WEIGHTLESS NEURAL NETWORKS FOR FACE AND PATTERN RECOGNITION: AN EVALUATION USING OPEN-SOURCE DATABASES

by:

KAZIMALI M. KHAKI

A thesis submitted for the degree of Doctor of Philosophy

School of Engineering & Design

Brunel University, London

UNITED KINGDOM

August 2013

Dedicated to my family

ABSTRACT

The interface with the real-world has proved to be extremely challenging throughout the past 70 years in which computer technology has been developing. The problem initially is assumed to be somewhat trivial, as humans are exceptionally skilled at interpreting real-world data, for example pictures and sounds. Traditional analytical methods have so far not provided the complete answer to what will be termed pattern recognition.

Biological inspiration has motivated pattern recognition researchers since the early days of the subject, and the idea of a neural network which has self-evolving properties has always been seen to be a potential solution to this endeavour. Unlike the development of computer technology in which successive generations of improved devices have been developed, the neural network approach has been less successful, with major setbacks occurring in its development. However, the fact that natural processing in animals and humans is a voltage-based process, devoid of software, and self-evolving, provides an on-going motivation for pattern recognition in artificial neural networks.

This thesis addresses the application of weightless neural networks using a ranking pre-processor to implement general pattern recognition with specific reference to face processing. The evaluation of the system will be carried out on open source databases in order to obtain a direct comparison of the efficacy of the method, in particular considerable use will be made of the MIT-CBCL face database. The methodology is cost effective in both software and hardware forms, offers real-time video processing, and can be implemented on all computer platforms. The results of this research show significant improvements over published results, and provide a viable commercial methodology for general pattern recognition.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

The unique experience of delving into a specific research field during the PhD would not have been completed if it was not for some people, and I would like to thank them for their help throughout this research.

First of all I would like to express my gratitude to Prof. John Stonham, who was a constant source of inspiration throughout my years at Brunel, without whom I would never have imagined pursuing a PhD. I will surely miss our brainstorming sessions, your seemingly infinite knowledge, and your lectures on my smoking.

A special thank you to Dr. Ian Dear, your optimism and helpfulness during our fairly regular talks always boosted my confidence, and reassured me that I could accomplish whatever I had set out to do.

During the course of the research, many image databases were used to test the methodology presented, and thanks go to the AT&T Laboratories Cambridge for the AT&T Database, Markus Weber and the California Institute of Technology for the Caltech Faces1999 Database, Ara V. Nefian and Georgia Tech for the Georgia Tech Face Database, Dr. Libor Spacek and Essex University for the Essex Faces94, Faces95, Faces96 and Grimace Databases.

Thanks also to Yale University for the use of Yale Face Database B, University of Massachusetts for Labeled Faces in the Wild, Michael Lyons, Miyuki Kamachi and Jiro Gyoba for the Japanese Female Facial Expressions Database. Portions of the research in this paper use the FERET database of facial images collected under the FERET program, sponsored by the DOD Counterdrug Technology Development Program Office, and credit is hereby given to the Massachusetts Institute of Technology and to the Centre for Biological and Computational Learning for providing the database of facial images

Last but not least, I would like to thank the Almighty for giving me this opportunity, my parents for their constant encouragement, and my wife for her unwavering support. I would also like to thank my siblings for being the subjects in the various image databases I produced throughout the course of this research. The least I can do in return is to dedicate this work to them.

Mr. Kazimali M. Khaki

August 2013

Chapter 1 :

Introduction

1. INTRODUCTION

Pattern recognition is a task performed by humans countless number of times on a daily basis, be it object recognition, scene recognition, face recognition, speech recognition, etc. Research has been conducted for decades into pattern recognition, due to a multitude of possible applications, such as in the entertainment industry – in video games, computer interaction, robot design, etc., and the security sector – driving licence and passport photo identification, CCTV video surveillance, etc. (Chellappa, Wilson and Sirohey, 1995)

Even though humans can accurately detect and recognise patterns in images, it has not been fully understood as to how this recognition occurs. Although multiple attempts have been made to design accurate and efficient recognition and detection systems, and multiple systems that have been designed in the past have been implemented commercially (FaceKey, 2013; Cognitec, 2013), a single solution that can operate accurately regardless of illumination, viewing distance, or image noise has yet to be designed.

The term pattern recognition is accurately defined by Bremermann, where he says:

> "Any pattern recognition task is concerned with a collection of objects or possible objects such as speech sounds, characters, etc. The collection of all possible objects is the 'universe of discourse', or the 'universe' for short. The universe need not be restricted to characters or sounds, but can be almost anything: moves in a chess game, mathematical formulas, cloud pictures taken by a weather satellite, medical symptoms, fingerprints, etc. A 'pattern' is a 'subset' of the universe. For example, a pattern could be all possible appearances of the letter 'A', all speech sounds 'oh', all 'good' moves in chess, all mathematical formulas that are true, all cloud pictures that indicate a hurricane, all medical symptom combinations that indicate heart disease, the fingerprints of a person wanted for a crime, etc. An 'object' is a particular 'element' of the universe, for example a particular hand-written character, a particular speech sound, a particular chess move, etc." (Bremermann, 1971)

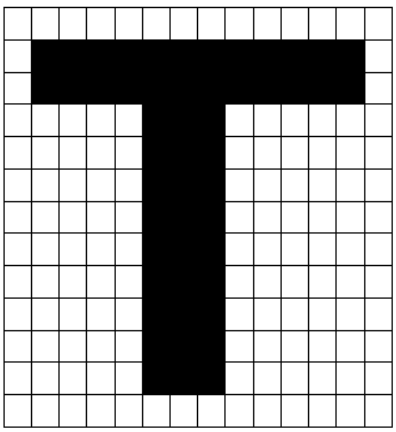


Figure 1.1 - Sample 16x16 pixel binary image

The main problem with any image recognition task is the enormity of the pattern space being processed. Take for example the simple 16x16 pixel binary image displaying the letter 'T', shown in Figure 1.1. This simple image contains 256 pixels, where each pixel has only 2 possible colours - black or white. The total number of possible patterns in the universe is 2^{256}, which is approximately 10^{80}. Given that the number of electrons in the universe is approximated to be 10^{80}, it brings into focus the enormity of the universe of possible patterns that can occur in an image with the technology available today, where the average individual has a mobile phone with a 3 million colour pixel camera inbuilt!

It is important to note that there are three main aspects to face recognition:

1. Face Detection – when the location of a face in an image is detected.
2. Face Recognition – when the image containing the face is tested against multiple other face 'classes' in order to classify it; this is a 1-to-many relationship.
3. Face Verification – when an image containing the face is tested against only one 'class' in order to check whether it belongs to it; this is a 1-to-1 relationship.

Face detection is normally used as a pre-processing step to face recognition systems, and the output data from this algorithm is then used in the recognition or verification stages. Although face verification and face recognition seem similar, the verification stage presumes prior knowledge of the subject being tested on, an example being where for an entry system a user has to first scan his ID card, and then have his face verified – because of the information gained from the ID card, the systems knows to check the face image captured against only the user's 'class'. If a face recognition system was to be used in the same scenario, the ID card would not be used, and after capturing the user's face, the system would check that data against all the 'classes' it had in its database.

Much research has been conducted into mathematical and statistical solutions in an attempt at designing an automatic or semi-automatic face recognition system, however as mentioned before no system has been designed that can match the accuracy and efficiency of the human brain. Although much progress has been made in the statistical approaches to pattern recognition, we believe the statement made by Bremermann in 1971 still stands to reason, when he says:

> "I believe that analogously formal structures exist that govern pattern recognition problems. This is a consequence of the fact that the problems are expressible in mathematical terms. While therefore a formal structure governing pattern recognition exists the same as the set of all theorems of a complete mathematical theory exist, the existence does not imply knowledge of the structure or know-ability of the structure." (Bremermann, 1971)

He continues in the same paper saying:

> "Consequently pattern recognition problems can be solved neither by unlimited brute force processing, nor can the theory of pattern recognition algorithms be explored in this way. What remains is the creative

> imagination of researchers and the examples of nature that demonstrate what is possible." (Bremermann, 1971)

The research presented in this thesis is a pattern/face recognition algorithm that is based on digital neural networks, and can be implemented as a video scene filter, face verification or face recognition system, etc. Modern technology allows a direct interface between the camera and a digital neural network, with information being captured at a pixel level, and this allows the network to remove the problematic statistical pre-processing stages, focussing on emulating the human brain in image pattern recognition.

