 54 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
46	0.4	0.001	0.00001	3	443	57 – 87.609	53.48675	0.11628
47	0.4	0.002	0.00004	3	439	56 – 87.826	66.6961	0.14385
48	0.4	0.0002	0.00001	3	448	57 – 87.609	38.27567	0.08321
49	0.8	0.002	0.00004	3	459	56 – 87.826	52.99506	0.11521
50	0.8	0.0002	0.00001	3	459	57 – 87.609	27.63902	0.06008

**Table D. 18 :** Results from tests performed using uxtmfr for training and uxtmfe for testing. The neural network architecture consists of 54 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
51	0.4	0.001	0.00002	3	436	61 – 86.739	57.69643	0.12543
52	0.4	0.002	0.00007	3	434	62 – 86.522	68.47857	0.14887
53	0.4	0.0002	0.00001	3	444	61 – 86.739	38.26178	0.08318
54	0.8	0.002	0.00005	3	459	62 – 86.522	54.61316	0.11872
55	0.8	0.0002	0.00001	3	460	61 – 86.739	28.44660	0.06184

**Table D. 19** : Results from tests performed using uxtmfe for training and uxtmfr for testing. The neural network architecture consists of 54 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
56	0.4	0.001	0.00001	3	443	69 – 85.000	65.51014	0.14241
57	0.4	0.002	0.00003	3	439	66 – 85.652	76.49706	0.16630
58	0.4	0.0002	0.00001	3	448	71 – 84.565	43.54995	0.09467
59	0.8	0.002	0.00003	3	459	67 – 85.435	61.75870	0.13426
60	0.8	0.0002	0.00001	3	459	71 – 84.565	32.03211	0.06964

**Table D. 20** : Results from tests performed using 1u10snfr for training and 1u10snfe for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 2u10snfr and 2u10snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
61	0.4	0.001	0.00016	3	235	19 – 92.245	25.86086	0.10555
62	0.4	0.002	0.00042	3	235	19 – 92.245	32.13540	0.13116
63	0.4	0.0002	0.00002	3	238	19 – 92.245	17.02889	0.06951
64	0.8	0.002	0.00042	3	245	19 – 92.245	25.21136	0.10290
65	0.8	0.0002	0.00002	3	245	19 – 92.245	11.64131	0.04752
662R			0.00016			185 – 13.953		
672E			0.00016			184 – 14.88		
682R			0.00042			207 – 3.721		
692E			0.00042			207 – 3.721		
702R			0.00002			207 – 3.721		
712E			0.00002			207 – 3.721		
722R			0.00042			207 – 3.721		
732E			0.00042			208 – 3.256		
742R			0.00002			207 – 3.721		
752E			0.00002			207 – 3.721		

**Table D. 21** : Results from tests performed using 1u10snfe for training and 1u10snfr for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 2u10snfr and 2u10snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
76	0.4	0.001	0.00001	3	234	16 – 93.469	27.91176	0.11393
77	0.4	0.002	0.00002	3	233	15 – 93.878	33.19090	0.13547
78	0.4	0.0002	0.00001	3	239	18 – 92.653	17.28478	0.07055
79	0.8	0.002	0.00002	3	245	16 – 93.469	24.86629	0.10150
80	0.8	0.0002	0.00001	3	245	17 – 93.061	11.8466	0.04851
812R			0.00001			210 – 2.326		
822E			0.00001			207 – 3.721		
832R			0.00002			210 – 2.326		
842E			0.00002			207 – 3.721		
852R			0.00001			212 – 1.395		
862E			0.00001			207 – 3.721		
872R			0.00002			210 – 2.326		
882E			0.00002			207 – 3.721		
88A			0.00001			212 – 1.395		
88B			0.00001			207 – 3.721		

**Table D. 22** : Results from tests performed using 1u10smfr for training and 1u10smfe for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 2u10smfr and 2u10smfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
89	0.4	0.001	0.00001	3	238	18 – 92.653	14.38531	0.05872
90	0.4	0.002	0.00034	3	231	18 – 92.653	30.69027	0.12527
91	0.4	0.0002	0.00002	3	237	18 – 92.653	16.49629	0.06733
92	0.8	0.002	0.00034	3	237	18 – 92.653	23.33883	0.09526
93	0.8	0.0002	0.00002	3	245	18 – 92.653	11.37552	0.04643
94r			0.00001			210 – 2.326		
95e			0.00001			207 – 3.721		
96r			0.00034			210 – 2.326		
97e			0.00034			206 – 4.186		
98r			0.00002			211 – 1.860		
99e			0.00002			207 – 3.721		

100r			0.00034			209 – 2.791		
101e			0.00034			206 – 4.186		
102r			0.00002			211 – 1.860		
102e			0.00002			207 – 3.721		

**Table D. 23 :** Results from tests performed using 1u10smfe for training and 1u10smfr for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 2u10smfr and 2u10smfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
104	0.4	0.001	0.00013	3	239	26 – 89.388	26.15061	0.10674
105	0.4	0.002	0.00033	3	236	26 – 89.388	34.39678	0.14040
106	0.4	0.0002	0.00002	3	240	27 – 88.980	17.41475	0.07108
107	0.8	0.002	0.00033	3	245	26 – 89.388	26.21369	0.10699
108	0.8	0.0002	0.00002	3	245	27 – 88.980	11.80035	0.04816
109r			0.00013			209 – 2.791		
110e			0.00013			205 – 4.651		
111r			0.00033			209 – 2.791		
112e			0.00033			204 – 5.116		
113r			0.00002			208 – 3.256		
114e			0.00002			208 – 3.256		
115r			0.00033			209 – 2.791		
116e			0.00033			204 – 5.116		
116a			0.00002			208 – 3.256		
116b			0.00002			206 – 4.186		

**Table D. 24 :** Results from tests performed using 2u10snfr for training and 2u10snfe for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 1u10snfr and 1u10snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
117	0.4	0.001	0.0001	3	200	36 – 83.256	40.57496	0.18872
118	0.4	0.002	0.00025	3	200	37 – 82.791	46.72325	0.21732
119	0.4	0.0002	0.00001	3	204	36 – 83.256	28.70868	0.13353
120	0.8	0.002	0.00028	3	215	37 – 82.791	38.56425	0.17937
121	0.8	0.0002	0.00001	3	215	36 – 83.256	20.61650	0.09589
122r			0.0001			241 – 1.633		
123e			0.0001			238 – 2.857		

124r			0.00025			241 – 1.633		
125e			0.00025			238 – 2.857		
126r			0.00001			241 – 1.633		
127e			0.00001			238 – 2.857		
128r			0.00028			241 – 1.633		
129e			0.00028			238 – 2.857		
130r			0.00001			241 – 1.633		
131e			0.00001			238 – 2.857		

**Table D. 25** : Results from tests performed using 2u10snfe for training and 2u10snfr for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 1u10snfr and 1u10snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
132	0.4	0.001	0.0001	3	204	30 – 86.047	29.85820	0.13885
133	0.4	0.002	0.00025	3	202	30 – 86.047	36.61826	0.17032
134	0.4	0.0002	0.00001	3	209	29 – 86.512	18.75943	0.08725
135	0.8	0.002	0.00028	3	214	30 – 86.047	26.92869	0.12525
136	0.8	0.0002	0.00001	3	215	29 – 86.512	12.51114	0.05819
137r			0.0001			241 – 1.633		
138e			0.0001			239 – 2.449		
139r			0.00025			242 – 1.224		
140e			0.00025			240 – 2.041		
141r			0.00001			240 – 2.041		
142e			0.00001			238 – 2.857		
143r			0.00028			240 – 2.041		
144e			0.00028			238 – 2.857		
145r			0.00001			240 – 2.041		
146e			0.00001			238 – 2.857		

**Table D. 26 :** Results from tests performed using 2u10smfr for training and 2u10smfe for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 1u10smfr and 1u10smfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
147	0.4	0.001	0.00006	3	215	28 – 86.977	31.24568	0.14533
148	0.4	0.002	0.00018	3	215	28 – 86.977	39.68174	0.18457
149	0.4	0.0002	0.00001	3	215	28 – 86.977	20.88604	0.09714
150	0.8	0.002	0.00018	3	215	28 – 86.977	29.54831	0.13743
151	0.8	0.0002	0.00001	3	215	28 – 86.977	15.00530	0.06979
152			0.00006			239 – 2.449		
153			0.00006			238 – 2.857		
154			0.00018			239 – 2.449		
155			0.00018			239 – 2.449		
156			0.00001			239 – 2.449		
157			0.00001			215 – 0		
158			0.00018			215 – 0		
159			0.00018			215 – 0		
160			0.00001			215 – 0		
161			0.00001			215 – 0		

**Table D. 27 :** Results from tests performed using 2u10smfe for training and 2u10smfr for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 1u10smfr and 1u10smfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
162	0.4	0.001	0.001	3	201	37 – 82.791	39.30192	0.18280
163	0.4	0.002	0.002	3	202	37 – 82.791	43.53885	0.20251
164	0.4	0.0002	0.0002	3	206	36 – 83.256	27.55734	0.12815
165	0.8	0.002	0.002	3	211	37 – 82.791	35.15211	0.16350
166	0.8	0.0002	0.002	3	211	36 – 83.256	17.73732	0.08250
167			0.001			240 – 2.041		
168			0.001			241 – 1.633		
169			0.002			240 – 2.041		
170			0.002			241 – 1.633		
171			0.0002			240 – 2.041		
172			0.0002			241 – 1.633		
173			0.002			240 – 2.041		

174			0.002			241 – 1.633		
175			0.002			240 – 2.041		
176			0.002			241 – 1.633		

**Table D. 28 :** Results from tests performed using 1u8snfr for training and 1u8snfe for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 2u8snfr and 2u8snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
177	0.4	0.001	0.00041	3	234	17 – 93.061	29.89997	0.12204
178	0.4	0.002	0.00094	3	232	17 – 93.061	38.87866	0.15869
179	0.4	0.0002	0.00006	3	245	18 – 92.653	20.28575	0.08290
180	0.8	0.002	0.00094	3	245	17 – 93.061	31.45033	0.12837
181	0.8	0.0002	0.00006	3	245	18 – 92.653	14.89676	0.06080
182r			0.00041			210 – 2.326		
183e			0.00041			208 – 3.256		
184r			0.00094			210 – 2.326		
185e			0.00094			208 – 3.256		
186r			0.00006			208 – 3.256		
187e			0.00006			208 – 3.256		
188r			0.00094			207 – 3.721		
189e			0.00094			208 – 3.256		
190r			0.00006			208 – 3.256		
191e			0.00006			208 – 3.256		