1.1 Motivation and objectives

There has been increasing interest from the general public and the scientific community in face recognition for high security applications. Mathematical analysis of images can only achieve a certain level of accuracy, and:

> "The only system that does seem to work well in the face of these challenges is the human visual system. It makes eminent sense, therefore, to attempt to understand the strategies this biological system employs, as a first step towards eventually translating them into machine-based algorithms." (Sinha *et al.*, 2006)

The aim of the research was to gain inspiration from the behaviour of biological systems in designing a self-evolving fast and efficient pattern recognition system that can operate in real-time and in a real-world scenario, with high accuracy rates in order to be viable for high security applications. The system was also required to have a low computational intensity and storage cost, in order to be feasible for commercial applications.

The objectives of this thesis are:

- To discuss the history behind neural network-based pattern/face recognition
- To present the designed algorithm, and investigate the various factors that can be used to optimise the system according to its implementation requirements.
- To investigate the accuracy of the system in video scene filtering and face recognition scenarios.
- To investigate the effect of variations in illumination, viewing distance, and image size on the accuracy of the system.

To test the accuracy of the system, open source databases will be used in order to make direct comparisons with results obtained by competing methodologies, together with video recordings, and the thesis will also discuss the various security implementations of the algorithm itself.

1.2 Thesis overview

To fulfil the above objectives, the chapters presented in this thesis are as follows:

Chapter 2 contains a review of the various techniques that have been used in the development of face recognition systems. Statistical techniques are discussed, and the history of analogue and digital neural networks is explained. The advantages and disadvantages of these various techniques are also discussed, and a resulting explanation as to why digital neural networks were used to develop the methodology presented in this thesis.

Chapter 3 introduces the system overview, and discusses the three main stages of the system – the data extraction, the function calculation, and the output processing. The neuron function storage requirements are also discussed, as is the implementation of the methodology in software, and the platforms that have been used to test it.

Chapter 4 presents the weightless neural network methodology being applied as an image filter for video scene recognition. The method of presentation of results in this thesis is discussed, with regards to the graphs that will be used, and the values that will be presented in this chapter and in the later chapters. To prove video scene filtering can be achieved by this methodology, a simple test is run on a video taken of a room, and a more complicated test is run on a news broadcast video downloaded from the internet.

Chapter 5 presents the methodology as a face recognition system, and therefore initially a review is performed of various image databases that are available for research purposes. From these databases, five are chosen to test the system, and a standardised system is chosen to run all the tests on. The results of each of these five tests is presented and discussed in detail.

Chapter 6 is an overview of the various aspects of the system that can be optimised, such as data extraction methods, and the neuron size. Theories are presented as to the effect of the different methods and sizes that can be used, and tests are performed on the same image dataset, in order to achieve a valid comparison of the results. The quality of each of the databases is also discussed, as is the effect of the number of images in the training set.

Chapter 7 discusses the effect that image manipulation has on the accuracy of the system, such as a change in image size using various resizing methods, and subject datasets which were created with variations in zoom and lighting conditions in order to test the robustness of the methodology.

Chapter 8 discusses the security aspects of the methodology, and presents an overview of the security of the system itself, and the subject data that the system ‘stores’. Tests are performed to prove the security of the system, and proposals are made as to aspects that can be included in a commercial implementation of the methodology presented in this thesis.

Chapter 9 concludes the various chapters that have been discussed in this thesis, and provides a summary of the various problems encountered in face recognition, and how the methodology presented can overcome these problems. Future developments using this technology are also identified.

Chapter 2 :

Literature Review

2. LITERATURE REVIEW

2.1 Introduction

Weightless neural network systems form the basis of this research. The principle application is face recognition, however the methodology is a general one, and so it can be used for video scene image filtering, face detection, and any image-based operation where there is evidence of recurring local features.

In this chapter, neural networks are critically reviewed, and their application to face recognition is assessed. It will be shown that weightless neural network-based systems have distinct advantages in terms of training and speed, cost effectiveness, accuracy, and are easier to implement in software and hardware than other methods.

Face recognition systems can be divided into two stages, face detection and face recognition. The research presented focuses on the recognition stage of the process, however face detection is a pre-processing stage of the application. It will be shown that although weightless neural networks have a potential for face detection, it is not the main focus of this thesis.

The review will consider the various statistical techniques used to solve automated pattern recognition, and then an introduction to neural network systems will be presented, along with the history of the first analogue systems. The inception of the first weightless neural network methodology will then be discussed, leading to the WISARD system (Aleksander, Stonham and Wilson, 1974), and the attempts to build algorithms based on it. As the methodology presented in Chapter 3 is also based upon a texture mapping algorithm, the history of this will also be discussed later in this section.

2.2 Pattern recognition techniques

In the past, template matching, syntactic approaches, statistical and neural networks (Jain, Duin and Jianchang Mao, 2000) have been used in the design of pattern recognition systems. However due to researchers combining these techniques to try and design an optimized solution, it is difficult to accurately classify algorithms solely by the technique that is used as some algorithms fall into more than one category (Zhao *et al.*, 2003).

Although methods of categorization based on psychological studies have been suggested for face detection and recognition systems (Zhao *et al.*, 2003), the most frequently encountered techniques are either statistical approaches, or neural network methods. The statistical approach covers correlation, feature extraction, or rule-based design, and the neural network approach can either be analogue weighted, or logical weightless models, which is the principle methodology used in this thesis.

2.2.1 Statistical approaches

Template matching is the one of the simplest and earliest methods of classifying images, where standard patterns provide the features of this category, and the sub-patterns extracted from the test images are compared to the original patterns using distance classifiers. The problem with this technique is that if the test image varies in scale or shape with regards to the stored image, poor correlations are achieved (Ming-Hsuan Yang, Kriegman and Ahuja, 2002; Sakai, Nagao and Fujibayashi, 1969). Further research was undertaken to create deformable templates, and although improved results were achieved (Yuille, Cohen and Hallinan, 1989), the method still suffered from the difficulty of initializing the deformable template "in the proximity of the object of interest" (Ming-Hsuan Yang, Kriegman and Ahuja, 2002).

The single patterns that are used in template matching provide features, which are numerical values, and combining numerical values produces vector-based approaches. The simplest possible patterns are in binary; however because of the complexity of images, even at low resolution the pattern space of a binary image is huge, and therefore it is difficult to extract the features, to produce the vectors, to calculate the numerical distance.

Numerical values have analogue variations, which result in a perceived linear sense of similarity, however if the image pattern is in binary, there are only two possibilities, black or white, which means only 2 decisions per pixel. However, the n-tuple method discussed later in this chapter was an inspiration, where the possibilities increase by processing more than one pixel at a time.

Numerous methodologies that are based on feature extraction have been created over the years based on Principal Component Analysis (Zhao *et al.*, 2003), Eigenfaces (Turk and Pentland, 1991), feature-line base methods (Li and Lu, 1999), and Fisherfaces (Chengjun Liu and Wechsler, 2001; Belhumeur and Kriegman, 1996).

Eigenfaces are used for both face detection and face recognition, and are primarily made up of eigenvectors. These eigenvectors "can be thought of as a set of features that together characterise the variation between face images" (Turk and Pentland, 1991), and are basically the "principal components of the distribution of faces" (Turk and Pentland, 1991). Although the Eigenface method results do not change with variations in illumination, a small change in the size of the face results in a significant decrease in the accuracy of the system; the major problem with this theory is the training of the system, where much time must be taken to optimise the eigenvectors (Chengjun Liu and Wechsler, 2001) used to build the Eigenfaces, in order to achieve high accuracy rates.

The nearest feature line method is built on the assumption that there are "distinct prototype feature points available in each class". A linear model is used on a feature space (Eigenface space), and a linear model is used to "interpolate and extrapolate each pair of prototype feature points belonging to the same class", where each feature point is a vector of weights in the space (Lu 1999). It should be noted that Eigenfaces are used as the start-point representation, and therefore the accuracy of the system greatly depends on the optimisation of the Eigenfaces.

Once the prototype feature points are extracted, they are generalised by the feature line, which is a line passing through the two points. This is assumed to approximate the variations in pose, illuminations and expression, theoretically enabling the system to generalise enough to recognise images of the subject in conditions that have not been trained on.

Aside from the method being time consuming and computationally intensive, the system has multiple requirements like an optimised Eigen space, or the assumption that at least two prototypes exist in every class.

Fisherfaces are built up of the resulting vectors when Linear Discriminant Analysis is used to find the subspace representation of a set of face images, where it is argued that since the learning set is labelled, it is better to use class specific linear methods to "shape the scatter in order to make it more reliable for classification" (Belhumeur, Hespanha and Kriegman, 1997).

Although experimental results displays improved accuracy when compared to the Principal Component Analysis approach, with variation in lighting, the algorithm only achieves comparatively lower error rates if the images used in testing are very similar to the images that have been trained on with regards to facial expression, etc.

It should also be noted that when this method is applied on the entire face, the entire mouth region is mainly ignored, as slight changes in facial expression result in a great change in the mouth position, shape, texture, and size.

"Independent component analysis is a generalization of principal component analysis, which decorrelates the high-order moments of the input in addition to the second order moments" (Stewart Bartlett, Lades and Sejnowski, 1998). ICA results in a more powerful data representation than PCA, however results achieved by different researches carried out when attempting to comparing PCA and ICA have been contradictory (Draper *et al.*, 2003). Regardless of this, the problem with this algorithm is that due to the higher order calculations being used, the system has a high computational intensity requirement, as well as requiring various image pre-processing techniques to be implemented.

Although the statistical approach has been used to design a number of commercial pattern recognition systems, the computational intensity required by such systems is high (Cognitec, 2013; Human Recognition Systems, 2013). Statistical approaches are also plagued by the "curse of dimensionality", which is a term coined by Bellman in 1961 (Bellman, 1961). Whilst it can be theorised that more features result in a better system performance, more features require a higher number of training samples (Jain, Duin and Jianchang Mao, 2000). Bellman observed that "the number of data points needed to sample a space grows exponentially in its dimensionality" (Priddy and Keller, 2005).

Due to the above reasoning, it is proposed that a simpler, optimized algorithm can be designed using the neural network approach.

2.2.2 Neural network methods

A 'neural network' is used to describe a system that is made up of many interconnected cellular processors. These individual elements are called 'neurons', as they are designed to mimic the neurons contained inside the human brain (Figure 2.1), where each individual neuron has a set of voltage inputs on which it performs a set of functions, and then outputs a voltage output (McCulloch and Pitts, 1943); a neuron can therefore be called an analogue voltage processor.

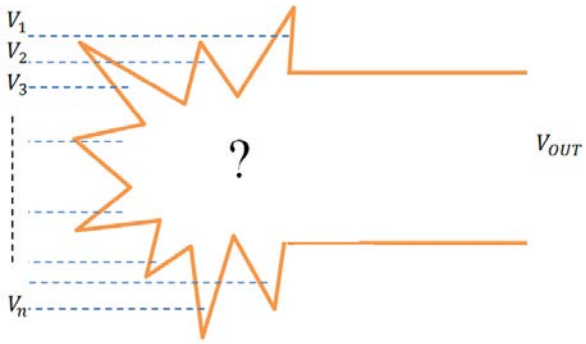


Figure 2.1 - The neuron - an analogue voltage processor

These 'neurons' are also called 'adaptable nodes' (Aleksander and Morton, 1995) as they are designed to 'adapt' or 'learn' based on the data provided. This 'learning' takes place when the system is in the training phase, and the accuracy of the system is determined when testing data is applied to it during the testing phase.

Neural network systems can be divided into two types, analogue systems (also called weighted neural networks) and digital systems (called weightless).

2.2.2.1 Difference between analogue and digital neural networks

Analogue patterns are perceived by hyper surface, multi-dimensional planes. Most analogue neural networks have an additional input to each neuron called the 'weight'. The training method used to train analogue neural networks is called 'supervised learning', where the system "requires an external source of information in order to adjust the network." (Kawaguchi, 2000) When training the system to recognise a specific class of images, the data input into the system consists of both types of images, those that require an overall positive response as they belong to the class, and other images that do not belong to the class, and hence require a negative response. When the output achieved by the system is wrong, the relevant 'weight' value is adjusted, and the training starts again.

The example displayed in Figure 2.2 and Figure 2.3 shows two classes 'A' and 'B' on a 2-dimensional scale. The analogue neural network system differentiates between the two classes by plotting a decision surface based on the training it has performed.

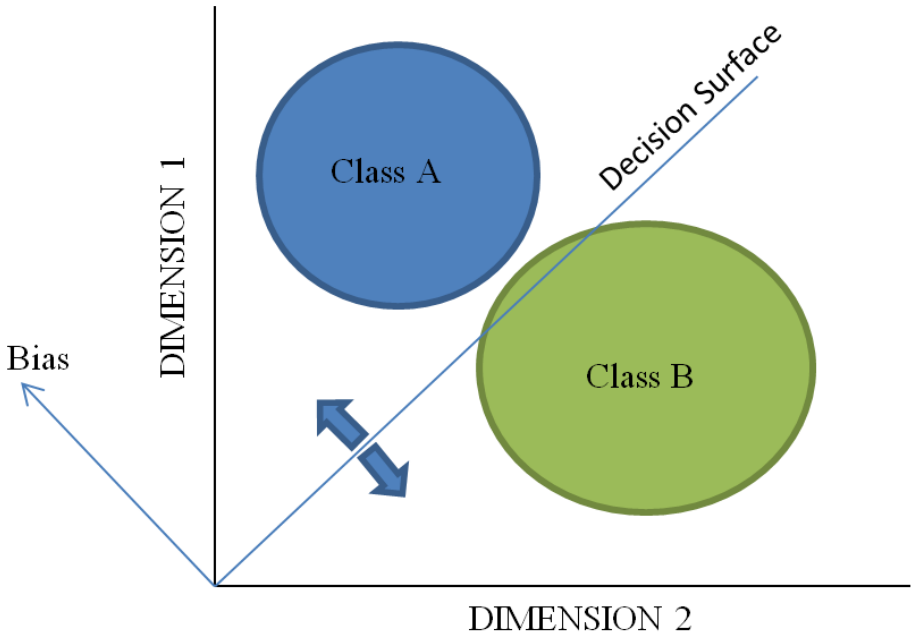


Figure 2.2 - Depiction of two analogue neural network classes and the decision surface changing during training

After a certain (unknown) amount of training time, the system produces a solution where both classes are accurately classified, as is shown in Figure 2.3.