**Table D. 29 :** Results from tests performed using 1u8snfe for training and 1u8snfr for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 2u8snfr and 2u8snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
192	0.4	0.001	0.00003	3	234	15 – 93.878	31.00424	0.12655
193	0.4	0.002	0.00001	3	234	15 – 93.878	37.88025	0.1461
194	0.4	0.0002	0.00001	3	236	16 – 93.469	21.51316	0.08781
195	0.8	0.002	0.00001	3	245	15 – 93.878	29.42778	0.12011
196	0.8	0.0002	0.00001	3	245	17 – 93.061	14.54067	0.05935
197r			0.00003			207 – 2.791		
198e			0.00003			208 – 2.256		
199r			0.00001			209 – 2.791		

200e			0.00001			208 – 3.256		
201r			0.00001			210 – 2.326		
202e			0.00001			206 – 4.186		
203r			0.00001			210 – 2.326		
204e			0.00001			208 – 3.256		
205r			0.00001			210 – 2.326		
206e			0.00001			206 – 4.186		

**Table D. 30 :** Results from tests performed using 1u8smfr for training and 1u8smfe for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 2u8smfr and 2u8smfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
207	0.4	0.001	0.00036	3	215	18 – 92.653	29.35135	0.11980
208	0.4	0.002	0.00092	3	231	18 – 92.653	35.24495	0.14386
209	0.4	0.0002	0.00004	3	234	18 – 92.653	20.85044	0.08510
210	0.8	0.002	0.00083	3	244	18 – 92.653	28.94075	0.11831
211	0.8	0.0002	0.00004	3	245	18 – 92.653	14.55873	0.05942
212r						212 – 1.395		
213e						208 – 3.256		
214r						212 – 1.395		
215e						208 – 3.256		
216r						212 – 1.395		
217e						208 – 3.256		
218r						212 – 1.395		
219e						208 – 3.256		
220r						212 – 1.395		
221e						208 – 3.256		

**Table D. 31 :** Results from tests performed using 1u8smfe for training and 1u8smfr for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 2u8smfr and 2u8smfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
222	0.4	0.001	0.00011	3	238	28 – 88.571	30.78066	0.12564
223	0.4	0.002	0.00028	3	236	28 – 88.571	39.60283	0.16164
224	0.4	0.0002	0.00001	3	239	28 – 88.571	20.47309	0.08356
225	0.8	0.002	0.00028	3	245	28 – 88.571	31.70593	0.12941



Appendix D

226	0.8	0.0002	0.00001	3	245	28 – 88.571	14.62940	0.05971
227r			0.00011			208 – 3.256		
228e			0.00011			204 – 5.116		
229r			0.00028			208 – 3.256		
230e			0.00028			204 – 5.116		
231r			0.00001			207 – 3.721		
232e			0.00001			204 – 5.116		
233r			0.00028			208 – 3.256		
234e			0.00028			203 – 5.581		
235r			0.00001			207 – 3.721		
236e			0.00001			204 – 5.116		

**Table D. 32 :** Results from tests performed using 2u8snfr for training and 2u8snfe for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 1u8snfr and 1u8snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
237	0.4	0.001	0.00003	3	200	35 – 83.721	44.17435	0.02054
238	0.4	0.002	0.00009	3	200	35 – 83.721	53.59703	0.24929
239	0.4	0.0002	0.00001	3	201	35 – 83.721	33.60172	0.15629
240	0.8	0.002	0.00009	3	215	35 – 83.721	43.89586	0.20417
241	0.8	0.0002	0.00001	3	215	35 – 83.721	23.93338	0.11132
242r			0.00003			242 – 1.224		
243e			0.00003			238 – 2.857		
244r			0.00009			242 – 1.224		
245e			0.00009			240 – 2.041		
246r			0.00001			238 – 2.857		
247e			0.00001			238 – 2.857		
248r			0.00009			242 – 1.224		
249e			0.00009			240 – 2.041		
250r			0.00001			242 – 1.224		
251e			0.00001			239 – 2.449		

**Table D. 33 :** Results from tests performed using 2u8snfe for training and 2u8snfr for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 1u8snfr and 1u8snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
252	0.4	0.001	0.00011	3	205	30 – 86.047	32.93299	0.15318
253	0.4	0.002	0.00019	3	202	29 – 86.512	42.71445	0.19867
254	0.4	0.0002	0.00001	3	202	30 – 86.047	23.01251	0.10703
255	0.8	0.002	0.00028	3	214	30 – 86.047	32.91035	0.15307
256	0.8	0.0002	0.00001	3	215	30 – 86.047	16.08066	0.07479
257r			0.00011			240 – 2.041		
258e			0.00011			239 – 2.449		
259r			0.00019			242 – 1.224		
260e			0.00019			240 – 2.041		
261r			0.00001			240 – 2.041		
262e			0.00001			239 – 2.449		
263r			0.00028			240 – 2.041		
264e			0.00028			238 – 2.857		
265r			0.00028			238 – 2.857		
266e			0.00028			240 – 2.041		

**Table D. 34 :** Results from tests performed using 2u8smfr for training and 2u8smfe for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 1u8smfr and 1u8smfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
267	0.4	0.001	0.00001	3	201	29 – 86.512	37.37313	0.17383
268	0.4	0.002	0.00029	3		29 – 86.512	43.98419	0.20458
269	0.4	0.0002	0.00001	3	206	27 – 87.422	23.67409	0.11011
270	0.8	0.002	0.00023	3	205	29 – 86.512	33.39094	0.15531
271	0.8	0.0002	0.00001	3	215	27 – 87.422	33.39094	0.15531
272r			0.0001			239 – 2.449		
273e			0.0001			238 – 2.857		
274r			0.00029			239 – 2.449		
275e			0.00029			238 – 2.857		
276r			0.00001			238 – 2.857		
277e			0.00001			239 – 2.449		
278r			0.00023			215 – 0		

279e			0.00023			215 – 0		
280r			0.00001			235 – 4.082		
281e			0.00001			238 – 2.857		

**Table D. 35** : Results from tests performed using 2u8smfe for training and 2u8smfr for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 1u8smfr and 1u8smfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
282	0.4	0.001	0.00004	3	202	33 – 84.651	44.36644	0.20636
283	0.4	0.002	0.00011	3	200	35 – 83.721	52.77213	0.24545
284	0.4	0.0002	0.00004	3	205	36 – 83.256	32.91333	0.15309
285	0.8	0.002	0.00014	3	215	33 – 84.651	42.80224	0.19908
286	0.8	0.0002	0.00004	3	215	36 – 83.256	23.61687	0.10985
287r			0.00004			240 – 2.041		
288e			0.00004			241 – 1.633		
289r			0.00011			240 – 2.041		
290e			0.00011			241 – 1.633		
291r			0.00004			240 – 2.041		
292e			0.00004			241 – 1.633		
293r			0.00014			240 – 2.041		
294e			0.00014			241 – 1.633		
295r			0.00004			240 – 2.041		
296e			0.00004			241 – 1.633		

**Table D. 36** : Results from tests performed using 1uxtntfr for training and 1uxtntfe for testing. The neural network architecture consists of 54 input elements and 41 output elements. The network is also tested on unseen data i.e. 2uxtntfr and 2uxtntfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
297	0.4	0.001	0.00040	3	234	17 – 93.061	31.99427	0.13059
298	0.4	0.002	0.00088	3	233	18 – 92.653	40.16058	0.16392
299	0.4	0.0002	0.00006	3	239	16 – 93.469	20.82939	0.08502
300	0.8	0.002	0.00087	3	245	18 – 92.653	32.68277	0.13340
301	0.8	0.0002	0.0006	3	245	16 – 93.469	15.47232	0.06315
302r			0.00040			206 – 4.186		
303e			0.00040			207 – 3.721		
304r			0.00088			206 – 4.186		

305e			0.00088			206 – 4.186		
306r			0.00006			207 – 3.721		
307e			0.00006			206 – 4.186		
308r			0.00087			206 – 4.186		
309e			0.00087			206 – 4.186		
310r			0.00006			207 – 3.721		
311e			0.00006			207 – 3.721		

**Table D. 37 :** Results from tests performed using 1uxtne for training and 1uxtnr for testing. The neural network architecture consists of 54 input elements and 41 output elements. The network is also tested on unseen data i.e. 2uxtnr and 2uxtne.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
312	0.4	0.001	0.00003	3	234	13 – 94.694	30.96836	0.12640
313	0.4	0.002	0.00002	3	230	13 – 94.694	39.2966	0.16038
314	0.4	0.0002	0.00001	3	236	15 – 93.878	21.55905	0.08800
315	0.8	0.002	0.00002	3	245	13 – 94.694	29.77066	0.12151
316	0.8	0.0002	0.00001	3	245	16 – 93.469	14.67016	0.05988
317r			0.00003			209 – 2.791		
318e			0.00003			207 – 3.721		
319r			0.00002			209 – 2.791		
320e			0.00002			208 – 3.256		
321r			0.00001			209 – 2.791		
322e			0.00001			207 – 3.721		
323r			0.00002			210 – 2.326		
324e			0.00002			208 – 3.256		
325r			0.00001			209 – 2.791		
326e			0.00001			207 – 3.721		

**Table D. 38 :** Results from tests performed using 2uxtnr for training and 2uxtne for testing. The neural network architecture consists of 54 input elements and 41 output elements. The network is also tested on unseen data i.e. 1uxtnr and 1uxtne.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
357	0.4	0.001	0.00002	3	195	35 – 83.721	49.13272	0.22936
358	0.4	0.002	0.00005	3	195	35 – 83.721	62.55439	0.28630
359	0.4	0.0002	0.00001	3	203	37 – 82.791	30.42624	0.14152

360	0.8	0.002	0.00005	3	215	35 – 82.721	48.98271	0.22783
361	0.8	0.0002	0.00002	3	215	37 – 82.791	26.79572	0.12463
362r			0.00002			242 – 1.224		
363e			0.00002			239 – 2.449		
364r			0.00005			242 – 1.224		
365e			0.00005			239 – 2.449		
366r			0.00001			242 – 1.224		
367e			0.00001			239 – 2.449		
368r			0.00005			242 – 1.224		
369e			0.00005			239 – 2.449		
370r			0.00002			242 – 1.224		
371e			0.00002			239 – 2.449		

**Table D. 39 :** Results from tests performed using 2uxtne for training and 2uxtnr for testing. The neural network architecture consists of 54 input elements and 41 output elements. The network is also tested on unseen data i.e. 1uxtnr and 1uxtne.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
372	0.4	0.001	0.00001	3	203	28 – 86.977	36.34881	1.6906
373	0.4	0.002	0.00012	3	201	30 – 86.047	46.85496	0.21793
374	0.4	0.0002	0.00002	3	209	31 – 85.581	21.46894	0.09986
375	0.8	0.002	0.00021	3	204	29 – 86.512	36.38642	0.16924
376	0.8	0.0002	0.00001	3	204	28 – 86.977	19.13155	0.08898
377r			0.00001			239 – 2.449		
378e			0.00001			238 – 2.857		
379r			0.00012			242 – 1.224		
380e			0.00012			239 – 2.449		
381r			0.00002			245 – 0		
382e			0.00002			239 – 2.449		
383r			0.00021			240 – 2.041		
384e			0.00021			237 – 3.265		
385r			0.00001			239 – 2.449		
386e			0.00001			237 – 3.265		