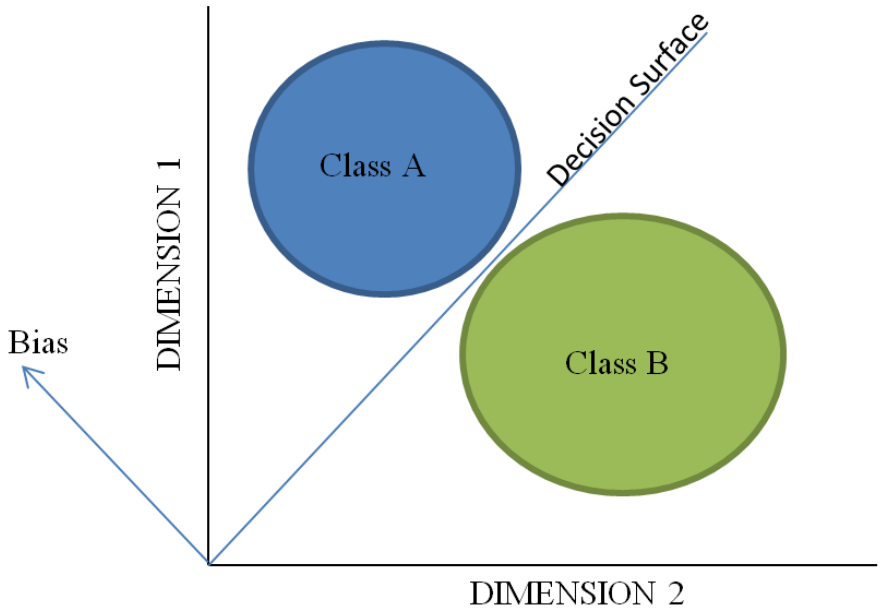


Figure 2.3 - Depiction of two analogue neural network classes being classified by the decision surface after training

It can be observed that patterns that have already been trained on in the past are not assured to be accurately classified if the system is trained on a new pattern, as the decision surface shifts. It should also be noted that the amount of training time required by the system is unknown, and in certain situations the system will continue its attempt at achieving an accurate decision surface where a solution cannot be found.

Digital patterns are multidimensional clusters, which we cannot draw, but can be represented in a 2-dimensional universe. The training method used to train most digital neural networks is called 'unsupervised learning', where "there is no external agent that overlooks the process of learning." (Kawaguchi, 2000) During the training phase, the system is only trained on images that belong to that class of images. If during the testing phase images that do not belong to the same class also achieve a positive response, then other aspects of the system require adjustment, an example being the functions performed by the neurons, or the way the neurons have been connected to each other, etc.

An example of the two dimensional clusters formed by a digital neural net is displayed in Figure 2.4, where class 'A' has three clusters, the training set T_A which is a subset of the data set D_A. Training on T_A produces a generalised net G_A, shown below:

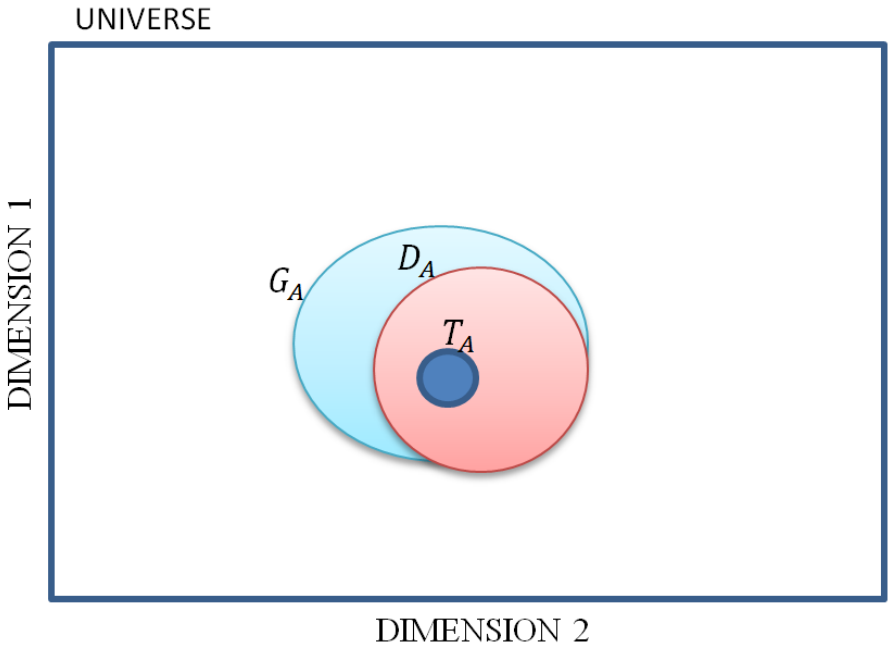


Figure 2.4 - Depiction of digital neural network classes on a 2D universe during training

Any patterns that have been trained on from the data set (and therefore appear in the training set) would achieve a 100% response if tested upon. When more images from D_A are used to train the net, the size of T_A increases slightly to encompass those trained patterns, however the size of G_A increases significantly more, as the net generalises further, as seen in Figure 2.5.

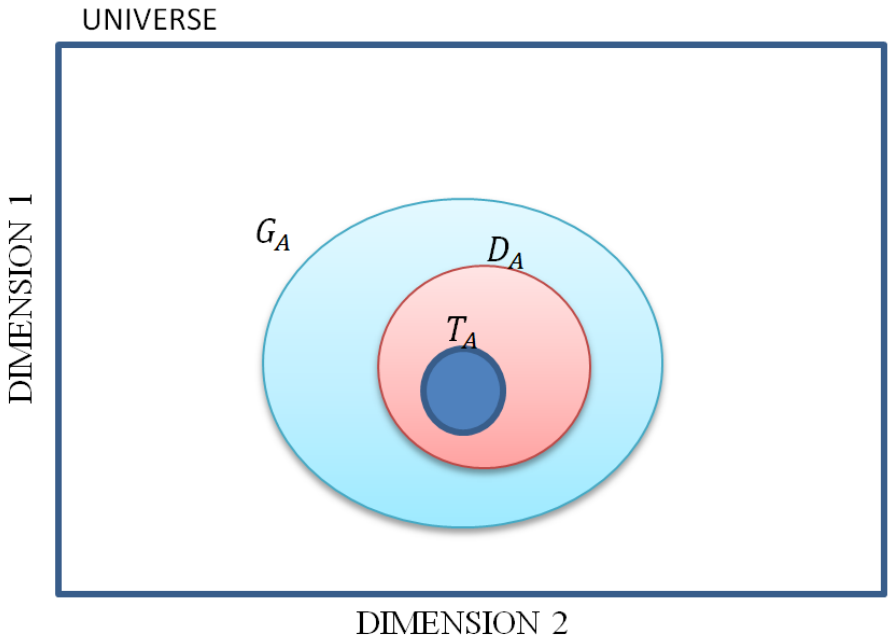


Figure 2.5 - Depiction of digital neural network classes on a 2D universe after adequate training

It can therefore be observed that the main difference between analogue and digital neural network systems is that in analogue systems even though a pattern has been trained on in the past, it is not guaranteed to achieve the correct response when tested upon, if other patterns were trained on after it. Whereas, in digital neural network systems, once a pattern has been trained on, it will achieve 100% accuracy if tested upon, and the generalisation that has occurred due to that pattern will never reduce. This make digital neural network approaches more powerful as an implementation method for pattern recognition systems.

2.2.2.2 History of analogue (weighted) neural network systems

2.2.2.2.1 McCulloch & Pitts

McCulloch & Pitts formulated the first artificial neuron in 1943, which was a system that applied a linear threshold to binary inputs and outputs, with weighted values being applied to each of the inputs (McCulloch and Pitts, 1943).

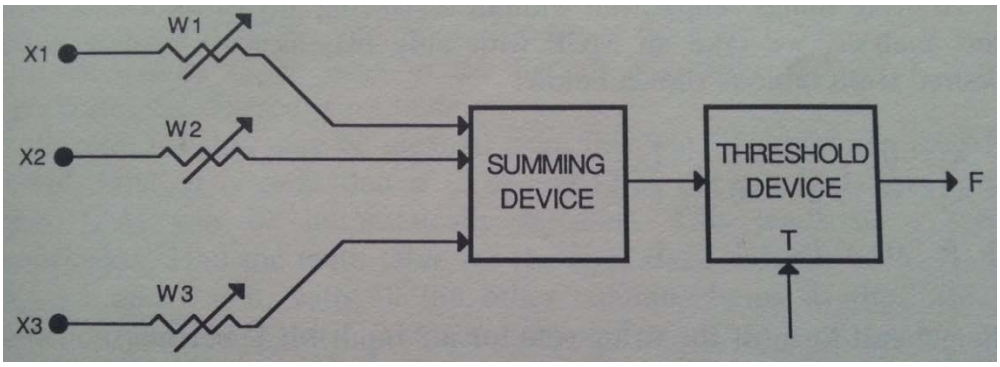


Figure 2.6 - An engineering model of the McCulloch & Pitts artificial neuron (Aleksander and Morton, 1995)

Figure 2.6 displays the McCulloch & Pitts model from an engineering point of view, where:

- X1, X2 and X3 are voltage values, where each voltage value can either be one of two pre-determined states – firing or not.
- W1, W2 and W3 are the independent weights that are applied to each input by means of a variable resistor.
- The THRESHOLD DEVICE is basically a voltage comparator, where the output of the SUMMING DEVICE is compared to a pre-set thresholding voltage T.
- F is the voltage output, from one of two voltage values, one corresponding to the neuron firing (when the SUMMING DEVICE voltage is greater than T) and the other to mean the neuron is not firing.

In the McCulloch & Pitts model, the value applied to each 'weight' determines to what degree the system should 'take notice' of firing signals from that input. The system fires (output 'F' is at firing voltage) if:

$$X_1W_1 + X_2W_2 + X_3W_3 + \ldots\ldots + X_nW_n > T$$

If for example the McCulloch & Pitts model was used to solve the 2-input Boolean OR gate below:

X_1	0	0	1	1
X_2	0	1	0	1
F	0	1	1	1

Table 2.1 - Boolean OR Gate truth table

Since:

$$F = 1 \; if \; X_1W_1 + X_2W_2 > T$$

Then inserting values from the truth table arrive at the following four inequalities:

X_1 X_2	0 0	0 1	1 0	1 1
F	0	1	1	1
Function	$0 < T$	$W_2 > T$	$W_1 > T$	$W_1 + W_2 > T$

Table 1.2 - Boolean OR Gate truth table with approximated functions

- If W_1 and W_2 are values between -1 and +1, then T has to be between -2 and +2 (to ensure that $W_1 + W_2 > T$ is satisfied).
- Since 0 < T, therefore: $0 < T \leq +2$
- Since $W_1 + W_2 > T$ and $W_1 > T$, it means $0 < T < +1$
- Therefore since $W_1 > T$ and $W_2 > T$, therefore both $T < W_1 < +1$ and $T < W_2 < +1$

Based on the truth table, the functions are depicted geometrically in Figure 2.7 below:

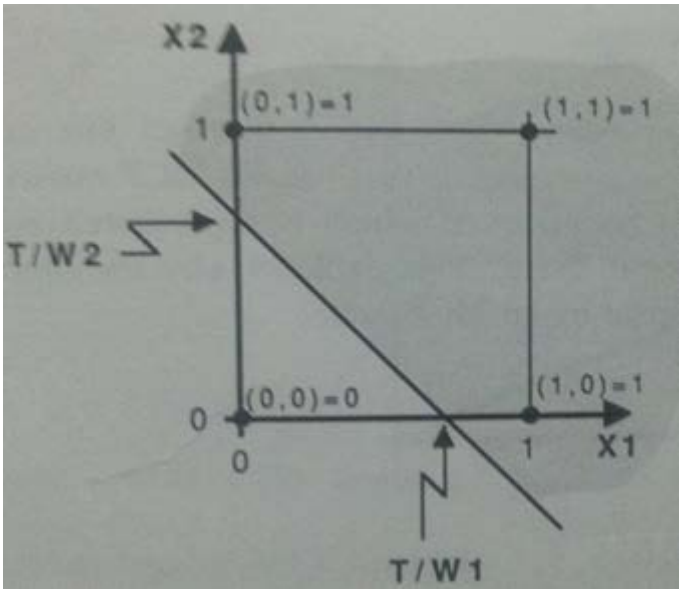


Figure 2.7 - Geometric depiction of the decision surface on the OR Gate (Aleksander and Morton, 1995)

In this graph, the two inputs X_1 and X_2 are the two dimensions, and since they are binary inputs, input can only be 0 or 1. Since $F = 1 \; if \; X_1W_1 + X_2W_2 > T$, the dividing line between the firing and non-firing stages of the neuron is when $X_1W_1 + X_2W_2 = T$; hence $X_1W_1 = T$ when $X_2 = 0$, and $X_2W_2 = T$ when $X_1 = 0$.

However these functions are still an approximation, and therefore values within the ranges specified above have to be used to test the functions, and the output checked against the truth table. This means that the McCulloch & Pitts system design performed function approximation, rather than input classification, which means that 'trial-and-error'

methods are used to try and arrive at a set of functions that would satisfy the output requirements.

2.2.2.2.2 Perceptron

In 1949, Hebb proposed what is now known as Hebb's rule, where he stated:

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

This rule was the first learning rule to be developed, and basically meant that if two neurons were firing together, then the weights should be adjusted to increase the strength between them.

Using both McCulloch & Pitts and Hebb's research, Rosenblatt developed the first perceptron in the late 1950s, a neural network model that could learn in the Hebean sense. (Rosenblatt, 1962) Although Rosenblatt created many variations of the perception, a simple version of the perceptron being used for pattern recognition consisted of a single-layer network, with 'association units' that "extract specific, localized features from some input image." (Aleksander and Morton, 1995) The network could be trained to recognise a set of features, and the perceptron gained great interest due to its ability to generalise from a set of training vectors.

Figure 2.8 below displays the version of the perceptron described above being implemented, where the system consists of 'p' number of association units, with 3 inputs to each unit. Weights are then applied to the output of each association unit, and the values are then summed and thresholded.

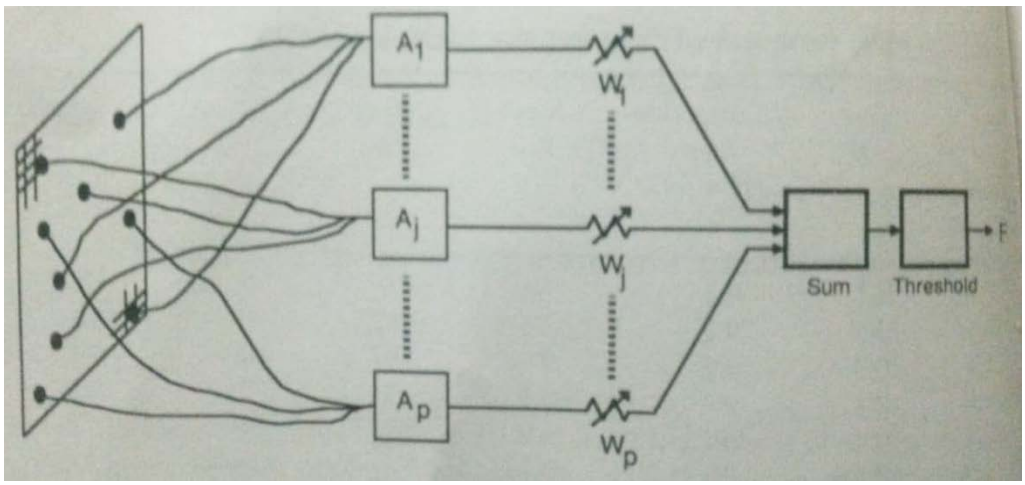


Figure 2.8 - Depiction of single-layer feed-forward perceptron for pattern recognition (Aleksander and Morton, 1995)

2.2.2.2.3 Multi-layer perceptrons

In the late 1960s, Minsky and Papert formulated that perceptrons could not be used to solve every task, and published a book attacking the limitations of the perceptron. (Minsky and Papert, 1969) By proving that a single layer perceptron could not arrive at a solution for the Boolean XOR gate, they theorised that single layer perceptrons could not solve linearly inseparable problems.

2.2.2.2.3.1 Feed-forward networks

The linear inseparability problem can be understood by observing the difference between the Boolean AND (Figure 2.9) and the Boolean XOR (Figure 2.10). The hollow circles on the graph represent an output of binary 0, whilst a black circle represents a binary value of 1. Whilst the two binary output classes can be separated by a single linear line drawn across the graph of the AND gate, the two classes in the XOR gate cannot be separated in this way, and any single linear line drawn across this two dimensional graph would result in at least one state being falsely classified.