**Table D. 40 :** Results from preliminary tests performed using n10snfr for training and n10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  is varied from 0.4 to 0.9, while  $\theta^-$  is kept constant at 0.2.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rn1	0.4	0.2	0.18909	3	325	118 – 74.348	1491.6060	3.24262
Rn2	0.5	0.2	0.150510	3	369	109 – 76.304	1364.4552	2.96621
Rn3	0.6	0.2	0.15793	3	399	106 – 76.937	1321.0161	2.87177
Rn4	0.7	0.2	0.221380	3	425	104 – 77.391	1301.9577	2.83025
Rn5	0.8	0.2	0.221380	3	442	105 – 77.174	1296.3126	2.81807
Rn6	0.9	0.2	0.221380	3	456	105 – 77.174	1291.1429	2.80683

**Table D. 41 :** Results from preliminary tests performed using n10snfr for training and n10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  is varied from 0.4 to 0.9, while  $\theta^-$  is kept constant at 0.02.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rn7	0.4	0.02	0.002	3	421	79 – 82.826	132.18530	0.28730
Rn8	0.5	0.02	0.00249	3	435	79 – 82.826	122.24379	0.26575
Rn9	0.6	0.02	0.00587	3	444	79 – 82.826	116.01974	0.25222
Rn10	0.7	0.02	0.00587	3	450	80 – 82.609	113.06348	0.24579
Rn11	0.8	0.02	0.00587	3	457	80 – 82.609	107.90546	0.23458
Rn12	0.9	0.02	0.00587	3	460	79 – 82.826	106.98688	0.23258

**Table D. 42 :** Results from preliminary tests performed using n10snfr for training and n10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.  $\theta^+$  is varied from 0.4 to 0.9, while  $\theta^-$  is kept constant at 0.002.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
Rn13	0.4	0.002	0.00021	3	447	74 – 83.913	50.34787	0.10945
Rn14	0.5	0.002	0.00021	3	448	74 – 83.913	44.91980	0.09765
Rn15	0.6	0.002	0.00021	3	454	74 – 83.913	40.91613	0.08895
Rn16	0.7	0.002	0.00021	3	454	74 – 83.913	38.86154	0.08448

**Table D. 43 :** Results from tests performed using n10snfr for training and n10snfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
417	0.4	0.02	0.002	3	422	79 – 82.826	132.18530	0.28736
418	0.4	0.002	0.00021	3	441	74 – 83.913	50.34787	0.10945
419	0.8	0.02	0.000587	3	457	80 – 82.609	107.90546	0.23458
420	0.8	0.002	0.00021	3	460	74 – 83.913	37.86361	0.08231

**Table D. 44 :** Results from tests performed using n10snfe for training and n10snfr for testing. The neural network architecture consists of 99 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
421	0.4	0.02	0.00402	3	424	69 – 85.000	138.34605	0.30075
422	0.4	0.002	0.00014	3	444	70 – 84.783	50.14229	0.10901
423	0.8	0.02	0.00402	3	457	69 – 85.000	102.22397	0.22223
424	0.8	0.002	0.00014	3	460	69 – 85.000	35.54495	0.07727

**Table D. 45 :** Results from tests performed using n10smfr for training and n10smfe for testing. The neural network architecture consists of 99 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
425	0.4	0.02	0.00412	3	422	70 – 84.783	119.32616	0.25940
426	0.4	0.002	0.00007	3	442	66 – 85.652	45.98283	0.9996
427	0.8	0.02	0.00348	3	457	70 – 84.783	102.21089	0.2220
428	0.8	0.002	0.00001	3	460	67 – 85.435	35.64630	0.07149

**Table D. 46 :** Results from tests performed using n10smfe for training and n10smfr for testing. The neural network architecture consists of 99 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
429	0.4	0.02	0.00530	3	425	83 – 81.957	147.99701	0.32173
430	0.4	0.002	0.00020	3	446	83 – 81.957	51.73392	0.11247
431	0.8	0.02	0.00530	3	458	81 – 82.391	108.77761	0.23643
432	0.8	0.002	0.00020	3	459	83 – 81.957	38.90541	0.08458

**Table D. 47 :** Results from tests performed using n8snfr for training and n8snfe for testing. The neural network architecture consists of 63 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
433	0.4	0.02	0.0010	3	415	78 – 83.043	152.11498	0.33076
434	0.4	0.002	0.00009	3	438	72 – 84.348	57.15257	0.12424
435	0.8	0.02	0.00360	3	456	78 – 83.043	123.75614	0.26904
436	0.8	0.002	0.00009	3	460	72 – 84.348	45.66237	0.09927

**Table D. 48 :** Results from tests performed using n8snfe for training and n8snfr for testing. The neural network architecture consists of 63 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
437	0.4	0.02	0.00268	3	412	74 – 83.913	173.60875	0.37741
438	0.4	0.002	0.00006	3	442	69 – 85.000	58.59186	0.12737
439	0.8	0.02	0.00281	3	457	70 – 84.783	116.21783	0.25265
440	0.8	0.002	0.00006	3	459	69 – 85.000	44.01051	0.09568

**Table D. 49 :** Results from tests performed using n8smfr for training and n8smfe for testing. The neural network architecture consists of 63 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
441	0.4	0.02	0.00218	3	419	68 – 85.217	133.29573	0.28977
442	0.4	0.002	0.00005	3	439	67 – 85.435	53.06662	0.11536
443	0.8	0.02	0.00218	3	455	67 – 85.435	116.50249	0.25327
444	0.8	0.002	0.00005	3	459	67 – 85.435	43.26603	0.09406

**Table D. 50 :** Results from tests performed using n8smfe for training and n8smfr for testing. The neural network architecture consists of 63 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
445	0.4	0.02	0.00423	3	413	76 – 83.478	171.40312	0.37262
446	0.4	0.002	0.000145	3	446	77 – 83.261	59.55173	0.12946
447	0.8	0.02	0.00423	3	458	77 – 83.261	125.27199	0.27233
448	0.8	0.002	0.00015	3	459	78 – 83.043	47.27989	0.10278



**Table D. 51** : Results from tests performed using ntxtfr for training and ntxtfe for testing. The neural network architecture consists of 54 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
449	0.4	0.02	0.00123	3	414	75 – 83.696	155.7614	0.33864
450	0.4	0.002	0.00068	3	434	71 – 84.565	59.16640	0.12862
451	0.8	0.02	0.00331	3	447	74 – 83.913	128.65012	0.27967
452	0.8	0.002	0.00008	3	451	70 – 84.783	47.98024	0.10430

**Table D. 52** : Results from tests performed using ntxtfe for training and ntxtfr for testing. The neural network architecture consists of 54 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
453	0.4	0.02	0.0027	3	415	68 – 85.217	175.62756	0.38180
454	0.4	0.002	0.0006	3	430	64 – 86.087	65.06663	0.14145
455	0.8	0.02	0.00283	3	457	66 – 85.652	121.26225	0.26361
456	0.8	0.002	0.0007	3	458	64 – 86.087	48.06721	0.10449

**Table D. 53** : Results from tests performed using ntxtmfr for training and ntxtmfe for testing. The neural network architecture consists of 54 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
457	0.4	0.02	0.00178	3	416	68 – 85.217	141.8667	0.30845
458	0.4	0.002	0.00003	3	440	69 – 85.000	55.18320	0.1996
459	0.8	0.02	0.00150	3	455	66 – 85.652	122.20951	0.26567
460	0.8	0.002	0.0003	3	459	68 – 85.217	45.52563	0.09897

**Table D. 54** : Results from tests performed using ntxtmfe for training and ntxtmfr for testing. The neural network architecture consists of 54 input elements and 82 output elements.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
461	0.4	0.02	0.00447	3	408	82 – 82.174	180.13457	0.39160
462	0.4	0.002	0.00017	3	444	82 – 82.174	64.34874	0.13989
463	0.8	0.02	0.00447	3	458	79 – 82.826	130.39729	0.28347
464	0.8	0.002	0.00017	3	458	82 – 82.174	51.34821	0.11163

**Table D. 55** : Results from tests performed using 1n10snfr for training and 1n10snfe for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 2n10snfr and 2n10snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
465	0.4	0.02	0.01348	3	227	26 – 89.388	76.76943	0.31334
466	0.4	0.002	0.00048	3	237	23 – 90.612	26.86808	0.10967
467	0.8	0.02	0.01318	3	242	26 – 89.388	65.86154	0.26882
468	0.8	0.002	0.00048	3	245	23 – 90.612	21.88599	0.08933
469r			0.01348			212 – 1.395		
470e			0.01348			209 – 2.791		
471r			0.00048			211 – 1.860		
472e			0.00048			208 – 3.256		
473r			0.01318			212 – 1.395		
474e			0.01318			209 – 2.791		
475r			0.00048			211 – 1.860		
476e			0.00048			208 – 3.256		

**Table D. 56** : Results from tests performed using 1n10snfe for training and 1n10snfr for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 2n10snfr and 2n10snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
477	0.4	0.02	0.00155	3	223	31 – 87.347	91.98940	0.3747
478	0.4	0.002	0.00003	3	235	29 – 88.163	31.6893	0.12934
479	0.8	0.02	0.00155	3	243	31 – 87.347	67.47902	0.27542
480	0.8	0.002	0.00003	3	244	28 – 88.571	24.54739	0.10019
481r			0.00155			212 – 1.395		
482e			0.00155			207 – 3.721		
483r			0.00003			212 – 1.395		
484e			0.00003			208 – 3.256		
485r			0.00155			212 – 1.395		
486e			0.00155			208 – 3.256		
487r			0.00003			212 – 1.395		
488e			0.00003			208 – 3.256		

**Table D. 57 :** Results from tests performed using 2n10snfr for training and 2n10snfe for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 1n10snfr and 1n10snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
489	0.4	0.02	0.00493	3	190	37 – 82.791	88.2693	0.41082
490	0.4	0.002	0.00021	3	190	37 – 82.791	38.66991	0.17986
491	0.8	0.02	0.00566	3	215	39 – 81.860	71.42371	0.33220
492	0.8	0.002	0.00021	3	215	38 – 82.326	28.22557	0.13128
493r			0.00493			242 – 1.224		
494e			0.00493			239 – 2.449		
495r			0.00021			242 – 1.224		
496e			0.00021			239 – 2.449		
497r			0.00566			242 – 1.224		
498e			0.00566			239 – 2.449		
499r			0.00021			242 – 1.224		
500e			0.00021			239 – 2.449		