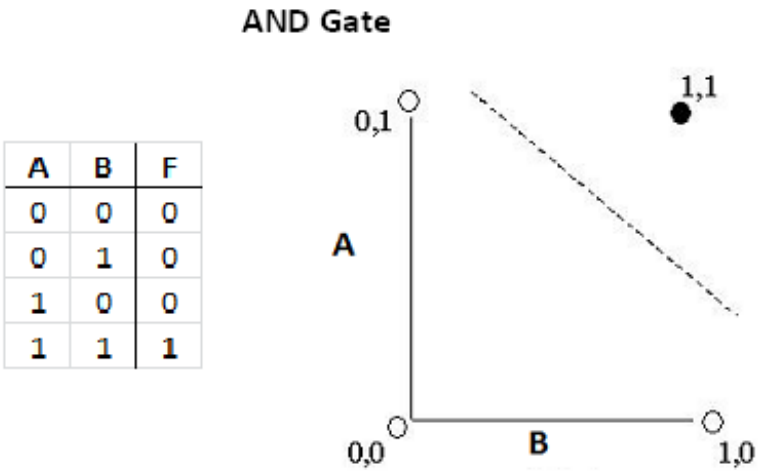


Figure 2.9 - Depiction of Boolean AND Gate

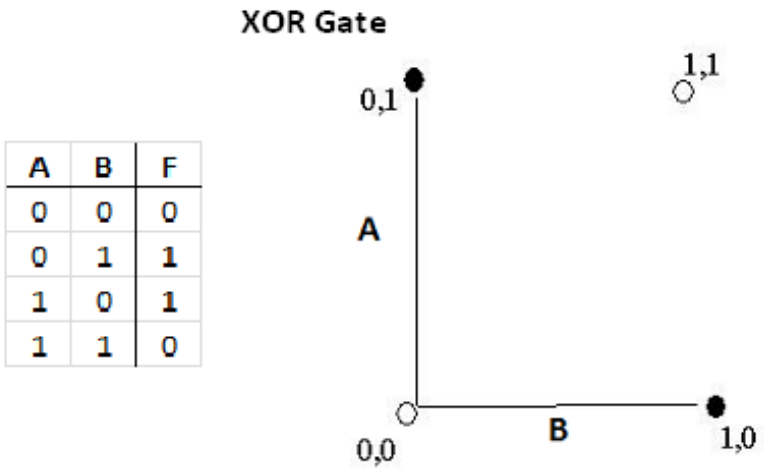


Figure 2.10 - Depiction of Boolean XOR Gate

> "Using arguments of symmetry, Minsky and Papert then showed, with their Group Invariance Theorem, that linear threshold functions which are invariant under a permutation group can be transformed into a function whose coefficients depend only on the group; a major result is that the only linear (i.e., order 1) functions invariant under such transitive groups as scaling, translation and rotation are simple size or area measures. Attempts to use linear threshold functions for, say, optical character recognition under these transitive conditions are thus doomed to failure." (Pollack, 1989)

Double layer feed-forward perceptron systems are also unable to solve linearly inseparable patterns such as the XOR problem (Beale and Jackson, 2010). The network (depicted in Figure 2.11) has two nodes in the first layer, where each node (node 1 and 2) calculates a dividing line based on its class. Node 3 then uses the outputs from the first two nodes to (supposedly) classify the XOR problem.

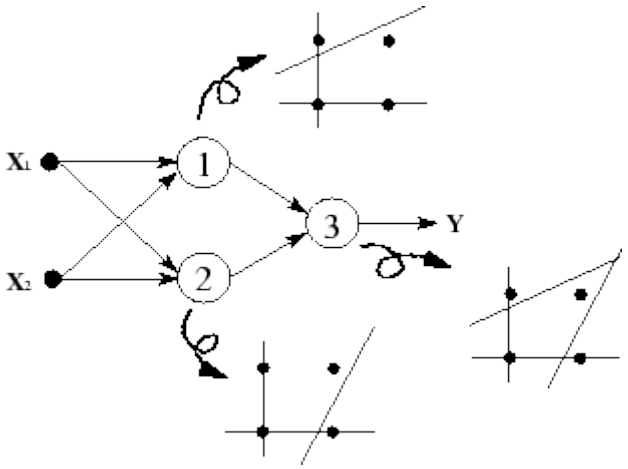


Figure 2.11 - Two-layer feed-forward perceptron attempting to solve the Boolean XOR Gate (Kawaguchi, 2000)

The problem is that during the learning stage:

> "The nodes in the output layer do not have access to the input information in order to adjust connection weights. Because the actual input signals are masked off by the intermediate layers of threshold perceptrons, there is no indication of how close they are to the threshold point." (Kawaguchi, 2000)

Therefore, it is not known how to adjust the weights of nodes 1 and 2, and therefore the training cannot be optimised, nor is it known if or when the training will converge to a solution.

2.2.2.2.3.2 Higher-order networks

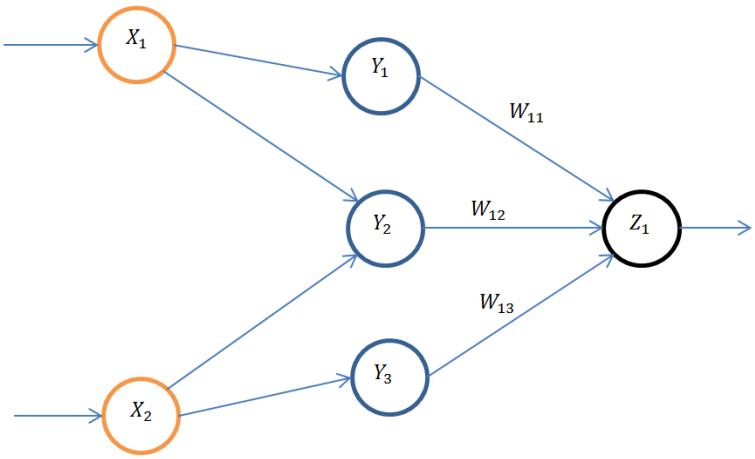


Figure 2.12 - An example of the sigma-pi feed-forward neural network

After Minsky & Papert's results, it was well known that single-layer feed-forward analogue neural networks could only implement linearly separable mappings, and although research into the use of multilayer networks in solving nonlinear problems was undertaken by some researchers, others decided to introduce higher-order units such as the sigma-pi unit (Giles and Maxwell, 1987; Rumelhart and McClelland, 1986), which is a higher-order exhaustive machine that is used to arrive at what is known as a multi-linear solution to a linearly inseparable problem.

Higher-order exhaustive machines are neural networks that use a higher combination of its inputs. For example, in the sigma-pi neural network (displayed in Figure 2.12) a weight is applied to the inputs and the higher-order conjucts of the inputs. The problem with sigma-pi networks are that the number of terms and weights increase drastically as the number of inputs increases, and therefore becomes unacceptably large for use in many situations. (Giles and Maxwell, 1987) Although it is possible to limit the number of nodes, and therefore the number of terms to a certain degree, this requires prior knowledge of the task at hand, and therefore the system largely depends on outside intervention.