**Table D. 58 :** Results from tests performed using 2n10snfe for training and 2n10snfr for testing. The neural network architecture consists of 99 input elements and 41 output elements. The network is also tested on unseen data i.e. 1n10snfr and 1n10snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
501	0.4	0.02	0.00321	3	215	32 – 85.116	75.80135	0.35256
502	0.4	0.002	0.00014	3	203	31 – 85.581	33.67884	0.15665
503	0.8	0.02	0.00427	3	214	33 – 84.651	60.97275	0.28359
504	0.8	0.002	0.00010	3	215	30 – 86.047	21.89684	0.10185
505r			0.00321			241 – 1.633		
506e			0.00321			237 – 3.265		
507r			0.00014			241 – 1.633		
508e			0.00014			238 – 2.857		
509r			0.00427			241 – 1.633		
510e			0.00427			237 – 3.265		
511r			0.00010			241 – 1.633		
512e			0.00010			237 – 3.265		

**Table D. 59 :** Results from tests performed using 1n8snfr for training and 1n8snfe for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 2n8snfr and 2n8snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
513	0.4	0.02	0.01598	3	218	26 – 89.388	91.01023	0.37147
514	0.4	0.002	0.00060	3	235	25 – 89.796	32.49483	0.13263
515	0.8	0.02	0.01546	3	242	26 – 89.388	75.56332	0.30842
516	0.8	0.002	0.00060	3	245	25 – 89.796	26.76291	0.10924
517r			0.01598			211 – 1.860		
518e			0.01598			208 – 3.256		
519r			0.00060			211 – 1.860		
520e			0.00060			208 – 3.256		
521r			0.01546			211 – 1.860		
522e			0.01546			208 – 3.256		
523r			0.00060			211 – 1.860		
524e			0.00060			208 – 3.256		

**Table D. 60 :** Results from tests performed using 1n8snfe for training and 1n8snfr for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 2n8snfr and 2n8snfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
525	0.4	0.02	0.00146	3	217	29 – 88.163	83.15492	0.33941
526	0.4	0.002	0.00003	3	220	27 – 88.980	30.02367	0.12255
527	0.8	0.02	0.00146	3	243	29 – 88.163	74.48922	0.30404
528	0.8	0.002	0.00003	3	243	26 – 89.388	26.87027	0.10907
529r			0.00146			213 – 0.930		
530e			0.00146			207 – 3.721		
531r			0.00003			213 – 0.930		
532e			0.00003			213 – 0.930		
533r			0.00146			213 – 0.930		
534e			0.00146			207 – 3.721		
535r			0.00003			213 – 0.930		
536e			0.00003			206 – 4.186		

**Table D. 61** : Results from tests performed using 2n8snfr for training and 2n8snfe for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 1n8snfr and 1n8snfe.

Test No.	$\theta^1$	$\theta^2$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
537	0.4	0.02	0.00215	3	187	35 – 83.721	100.34458	0.46672
538	0.4	0.002	0.00018	3	199	37 – 82.791	45.46615	0.21147
539	0.8	0.02	0.00535	3	214	37 – 82.791	79.41990	0.36939
540	0.8	0.002	0.00018	3	215	37 – 82.791	33.02560	0.153.61
541r			0.00215			242 – 1.224		
542e			0.00215			239 – 2.449		
543r			0.00018			242 – 1.224		
544e			0.00018			241 – 1.633		
545r			0.00535			242 – 1.224		
546e			0.00535			239 – 2.449		
547r			0.00018			242 – 1.224		
548e			0.00018			239 – 2.449		

**Table D. 62** : Results from tests performed using 2n8snfe for training and 2n8snfr for testing. The neural network architecture consists of 63 input elements and 41 output elements. The network is also tested on unseen data i.e. 1n8snfr and 1n8snfe.

Test No.	$\theta^1$	$\theta^2$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
549	0.4	0.02	0.00382	3	194	33 – 84.651	90.40643	0.42050
550	0.4	0.002	0.00006	3	202	32 – 85.116	31.30560	0.14561
551	0.8	0.02	0.00339	3	214	32 – 85.116	71.40965	0.33214
552	0.8	0.002	0.00007	3	214	32 – 85.116	27.84142	0.12949
553r			0.00382			241 – 1.633		
554e			0.00382			238 – 2.857		
555r			0.00006			240 – 2.041		
556e			0.00006			239 – 2.449		
557r			0.00339			241 – 1.633		
558e			0.00339			238 – 2.857		
559r			0.00007			241 – 1.633		
560e			0.00007			239 – 2.449		

**Table D. 63 :** Results from tests performed using 1nxtnfr for training and 1nxtnfe for testing. The neural network architecture consists of 54 input elements and 41 output elements. The network is also tested on unseen data i.e. 2nxtnfr and 2nxtnfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
561	0.4	0.02	0.01478	3	194	25 – 89.796	99.27113	0.40519
562	0.4	0.002	0.00064	3	202	24 – 90.204	31.90216	0.13021
563	0.8	0.02	0.01478	3	214	26 – 89.388	78.32628	0.31970
564	0.8	0.002	0.00064	3	214	24 – 90.204	27.34801	0.11162
565r			0.01478			210 – 2.326		
566e			0.01478			208 – 3.256		
567r			0.00064			210 – 2.326		
568e			0.00064			209 – 2.791		
569r			0.01478			211 – 1.860		
570e			0.01478			209 – 2.791		
571r			0.00064			210 – 2.326		
572e			0.00064			209 – 2.791		

**Table D. 64 :** Results from tests performed using 1nxtnfe for training and 1nxtnfr for testing. The neural network architecture consists of 54 input elements and 41 output elements. The network is also tested on unseen data i.e. 2nxtnfr and 2nxtnfe.

Test No.	$\theta^+$	$\theta^-$	Min. O/P	Train. Cycles	Hidden Units	Class. Rate	SSE	MSE
573	0.4	0.02	0.00139	3	312	27 – 88.980	103.47397	0.42234
574	0.4	0.002	0.00002	3	328	24 – 90.204	35.84742	0.14632
575	0.8	0.02	0.00138	3	338	26 – 89.388	75.09632	0.30652
576	0.8	0.002	0.00002	3	339	25 – 89.796	28.09947	0.11469
577r			0.00139			211 – 1.860		
578e			0.00139			208 – 3.256		
579r			0.00002			211 – 1.860		
580e			0.00002			206 – 4.186		
581r			0.00138			210 – 2.326		
582e			0.00138			211 – 1.860		
583r			0.00002			211 – 1.860		
584e			0.00002			204 – 4.186		

**Table D. 65 :** Results from tests performed using 2nxtnfr for training and 2nxtnfe for testing. The neural network architecture consists of 54 input elements and 41 output elements. The network is also tested on unseen data i.e. 1nxtnfr and 1nxtnfe.

585	0.4	0.02	0.00186	3	283	38 – 82.326
586	0.4	0.002	0.0002	3	291	36 – 83.256

- Cut off - This figure represents the threshold that was used in the output neuron. If the result of the output neuron was below the cut off, it was classified as a 0 (negative), else it was classified as a 1 (positive).
- Person rec. (%) - This raw figure defines the person recognition, as opposed to face recognition. If at least one of the views of a subject is incorrectly classified, then the person recognition is incremented. It describes the number of persons/people that were incorrectly classified (i.e. classified as the valid subject for which the network was trained ). It is reported as a raw value (as a false positive) as well as a percentage (as a true negative).
- Face rec. (%) - This figure describes face recognition . It describes the number of faces that were incorrectly classified, in relation to the whole database. It is reported as a raw value (as a false positive) and as a percentage (as a true negative)



- Subject's output(A) - These figures are the outputs for the views of the valid subject for which the network was trained.
- Subject's output(B) - These figures represent outputs for views **where the valid subject was wearing hats, caps or glasses.**

Cells of the following tables that contain information for the Subject's output(A) and Subject's output(B) are thickened. This is done to show that the last two columns should be treated as separate tables, and have no relationship to the threshold information of column 1. They are the outputs for the 5 different views in the testing set.

The person recognition percentage is calculated as follows:

$$100 - 100 * \left( \frac{\text{person recognition}}{83} \right) \quad \text{E- 1}$$

The face recognition percentage is calculated as follows:

$$100 - 100 * \left( \frac{\text{face recognition}}{465} \right) \quad \text{E- 2}$$

Notice that the results for face recognition are calculated with a variable of 465 representing the whole database. However 83\*5 is 415. The extra 50 are the 50/5=10 extra sets of images (See Appendix B for an analysis of the 10 extra sets of images) for valid subjects that wore glasses, hats and/or caps, head bands and that had occluded views.

**Table E. 1 :** Results from neural network that was trained to classify Subject #1.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	1	1	98.795	99.785	0.99495	N/A
0.85	1	1	98.795	99.785	0.99631	
0.9	0	0	100	100	0.99631	
0.93	0	0	100	100	0.96367	
0.95	0	0	100	100	0.99291	

**Table E. 2 :** Results from neural network that was trained to classify Subject #2.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	7	19	91.566	95.914	0.88489	0.02024
0.85	7	19	91.566	95.914	0.86103	0.37862
0.9	6	13	92.771	97.204	0.95482	0.91624
0.93	5	7	93.976	98.495	0.00031	0.03972
0.95	5	7	93.976	98.495	0.42697	0.07588

**Table E. 3 :** Results from neural network that was trained to classify Subject #3.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	4	6	95.181	98.710	0.92711	0.35226
0.85	2	3	97.590	99.355	0.92335	0.8276
0.9	1	1	98.795	99.785	0.89891	0.85644
0.93	0	0	100	100	0.85824	0.89417
0.95	0	0	100	100	0.9462	0.8949

**Table E. 4 :** Results from neural network that was trained to classify Subject #4.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.83883	
0.85	0	0	100	100	0.89088	
0.9	0	0	100	100	0.91099	
0.93	0	0	100	100	0.91018	
0.95	0	0	100	100	0.91018	

**Table E. 5 :** Results from neural network that was trained to classify Subject #5.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.94188	
0.85	0	0	100	100	0.96897	
0.9	0	0	100	100	0.9244	
0.93	0	0	100	100	0.10822	
0.95	0	0	100	100	0.71816	

**Table E. 6 :** Results from neural network that was trained to classify Subject #6.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	7	96.386	98.495	0.9012	
0.85	3	7	96.386	98.495	0.9943	
0.9	3	5	96.386	98.925	0.9976	
0.93	3	4	96.386	99.140	0.7372	
0.95	0	0	100	100	0.9135	

**Table E. 7 :** Results from neural network that was trained to classify Subject #7.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.98128	
0.85	0	0	100	100	0.99705	
0.9	0	0	100	100	0.95257	
0.93	0	0	100	100	0.13845	
0.95	0	0	100	100	0.94679	

**Table E. 8 :** Results from neural network that was trained to classify Subject #8.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	4	5	95.181	98.925	0.85578	0.00127
0.85	2	3	97.590	99.355	0.99596	0.001715
0.9	1	2	98.795	99.570	0.98339	0.04292
0.93	1	1	98.795	99.785	0.31658	0.00769
0.95	1	1	98.795	99.785	0.9907	0.00708

**Table E. 9 :** Results from neural network that was trained to classify Subject #9.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	7	19	91.566	95.914	0.37867	0.00318
0.85	6	17	92.771	96.344	0.98969	0.83455
0.9	6	15	92.771	96.774	0.99881	0.96208
0.93	5	14	93.976	96.989	0.9081	0.20148
0.95	5	13	93.976	97.204	0.99421	0.2683

**Table E. 10 :** Results from neural network that was trained to classify Subject #10.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	7	20	91.566	95.699	0.99307	0.12078
0.85	6	18	92.771	96.129	0.99307	0.17775
0.9	4	14	95.181	96.989	0.99519	0.81037
0.93	4	14	95.181	96.989	0.91586	0.12937
0.95	4	12	95.181	97.419	0.97214	0.12937

0.99645 0.9961 0.99614 0.99519 0.99307

**Table E. 11 :** Results from neural network that was trained to classify Subject #11.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	1	5	98.795	98.925	0.69349	
0.85	1	5	98.795	98.925	0.95753	
0.9	1	5	98.795	98.925	0.98723	
0.93	1	5	98.795	98.925	0.93339	
0.95	1	4	98.795	99.140	0.8408	

**Table E. 12 :** Results from neural network that was trained to classify Subject #12.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	3	5	96.386	98.925	0.95798	
0.85	1	3	98.795	99.355	0.9769	
0.9	0	0	100	100	0.97282	
0.93	0	0	100	100	0.95899	
0.95	0	0	100	100	0.97948	

**Table E. 13 :** Results from neural network that was trained to classify Subject #13.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	8	14	90.361	96.989	0.62348	
0.85	8	14	90.361	96.989	0.53019	
0.9	7	13	91.566	97.204	0.99929	
0.93	7	11	91.566	97.204	0.85068	
0.95	6	9	92.771	98.065	0.99181	

**Table E. 14 :** Results from neural network that was trained to classify Subject #14.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.9324	
0.85	0	0	100	100	0.93139	
0.9	0	0	100	100	0.93123	
0.93	0	0	100	100	0.24352	
0.95	0	0	100	100	0.840153	

**Table E. 15 :** Results from neural network that was trained to classify Subject #15.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	8	19	90.361	95.914	0.99542	
0.85	8	17	90.361	96.344	0.99466	
0.9	8	13	90.361	97.204	0.99542	
0.93	8	12	90.361	97.419	0.99409	
0.95	5	8	93.976	98.280	0.99466	

**Table E. 16 :** Results from neural network that was trained to classify Subject #16

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	7	96.386	98.495	0.13419	
0.85	2	5	97.590	98.925	0.96207	
0.9	2	3	97.590	99.355	0.93346	
0.93	0	0	100	100	0.92328	
0.95	0	0	100	100	0.81901	

**Table E. 17 :** Results from neural network that was trained to classify Subject #17.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	10	97.590	97.849	0.98511	0.94035
0.85	2	10	97.590	97.849	0.99274	0.95418
0.9	2	10	97.590	97.849	0.98935	0.98702
0.93	2	9	97.590	98.065	0.9886	0.73897
0.95	2	9	97.590	98.065	0.99274	0.96608

**Table E. 18 :** Results from neural network that was trained to classify Subject #18.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	4	96.386	99.140	0.0575	
0.85	3	4	96.386	99.140	0.8644	
0.9	1	2	98.795	99.570	0.95402	
0.93	1	1	98.795	99.785	0.86414	
0.95	1	1	98.795	99.785	0.95402	

**Table E. 19 :** Results from neural network that was trained to classify Subject #19.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	1	1	98.795	98.785	0.83679	
0.85	0	0	100	100	0.98602	
0.9	0	0	100	100	0.99614	
0.93	0	0	100	100	0.0705	
0.95	0	0	100	100	0.94247	

**Table E. 20 :** Results from neural network that was trained to classify Subject #20.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	6	15	92.771	96.774	0.86722	
0.85	5	12	93.976	97.419	0.98241	
0.9	5	10	93.976	97.849	0.98261	
0.93	4	6	95.181	98.710	0.11736	
0.95	3	4	96.386	99.140	0.87175	

**Table E. 21 :** Results from neural network that was trained to classify Subject #21.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	1	5	98.795	98.925	0.96162	
0.85	1	5	98.795	98.925	0.94887	
0.9	1	5	98.795	98.925	0.99287	
0.93	1	5	98.795	98.925	0.00142	
0.95	1	5	98.795	98.925	0.02777	



**Table E. 22** : Results from neural network that was trained to classify Subject #22.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	7	17	91.566	96.344	0.97676	
0.85	6	16	92.771	96.559	0.99206	
0.9	5	14	93.976	96.989	0.99272	
0.93	5	11	93.976	97.634	0.89788	
0.95	5	9	93.976	98.065	0.90133	

**Table E. 23** : Results from neural network that was trained to classify Subject #23.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	7	21	91.566	95.484	0.99675	
0.85	7	21	91.566	95.484	0.99741	
0.9	6	20	92.771	95.699	0.99748	
0.93	2	18	92.771	96.129	0.13449	
0.95	5	17	93.976	96.344	0.039991	

**Table E. 24** : Results from neural network that was trained to classify Subject #24.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	3	97.590	99.355	0.99584	
0.85	2	3	97.590	99.355	0.99911	
0.9	1	2	98.795	99.570	0.97971	
0.93	1	1	98.795	99.785	0.99858	
0.95	1	1	98.795	99.785	0.99891	

**Table E. 25** : Results from neural network that was trained to classify Subject #25.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	8	29	90.361	93.763	0.08674	
0.85	8	26	90.361	94.409	0.97209	
0.9	7	24	91.566	94.839	0.97407	
0.93	7	24	91.566	94.839	0.42602	
0.95	7	19	91.566	95.914	0.92569	

**Table E. 26** : Results from neural network that was trained to classify Subject #26.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.98132	
0.85	0	0	100	100	0.98943	
0.9	0	0	100	100	0.98937	
0.93	0	0	100	100	0.00398	
0.95	0	0	100	100	0.00581	

**Table E. 27** : Results from neural network that was trained to classify Subject #27.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	2	97.590	99.570	0.20478	
0.85	2	2	97.590	99.570	0.99469	
0.9	2	2	97.590	99.570	0.99513	
0.93	2	2	97.590	99.570	0.44105	
0.95	1	1	98.795	99.785	0.90288	

**Table E. 28** : Results from neural network that was trained to classify Subject #28.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	9	29	89.157	93.763	0.9864	
0.85	9	25	89.157	94.624	0.99197	
0.9	9	22	89.157	95.269	0.97746	
0.93	9	16	89.157	96.559	0.9701	
0.95	6	11	92.771	97.634	0.99507	

**Table E. 29** : Results from neural network that was trained to classify Subject #29.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	5	5	93.976	98.925	0.12702	
0.85	4	4	95.181	99.140	0.98013	
0.9	3	3	96.386	99.355	0.99065	
0.93	3	3	96.386	99.355	0.29952	
0.95	3	3	96.386	99.355	0.85968	

**Table E. 30** : Results from neural network that was trained to classify Subject #30.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	5	9	93.976	98.065	0.98632	
0.85	5	9	93.976	98.065	0.99752	
0.9	4	7	95.181	98.495	0.99616	
0.93	4	5	95.181	98.925	0.99803	
0.95	4	5	95.181	98.925	0.99858	

**Table E. 31** : Results from neural network that was trained to classify Subject #31.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	7	12	91.566	97.419	0.08617	
0.85	5	9	93.976	98.065	0.99844	
0.9	4	8	95.181	98.280	0.99901	
0.93	4	8	95.181	98.280	0.97903	
0.95	3	7	96.386	98.495	0.99859	

**Table E. 32** : Results from neural network that was trained to classify Subject #32.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	11	96.386	97.634	0.99877	
0.85	3	10	96.386	97.849	0.99823	
0.9	3	10	96.386	97.849	0.99668	
0.93	3	10	96.386	97.849	0.98122	
0.95	3	9	96.386	98.065	0.99522	

**Table E. 33** : Results from neural network that was trained to classify Subject #33.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	4	11	95.181	97.634	0.95721	
0.85	3	9	96.386	98.065	0.94712	
0.9	3	9	96.386	98.065	0.9945	
0.93	3	9	96.386	98.065	0.50394	
0.95	2	5	97.590	98.925	0.25731	

**Table E. 34** : Results from neural network that was trained to classify Subject #34.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	5	9	93.976	98.065	0.0078	0.9800
0.85	3	7	96.386	98.495	0.9606	0.9936
0.9	2	5	97.590	98.925	0.9733	0.9942
0.93	2	4	97.590	99.140	0.8723	0.9911
0.95	2	3	97.590	99.355	0.9266	0.9942

**Table E. 35** : Results from neural network that was trained to classify Subject #35.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	6	97.590	98.710	0.96128	
0.85	2	6	97.590	98.710	0.98483	
0.9	2	6	97.590	98.710	0.98837	
0.93	1	5	98.795	98.925	0.98407	
0.95	1	5	98.795	98.925	0.97923	

**Table E. 36** : Results from neural network that was trained to classify Subject #36.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.93101	
0.85	0	0	100	100	0.99137	
0.9	0	0	100	100	0.99465	
0.93	0	0	100	100	0.94933	
0.95	0	0	100	100	0.99411	

**Table E. 37** : Results from neural network that was trained to classify Subject #37.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	6	11	92.771	97.634	0.99216	
0.85	6	11	92.771	97.634	0.99341	
0.9	5	10	93.976	97.849	0.99007	
0.93	5	10	93.976	97.849	0.99341	
0.95	5	10	93.976	97.849	0.99107	

**Table E. 38** : Results from neural network that was trained to classify Subject #38.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	5	15	93.976	96.774	0.11386	
0.85	5	14	93.976	96.989	0.98715	
0.9	5	14	93.976	96.989	0.99872	
0.93	5	13	93.976	97.204	0.04114	
0.95	5	13	93.976	97.204	0.99092	

**Table E. 39** : Results from neural network that was trained to classify Subject #39.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	2	97.590	99.570	0.94308	
0.85	2	2	97.590	99.570	0.991	
0.9	2	2	97.590	99.570	0.99977	
0.93	2	2	97.590	99.570	0.03307	
0.95	2	2	97.590	99.570	0.44845	