2.2.2.2.3.3 Back-propagation

> "One of the great advantages of the single-layer perceptron was the ease with which one could apply the 'learning' rule to adjust the weights until a desired output was obtained. At first glance there seems to be no obvious systematic way to adjust the weights between the 'hidden' layers in a multilayer perceptron. However, this apparent obstacle was overcome by Rumelhart, Hinton, and Williams in 1986 with the development of the back-propagation rule. Similar rules had been obtained by Werbos in 1974, Parker in 1985, and LeCun in 1985. But it was the influential work Parallel Distributed Processing edited by Rumelhart and McClelland that made back-propagation the most popular learning rule for multilayer perceptrons." (Marsalli, 2006)

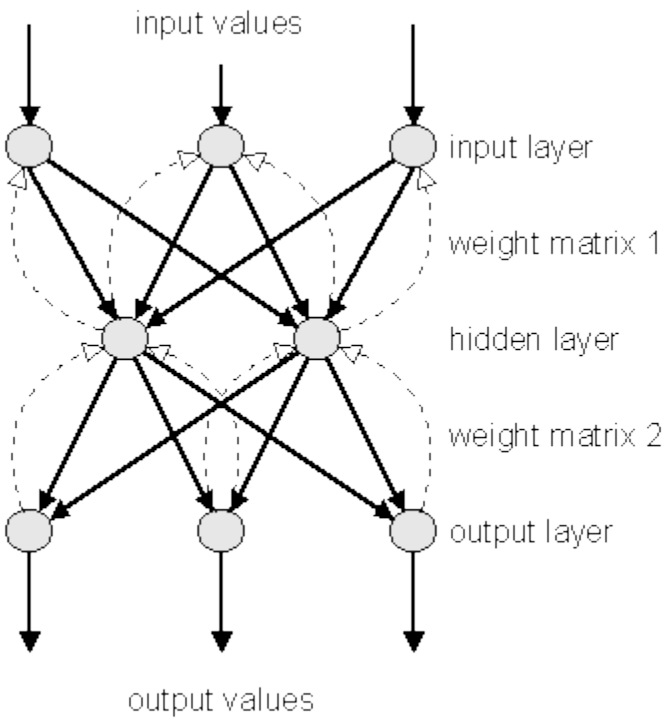


Figure 2.13 - Example of a multi-layer neural network (Fröhlich, 2004)

Back-propagation can only be performed in a multi-layer network where each neuron any layer of the network (apart from the input layer) has to be connected to all the neurons in the previous layer, as shown in Figure 2.13. This learning methodology consists of two phases, the forward phase, and the backward phase:

1. Forward Phase – during this phase, "a given collection of inputs is fed into the network, and the output of the network is observed. The observed output is subtracted from the desired output to produce an error." (Marsalli, 2006)
2. Backward Phase – During this phase, the error that was calculated in the forward phase is now fed backwards, where "at each step the weights are adjusted so as to make the observed output closed to the desired output." (Marsalli, 2006)

The back-propagation learning methodology gets its name from the backward phase, and the perceptrons are trained to produce increasingly accurate results by passing the errors they achieved on the same data backwards through the system. The weights in the system can either be updated after a single pattern is trained on, which is called on-line learning, or after the training patterns in an epoch have been presented to the network, called off-line learning. (Haykin, 1998)

Although there are some reasonable criteria which may be used to terminate the weight adjustments, such as if the "Euclidean norm of the gradient vector reaches a sufficiently small gradient threshold," the main drawback of the back-propagation approach is long learning times, and that the "back-propagation algorithm cannot, in general, be shown to converge, nor are there well defined criteria for stopping its operation."(Haykin, 1998)

2.2.2.3 History of digital (weightless) neural network systems

2.2.2.3.1 The Browning & Bledsoe algorithm

Weightless Neural Networks (WNNs) originate from the first N-tuple sampling system designed by Browning & Bledsoe in the late 50s (Bledsoe and Browning, 1959). It was an approximate statistical method that focussed on combinational co-occurrence statistics - the possibility of two points having specific recurrent values in different patterns belonging to the same class. Browning & Bledsoe designed a general weightless neural network algorithm that processed binary data, with the intended use being pattern recognition.

Browning & Bledsoe stated in the paper that “at this point this system is highly general – that is:

1. It handles all kinds of patterns with equal facility.
2. Because it does not depend upon absolute pattern-matching, it can identify a pattern which is not exactly alike, but only similar to, a pattern it has previously learned.
3. It does not depend significantly upon the location of a pattern on the photomosaic for identification.
4. It is only partially dependent upon the orientation and magnitude of a pattern for identification.” (Bledsoe and Browning, 1959)

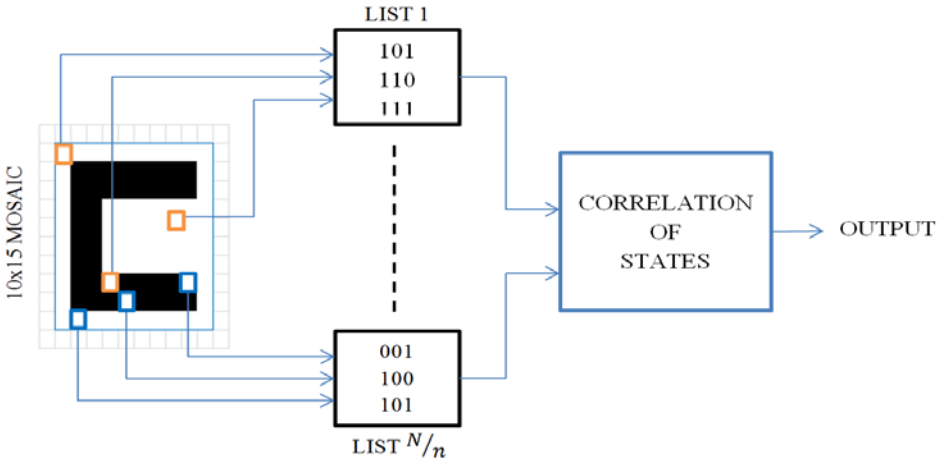


Figure 2.14 - Overview of the Browning & Bledsoe System

In order to test the algorithm, they implemented the system in logic, with the patterns being handwritten characters on a 10x15 photocell mosaic, each cell being either black or white; an overview of the system is displayed above in Figure 2.14. Random mapping is used to extract cell data in groups called ‘n-tuples’, and during training this data is then the ‘stored’ in its respective list. During testing, the data extracted was checked against

the stored lists, the scores are compared by the computer, and the list with the highest response score is identified as the class the pattern belongs to.

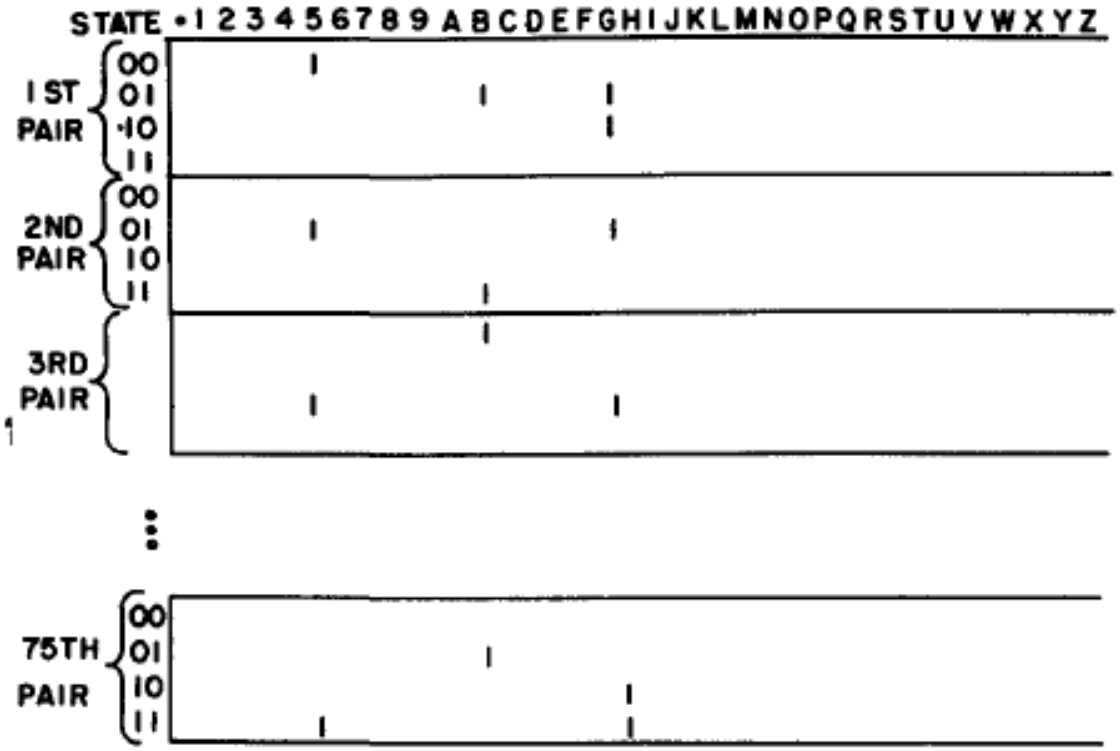


Figure 2.15 - Memory matrix used by Browning & Bledsoe to store trained states (Bledsoe and Browning, 1959)