**Table E. 40** : Results from neural network that was trained to classify Subject #40.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	6	97.590	98.710	0.98995	
0.85	2	6	97.590	98.710	0.98729	
0.9	2	2	97.590	99.570	0.99454	
0.93	1	1	98.795	99.785	0.92001	
0.95	1	1	98.795	99.785	0.73116	

**Table E. 41** : Results from neural network that was trained to classify Subject #41.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	1	1	98.795	99.785	0.02507	
0.85	1	1	98.795	99.785	0.9596	
0.9	1	1	98.795	99.785	0.99587	
0.93	1	1	98.795	99.785	0.81282	
0.95	1	1	98.795	99.785	0.57351	

**Table E. 42** : Results from neural network that was trained to classify Subject #42.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	4	8	95.181	98.280	0.99059	
0.85	4	7	95.181	98.495	0.99375	
0.9	4	5	95.181	98.925	0.94734	
0.93	4	5	95.181	98.925	0.99059	
0.95	3	3	96.386	99.355	0.98068	

**Table E. 43 :** Results from neural network that was trained to classify Subject #43.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	5	8	93.976	98.280	0.03853	
0.85	4	5	95.181	98.925	0.91475	
0.9	3	4	96.386	99.140	0.97125	
0.93	3	4	96.386	99.140	0.00178	
0.95	2	2	97.590	99.570	0.97759	

**Table E. 44 :** Results from neural network that was trained to classify Subject #44.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	7	20	91.566	95.699	0.87579	
0.85	7	20	91.566	95.699	0.99863	
0.9	7	18	91.566	96.129	0.92784	
0.93	6	14	92.771	96.989	0.00186	
0.95	4	12	95.181	97.419	0.99794	

**Table E. 45 :** Results from neural network that was trained to classify Subject #45.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	5	97.590	98.925	0.99962	
0.85	2	5	97.590	98.925	0.99959	
0.9	1	3	98.795	99.355	0.99864	
0.93	1	2	98.795	99.570	0.00015	
0.95	1	1	98.795	99.785	0.01089	



**Table E. 46 :** Results from neural network that was trained to classify Subject #46.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	5	10	93.976	97.849	0.88563	
0.85	5	10	93.976	97.849	0.99594	
0.9	4	8	95.181	98.280	0.99247	
0.93	4	7	95.181	98.495	0.88963	
0.95	2	3	97.590	99.355	0.99427	

**Table E. 47 :** Results from neural network that was trained to classify Subject #47.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	4	7	95.181	98.495	0.97873	
0.85	3	6	96.386	98.710	0.99222	
0.9	2	5	97.590	98.925	0.99222	
0.93	2	3	97.590	99.355	0.9845	
0.95	1	1	98.795	99.785	0.98818	

**Table E. 48 :** Results from neural network that was trained to classify Subject #48.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	5	19	93.976	95.914	0.18207	
0.85	5	19	93.976	95.914	0.92783	
0.9	5	18	93.976	96.129	0.9931	
0.93	5	15	93.976	96.774	0.95409	
0.95	5	14	93.976	96.989	0.99309	

**Table E. 49** : Results from neural network that was trained to classify Subject #49.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.99059	
0.85	0	0	100	100	0.99617	
0.9	0	0	100	100	0.99582	
0.93	0	0	100	100	0.8934	
0.95	0	0	100	100	0.98882	

**Table E. 50** : Results from neural network that was trained to classify Subject #50.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	1	5	98.795	98.925	0.94712	
0.85	1	5	98.795	98.925	0.99456	
0.9	1	4	98.795	99.140	0.99473	
0.93	1	4	98.795	99.140	0.00275	
0.95	1	4	98.795	99.140	0.84574	

**Table E. 51** : Results from neural network that was trained to classify Subject #51.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	8	96.386	98.280	0.99619	
0.85	3	8	96.386	98.280	0.99849	
0.9	2	7	97.590	98.495	0.97295	
0.93	2	7	97.590	98.495	0.41808	
0.95	2	6	97.590	98.710	0.9219	

**Table E. 52** : Results from neural network that was trained to classify Subject #52.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	2	2	97.590	99.570	0.83576	0.9999
0.85	2	2	97.590	99.570	0.99917	0.9999
0.9	2	2	97.590	99.570	0.9973	0.9999
0.93	1	1	98.795	99.785	0.859	0.9999
0.95	1	1	98.795	99.785	0.99816	0.9999

**Table E. 53** : Results from neural network that was trained to classify Subject #53.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	3	11	96.386	97.634	0.8938	
0.85	3	11	96.386	97.634	0.96428	
0.9	2	10	97.590	97.849	0.9973	
0.93	2	10	97.590	97.849	0.99195	
0.95	2	7	97.590	98.495	0.99507	

**Table E. 54** : Results from neural network that was trained to classify Subject #54.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. ( %)	Subject's output(A)	Subject's output(B)*
0.8	2	5	97.590	98.925	0.99623	
0.85	2	5	97.590	98.925	0.99806	
0.9	2	5	97.590	98.925	0.99893	
0.93	1	3	98.795	99.355	0.92988	
0.95	1	3	98.795	99.355	0.96805	

**Table E. 55 :** Results from neural network that was trained to classify Subject #55.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	2	97.590	99.570	0.99749	
0.85	2	2	97.590	99.570	0.99345	
0.9	2	2	97.590	99.570	0.99456	
0.93	2	2	97.590	99.570	0.99637	
0.95	2	2	97.590	99.570	0.98713	

**Table E. 56 :** Results from neural network that was trained to classify Subject #56.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	1	1	98.795	99.785	0.92338	
0.85	0	0	100	100	0.9804	
0.9	0	0	100	100	0.99304	
0.93	0	0	100	100	0.98209	
0.95	0	0	100	100	0.9784	

**Table E. 57 :** Results from neural network that was trained to classify Subject #57.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	4	96.386	99.140	0.0807	
0.85	2	2	97.590	99.570	0.3164	
0.9	1	1	98.795	99.785	0.9795	
0.93	1	1	98.795	99.785	0.0018	
0.95	1	1	98.795	99.785	0.5886	

**Table E. 58 :** Results from neural network that was trained to classify Subject #58.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.98589	
0.85	0	0	100	100	0.99268	
0.9	0	0	100	100	0.99327	
0.93	0	0	100	100	0.99609	
0.95	0	0	100	100	0.97681	

**Table E. 59 :** Results from neural network that was trained to classify Subject #59.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	4	7	95.181	98.495	0.6402	
0.85	4	6	95.181	98.710	0.98956	
0.9	3	5	96.386	98.925	0.98439	
0.93	2	3	97.590	99.355	0.96314	
0.95	2	3	97.590	99.355	0.9911	

**Table E. 60 :** Results from neural network that was trained to classify Subject #60.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	0	0	100	100	0.964662	
0.85	0	0	100	100	0.99613	
0.9	0	0	100	100	0.99659	
0.93	0	0	100	100	0.98849	
0.95	0	0	100	100	0.99701	

**Table E. 61** : Results from neural network that was trained to classify Subject #61.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	7	96.386	98.495	0.98814	
0.85	3	6	96.386	98.710	0.99642	
0.9	2	5	97.590	98.925	0.99761	
0.93	2	5	97.590	98.925	0.46866	
0.95	2	5	97.590	98.925	0.95629	

**Table E. 62** : Results from neural network that was trained to classify Subject #62.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	6	97.590	98.710	0.99741	
0.85	2	6	97.590	98.710	0.99877	
0.9	2	6	97.590	98.710	0.99769	
0.93	2	6	97.590	98.710	0.99535	
0.95	2	6	97.590	98.710	0.1849	

**Table E. 63** : Results from neural network that was trained to classify Subject #63.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	10	97.590	97.849	0.97661	0.33631
0.85	2	9	97.590	98.065	0.99732	0.96179
0.9	2	9	97.590	98.065	0.99715	0.99058
0.93	2	9	97.590	98.065	0.86041	0.00016
0.95	2	6	97.590	98.710	0.99542	0.00033

**Table E. 64 :** Results from neural network that was trained to classify Subject #64.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	3	97.590	99.355	0.91373	
0.85	2	3	97.590	99.355	0.99547	
0.9	2	2	97.590	99.570	0.99761	
0.93	2	2	97.590	99.570	0.00674	
0.95	1	1	98.795	99.785	0.30109	

**Table E. 65 :** Results from neural network that was trained to classify Subject #65.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	8	96.386	98.280	0.97046	
0.85	3	8	96.386	98.280	0.97046	
0.9	3	8	96.386	98.280	0.95313	
0.93	3	8	96.386	98.280	0.97272	
0.95	3	8	96.386	98.280	0.98207	

**Table E. 66 :** Results from neural network that was trained to classify Subject #66.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	1	1	98.795	99.785	0.0008	
0.85	1	1	98.795	99.785	0	
0.9	1	1	98.795	99.785	0	
0.93	1	1	98.795	99.785	0	
0.95	1	1	98.795	99.785	0.0001	

**Table E. 67 :** Results from neural network that was trained to classify Subject #67.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	3	96.386	99.355	0.34073	
0.85	3	3	96.386	99.355	0.80963	
0.9	3	3	96.386	99.355	0.9997	
0.93	3	3	96.386	99.355	0.97357	
0.95	3	3	96.386	99.355	0.99981	

**Table E. 68 :** Results from neural network that was trained to classify Subject #68.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	8	97.590	98.280	0.95238	
0.85	2	8	97.590	98.280	0.98554	
0.9	2	8	97.590	98.280	0.98554	
0.93	2	8	97.590	98.280	0.95646	
0.95	2	7	97.590	98.495	0.97654	

**Table E. 69 :** Results from neural network that was trained to classify Subject #69.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	9	96.386	98.065	0.98743	
0.85	3	4	96.386	99.140	0.99563	
0.9	1	1	98.795	99.785	0.99559	
0.93	1	1	98.795	99.785	0.98725	
0.95	1	1	98.795	99.785	0.99356	



**Table E. 70** : Results from neural network that was trained to classify Subject #70.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	10	24	87.952	94.839	0.13183	
0.85	9	21	89.157	95.484	0.98152	
0.9	9	20	89.157	95.699	0.99924	
0.93	8	17	90.361	96.344	0.99727	
0.95	8	17	90.361	96.344	0.99547	

**Table E. 71** : Results from neural network that was trained to classify Subject #71.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	11	24	86.747	94.839	0.99074	
0.85	8	18	90.361	96.129	0.99219	
0.9	7	15	91.566	96.774	0.99319	
0.93	4	12	95.181	97.419	0.9161	
0.95	4	9	95.181	98.065	0.98903	

**Table E. 72** : Results from neural network that was trained to classify Subject #72.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	5	96.386	98.925	0.99616	
0.85	2	3	97.590	99.355	0.99655	
0.9	1	1	98.795	99.785	0.99919	
0.93	1	1	98.795	99.785	0.996	
0.95	1	1	98.795	99.785	0.99767	

**Table E. 73 :** Results from neural network that was trained to classify Subject #73.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	2	5	97.590	98.925	0.23573	
0.85	1	4	98.795	99.140	0.97484	
0.9	1	1	98.795	99.785	0.60022	
0.93	1	1	98.795	99.785	0.97396	
0.95	0	0	100	100	0.83456	

**Table E. 74 :** Results from neural network that was trained to classify Subject #74.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	5	16	93.976	96.559	0.99857	
0.85	4	13	95.181	97.204	0.97231	
0.9	4	12	95.181	97.419	0.99467	
0.93	4	11	95.181	97.634	0.99527	
0.95	4	11	95.181	97.634	0.99714	

**Table E. 75 :** Results from neural network that was trained to classify Subject #75.

Cut-off	Person rec.	Face rec.	Person rec. (%)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	3	96.386	98.795	0.12117	
0.85	3	3	96.386	98.795	0.99873	
0.9	3	3	96.386	98.795	0.91212	
0.93	3	3	96.386	98.795	0.00477	
0.95	1	1	98.795	99.785	0.9776	

**Table E. 76** : Results from neural network that was trained to classify Subject #76.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	10	27	87.952	94.194	0.98417	
0.85	8	25	90.361	94.624	0.99102	
0.9	7	22	91.566	95.269	0.99149	
0.93	6	19	92.771	95.914	0.97977	
0.95	5	17	93.976	96.344	0.94922	

**Table E. 77** : Results from neural network that was trained to classify Subject #77.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	6	15	92.771	96.774	0.96172	
0.85	5	13	93.976	97.204	0.99342	
0.9	5	12	93.976	97.419	0.9949	
0.93	4	10	95.181	97.849	0.20578	
0.95	4	10	95.181	97.849	0.99305	

**Table E. 78** : Results from neural network that was trained to classify Subject #78.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	7	17	91.566	96.344	0.9987	
0.85	7	17	91.566	96.344	0.99909	
0.9	6	15	92.771	96.774	0.99946	
0.93	4	13	95.181	97.204	0.99946	
0.95	4	13	95.181	97.204	0.9989	

**Table E. 79** : Results from neural network that was trained to classify Subject #79.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	5	96.386	98.925	0.9904	
0.85	1	3	98.795	99.355	0.999	
0.9	1	2	98.795	99.570	0.9994	
0.93	1	2	98.795	99.570	0.10097	
0.95	1	2	98.795	99.570	0.99872	

**Table E. 80** : Results from neural network that was trained to classify Subject #80.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	3	10	96.386	97.849	0.9282	
0.85	3	10	96.386	97.849	0.99971	
0.9	3	7	96.386	98.495	0.99979	
0.93	2	6	97.590	98.710	0.99771	
0.95	1	5	98.795	98.925	0.94779	

**Table E. 81** : Results from neural network that was trained to classify Subject #81.

Cut-off	Person rec.	Face rec.	Person rec. ( %)	Face rec. (%)	Subject's output(A)	Subject's output(B)*
0.8	8	17	90.361	96.344	0.99171	
0.85	5	14	93.976	96.989	0.99009	
0.9	4	11	95.181	97.634	0.99177	
0.93	4	9	95.181	98.065	0.98524	
0.95	4	7	95.181	98.495	0.98192	

**Table E. 82:** Table showing analysis of data for Person recognition. (True Negative Recognition Rates).

Threshold	Mean Person rec.	Std Person rec.	Mean Person rec. %	Std Person rec. %
0.1	10.148	6.616	87.803	7.963
0.2	7.988	5.562	90.406	6.685
0.3	6.84	4.771	91.789	5.728
0.4	6.148	4.328	92.622	5.187
0.5	5.506	3.86	93.396	4.632
0.6	4.938	3.483	94.08	4.17
0.7	4.321	3.016	94.824	3.607
0.8	3.654	2.703	95.627	3.23
0.9	3.198	2.462	96.162	2.952
0.93	2.765	2.331	96.683	2.803
0.95	2.506	2.186	96.995	2.627

**Table E. 83:** Table showing analysis of data for Person recognition. (True Negative Recognition Rates)

Threshold	Mean Person rec.	Std Person rec.	Mean Person rec. %	Std Person rec. %
0.1	24.901	19.052	94.671	4.083
0.2	19.469	15.593	95.829	3.343
0.3	16.358	12.812	96.498	2.742
0.4	14.519	11.282	96.894	2.408
0.5	13.111	10.494	97.196	2.24
0.6	11.667	9.526	97.507	2.029
0.7	10.198	8.446	97.823	1.796
0.8	8.716	7.444	98.142	1.583
0.9	7.765	6.847	98.343	1.457
0.93	6.58	6.275	98.598	1.339
0.95	5.679	5.556	98.792	1.181

**Table E. 84:** Table of values for Mean True Positive Recognition Rate – Excluding extra views.

Threshold	Mean True Positive Rec. Rate – Excl. extra views
0.1	87.078
0.2	84.115
0.3	82.634
0.4	81.646
0.5	80.288
0.6	79.3
0.7	78.313
0.8	77.572
0.9	75.226
0.93	71.029
0.95	65.514

**Table E. 85:** Table of values for Mean True Positive Recognition Rate – Including extra views.

Threshold	Mean True Positive Rec. Rate – Incl. extra views
0.1	91.481
0.2	88.189
0.3	86.461
0.4	85.103
0.5	83.745
0.6	82.757
0.7	81.77
0.8	80.905
0.9	78.23
0.93	73.663
0.95	68.025

**Table E. 86:** Table of Mean and Std. of output neuron

	Excluding Extra Views	Including Extra Views
Mean	0.8382	0.7962
Std		



## APPENDIX F

This appendix describes the results of the tests performed with the counter-propagation network. The results are presented in a tabular format. A brief explanation of the headings of the tables are first given:

- $\alpha$  - Counter-propagation network training parameter (See Chapter 5.)
- $\beta$  - Counter-propagation network training parameter (See Chapter 5.)
- Training cycles - Number of cycles during which training of the network was done.
- Hidden Units - The number of hidden units that were inserted by the DDA for each test.
- Min. O/P - This is the minimum value that was classified correctly as a true positive.
- No. of Subject - This is the number of subjects that were classified incorrectly (false positives).
- Class. Rate - This is the classification rate, as a percentage expression of the faces that were classified correctly. The true positive percentage is calculated as follows :

$$100 - 100 * \left( \frac{x}{y} \right) \quad \text{F- 1}$$

where x is the number of faces that were classified incorrectly and y is the total number of faces in the pattern set.



Table F. 1 : Results from counterpropagation network

$\alpha$	$\beta$	Train. Cycles	No. of Hidden Units	Min. Output	No. of Subject	Class. Rate %	SSE.	MSE.
0.1	0.1	100	50	0.19655	276	40	312.5059	0.67936
0.1	0.1	100	50	0.13037	284	38.26087	313.2054	0.68088
0.1	0.1	100	50	0.20108	283	38.47826	317.9983	0.6913
0.1	0.1	100	100	0.2511	201	56.30435	252.06	0.54796
0.1	0.1	100	100	0.21123	210	54.34783	257.597	0.55999
0.1	0.1	100	100	0.251089	196	57.3913	247.4556	0.53795
0.1	0.1	100	150	0.333029	166	63.91304	249.7977	0.54304
0.1	0.1	100	150	0.25134	149	67.6087	230.5138	0.50112
0.1	0.1	100	150	0.25075	162	64.78261	230.9796	0.50213
0.1	0.1	100	200	0.25059	124	73.04348	228.7678	0.49732
0.1	0.1	100	200	0.25123	124	73.04348	243.9461	0.53032
0.1	0.1	100	200	0.33236	128	72.17391	225.047	0.48923
0.1	0.1	100	250	0.33387	120	73.91304	272.0616	0.59145
0.1	0.1	100	250	0.33431	106	76.95652	259.4758	0.56408
0.1	0.1	100	250	0.20138	109	76.30435	266.1243	0.57853
0.1	0.1	100	300	0.33494	97	78.91304	318.1378	0.6916
0.1	0.1	100	300	0.33459	95	79.34783	320.7321	0.69724
0.1	0.1	100	300	0.33445	89	80.65217	319.8936	0.69542
0.1	0.1	100	350	0.33541	79	82.82609	381.7019	0.82979
0.1	0.1	100	350	0.33659	78	83.04348	375.3324	0.81594
0.1	0.1	100	350	0.33659	78	83.04348	375.3324	0.81594
0.1	0.1	100	400	0.50225	64	86.08696	440.0111	0.95655
0.1	0.1	100	400	0.50176	65	85.86957	444.6306	0.96659
0.1	0.1	100	400	0.33541	72	84.34783	448.2296	0.97441
0.1	0.1	100	450	0.50228	57	87.6087	520.703	1.13196
0.1	0.1	100	450	0.5018	58	87.3913	520.6974	1.31974
0.1	0.1	100	450	0.50337	55	88.04348	520.91	1.13198
0.1	0.1	100	500	0.89708	55	88.04348	392.5742	0.85342
0.1	0.1	100	500	0.85743	55	88.04348	118.9694	0.2585
0.1	0.1	100	500	0.8441	55	88.04348	65.12747	0.14158

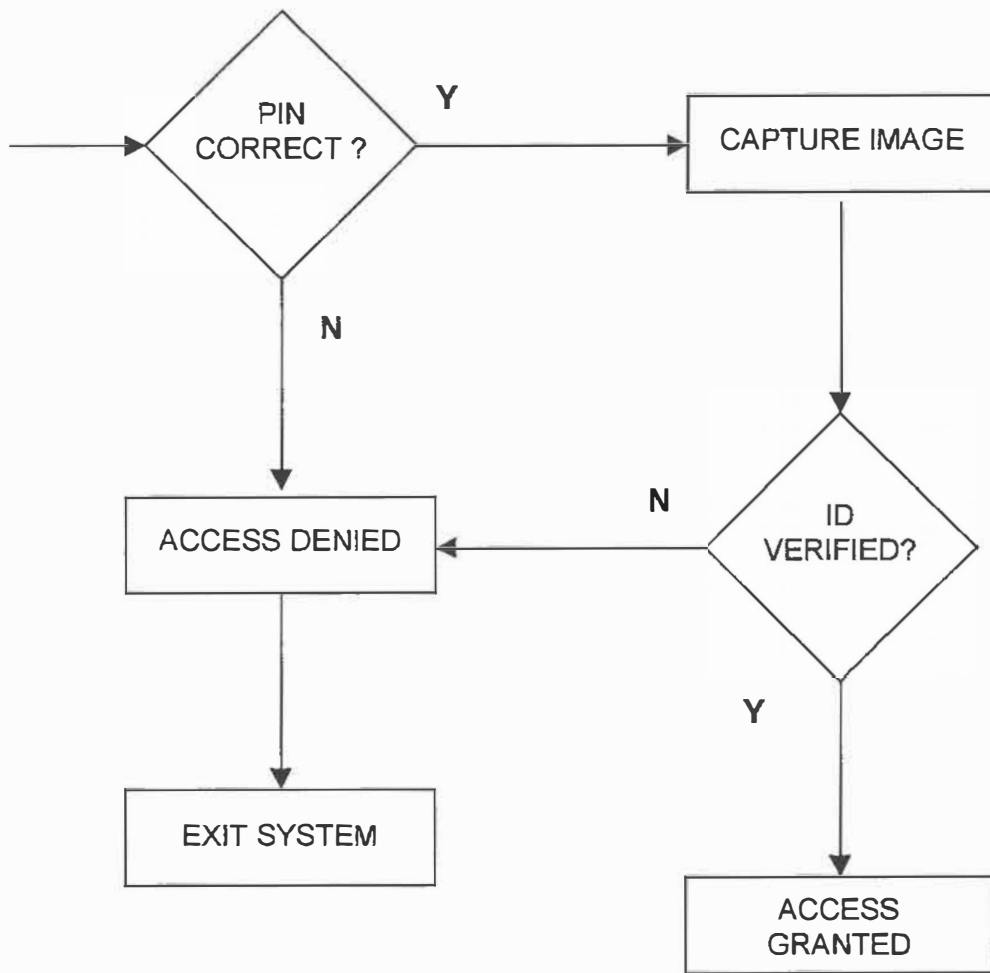
## APPENDIX G

---

The face recognition algorithm (FRA) described in the main body of this thesis would work well in an environment where a subject's identity needs to be verified. Assume that the algorithm will be implemented in a security system where the first level of access would be the subject's PIN (e.g. ATM application). Hence, in such a system, the subject's PIN would be used in addition to the FRA for subject verification. The enhanced security system could be described in flow diagram form as depicted in Figure G.1.

Before describing how the FRA would be implemented into the ATM security system, it is necessary to define the type of card that the subject would use, as well as its implications. The FRA uses neural networks as the core verification principle, and hence it is necessary to store the artificial neural network (ANN) weights of the valid subjects somewhere.

One option is to store the weights on a smart card/ATM card. Thus, the weights could be read, passed to a neural network (preferably processing takes place at the ATM), and verification can take place. This type of implementation would utilise the "neural network per person" approach. The neural network is trained to verify the valid subject (i.e. the owner of the ATM card) and reject all impostors. Although the high true negative and high true positive recognition rates associated with this approach are quite appealing, the disadvantage is that a separate network has to be trained for each subject. This may not be viable if the number of valid subjects is large. However, it could be accomplished if "batch training" (a facility offered by SNNS) was performed. The flow diagram which describes this approach is depicted in Figure G.2.



**Figure G. 1 :** High-level flow diagram of FRA integrated with level-one PIN access security system.

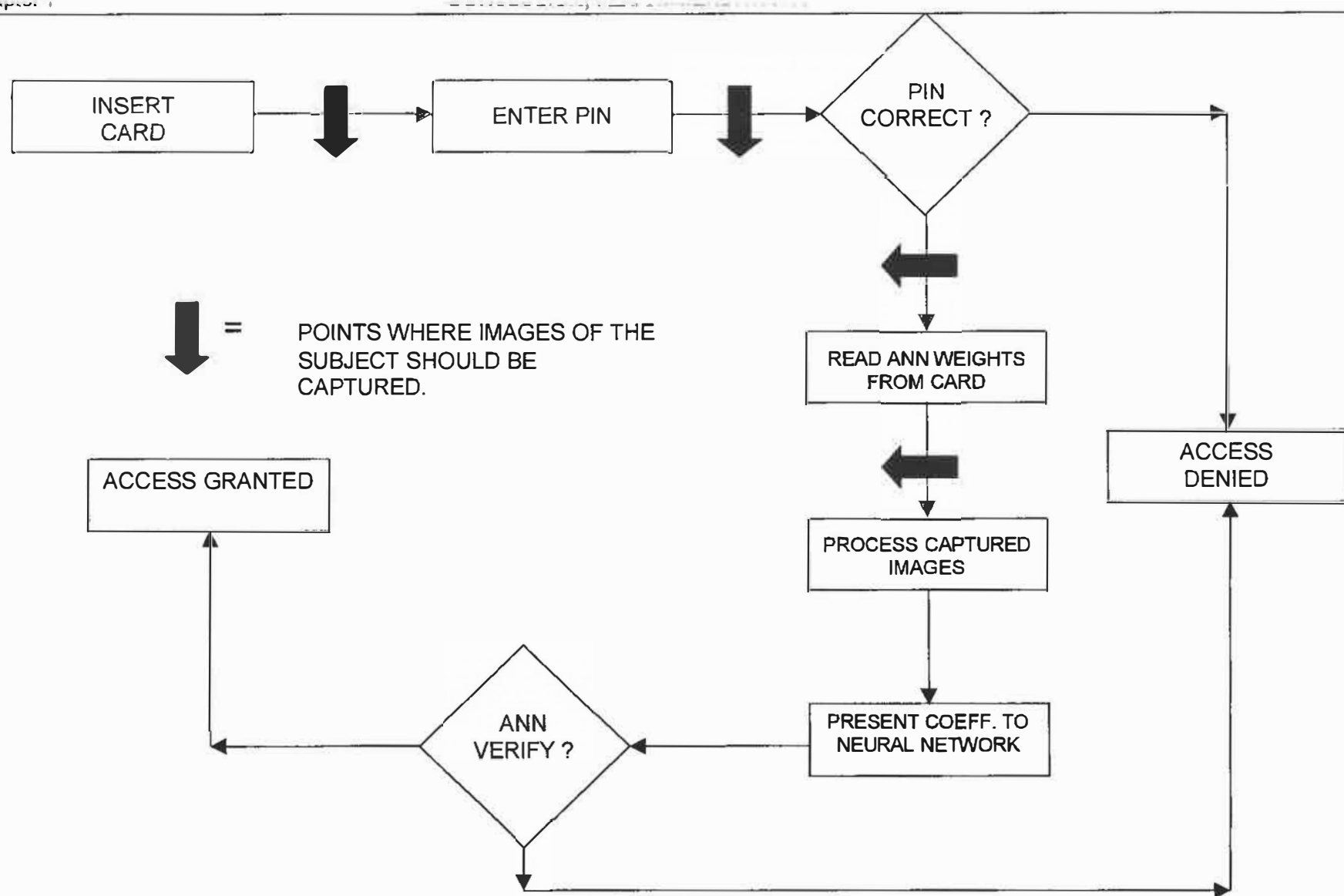
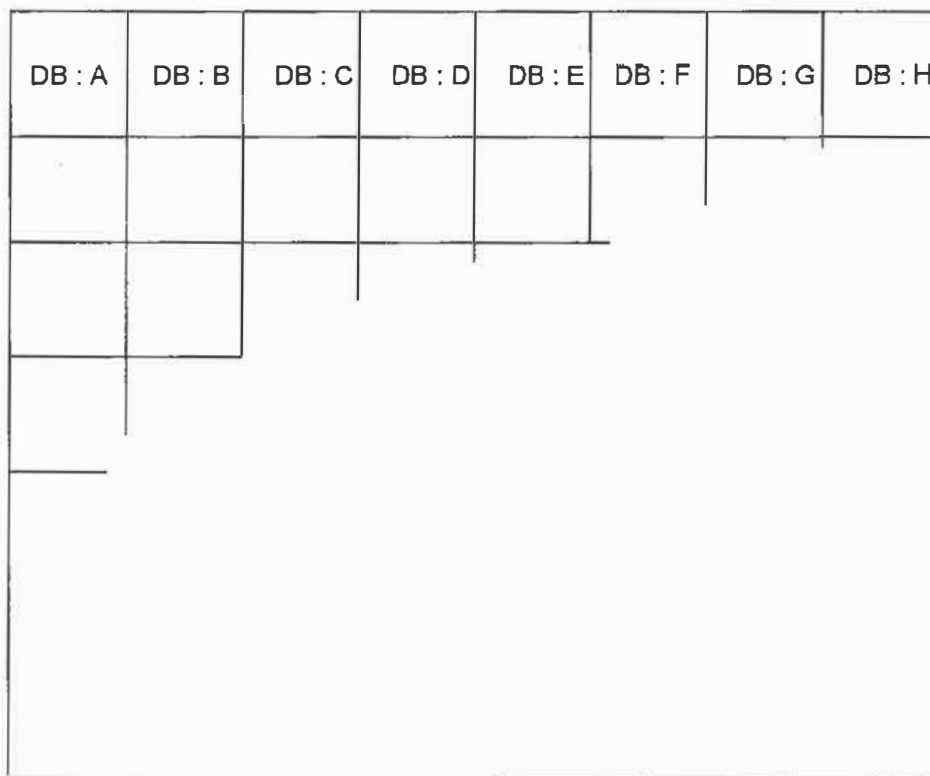


Figure G. 2: High-level flow diagram of security system using smart card to store weights of the ANN.

The second option is to store the subject's weights in a database of weights, on some remote server. The valid subject's ANN weights can be accessed from the database via the subject's PIN. The flow diagram that describes this approach is much like that of Figure G.2.

The large number of valid subjects to be trained can be exploited to the advantage of the security system. It was observed for split databases in Chapter 6 (i.e. training on part of database 1 and testing on the remainder of database 1 and all of database 2), that the "unseen" database produced high true negative recognition rates.

The large training database could be split into smaller databases as follows:



**Figure G. 3** : Depiction of small databases within one large database.

The valid subjects from each database can only be recognised in their database i.e. if subject  $n$  belongs to database A, then subject  $n$  would be verified correctly in database A and not (within some acceptable error) in any of the other databases. However, the

optimum number of subjects in each database requires some investigation, and was not completely studied in this research due to subject availability. This method of training can be likened to the scope of variables in a function. The variables can be seen within the function, but not outside the function.

The general symmetry transform was proposed as a possible solution for automatic face location, but it was found that it was computationally intensive. This provides a source for further work to be done in this particular field. Face location deserves a study dedicated to it alone. It is easy for humans to be able to take into account varying light levels, rotation, various backgrounds etc, and still perform effective face recognition. However it is difficult to reproduce this process on a computer.

This thesis marks an initial study into automatic face recognition. It was shown the two dimensional discrete cosine transforms and neural networks do form a viable solution to the problem of automatic face recognition. However, it is not the optimal solution. With the advent of faster and more powerful processors techniques such as elastic graph matching and hidden markov models a more realistic automatic face recognition algorithm could be implemented.