Browning and Bledsoe used a memory matrix to store these lists, to allow fast serial access to the data. Figure 2.15 above displays an example of such memory matrix, where two-pixel n-tuples (n=2) are used to extract data from the photocells, and any state that occurs has a binary value of 1 assigned to it. Since the mosaic size is 10x15, the number of n-tuples used is 75 (N=75). The memory matrix above shows that the characters '5', 'B', and 'G' have been trained on. It should be noted that the column for each individual character in the matrix is called its 'net', and the nets above show that 2 images have been used to train the net to recognise the character 'G', whilst only 1 image each has been used for the other nets.

In order to test the rate with which the system generalised (and saturated), on the same 10x15 photocell mosaic, Browning & Bledsoe ran tests on the 36 characters, whilst changing the n-tuple size, and the number of patterns trained on. Figure 2.16 displays a graph of the comparison of the amount of characters accurately classified, versus the number of patterns trained on, for n-tuple sizes ranging from 1-5 pixels.

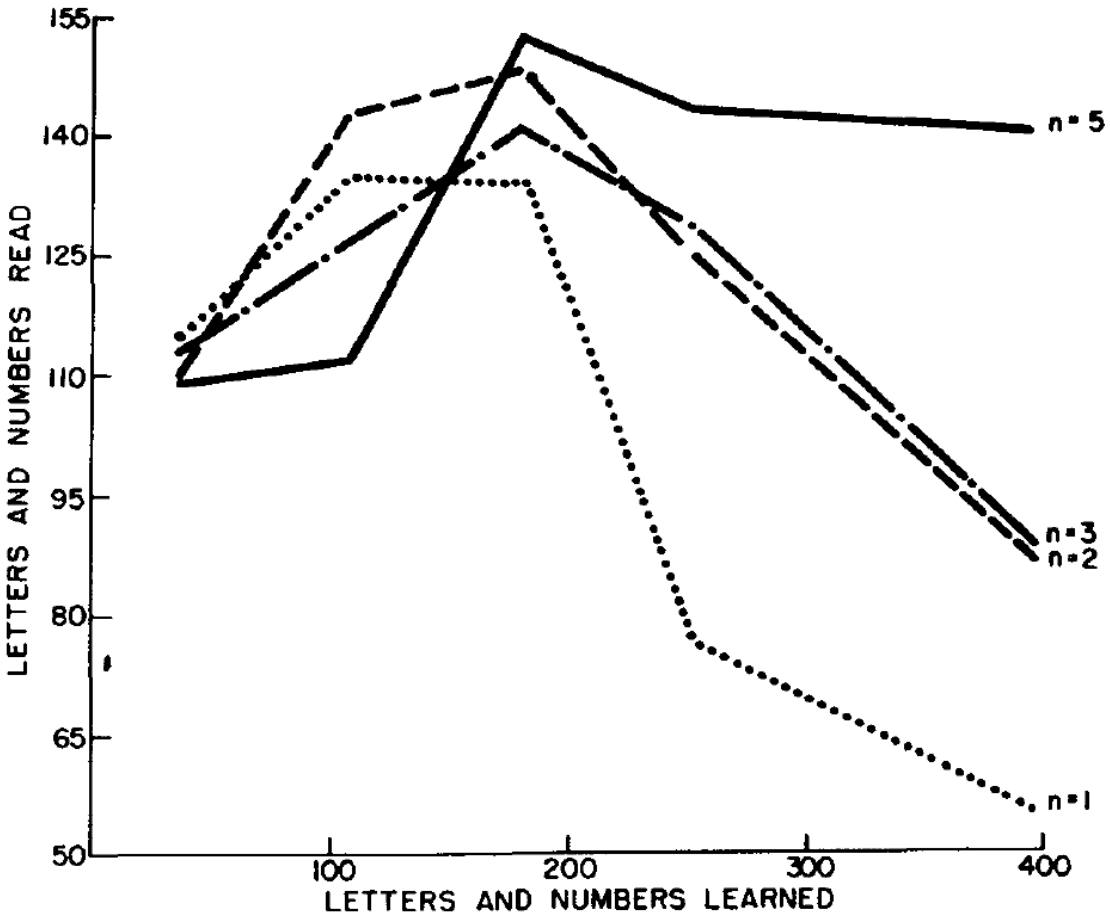


Figure 2.16 - Experimental results presented by Browning & Bledsoe on character recognition with different n-tuple sizes (Bledsoe and Browning, 1959)

Using the experimental results in Figure 2.16 above, Browning & Bledsoe suggested that a "greater amount of logic produces better discrimination," and that "the primary effect of varying 'n' is that as 'n' increases, the percentage of recognition increases with increased learning." This statement suggested that pixel n-tuple sizes required a larger number of patterns to train on, in order to generalise adequately, but it also suggested that since a lower pixel n-tuple size generalised quicker, it would also achieve net saturation faster.

The advantages of this system compared to the analogue neural network systems developed for pattern recognition were:

1. Unlike analogue neural networks, the system was not approximating functions that would divide the two classes in the universe, and therefore any pattern that was trained on would result in 100% accuracy, regardless of when it was trained on.
2. The generalisation of each net increased greatly as the number of training patterns increased.
3. The system was based on unsupervised learning - no external assistance was required to process any of the patterns.

4. Due to the memory matrix implementation of the system, it was fast and efficient during both training and testing stages.

The immense advantages of this system can be summarised effectively as such:

> "Although eclipsed in popularity by methods such as multilayer perceptrons and radial basis function networks, the n-tuple method continues to offer properties which make it vastly superior for certain common purposes. First among these properties is its speed of operation. The training algorithm is a one-shot memorization task, computationally trivial compared to solving linear systems or minimizing non-linear functions. Another advantage is the sheer simplicity of learning by memorization."(Rohwer and Morciniec, 1998)

However, in their paper, Browning & Bledsoe also stated that there were "two main disadvantages of the system:

1. When the learned patterns are quite variable, the memory can be saturated, especially in certain cases.
2. A very coarse mosaic, especially if it has inconsistent photocell performance, produces images of small letters which do not contain enough information for recognition. … The large letters, however, do not present this problem." (Bledsoe and Browning, 1959)

These two disadvantages were greatly due to the technology available at the time – the test was run on a 10x15 photocell mosaic due to its availability, and as Browning & Bledsoe stated in their paper, "these disadvantages can be at least partially overcome … by using a mosaic with more photocells." (Bledsoe and Browning, 1959)

2.2.2.3.2 The RAM node & discriminator

Aleksander built on the initial work of Browning & Bledsoe, and in the late 1960s introduced the Stored Logic Adaptive Microcircuit (SLAM). (Aleksander, 1966) This circuit was suggested as a universal logic circuit, to replace the memory matrixes used by Browning & Bledsoe in the learning neural network. The general concept that this microcircuit is based on is displayed in Figure 2.17, where an 8-bit storage unit can be accessed by just 3 binary inputs. These binary inputs are decoded, and data can be read from, or written to each individual (1-bit) storage location in parallel.

Although this concept seems very simple, it was a ground breaking development, which allowed binary data to be accessed in parallel, thus decreasing the read/write time drastically. This concept led to the development of the first RAM node, detailed diagrams of which have been discussed by Ludermir. (Ludermir *et al.*, 1999)

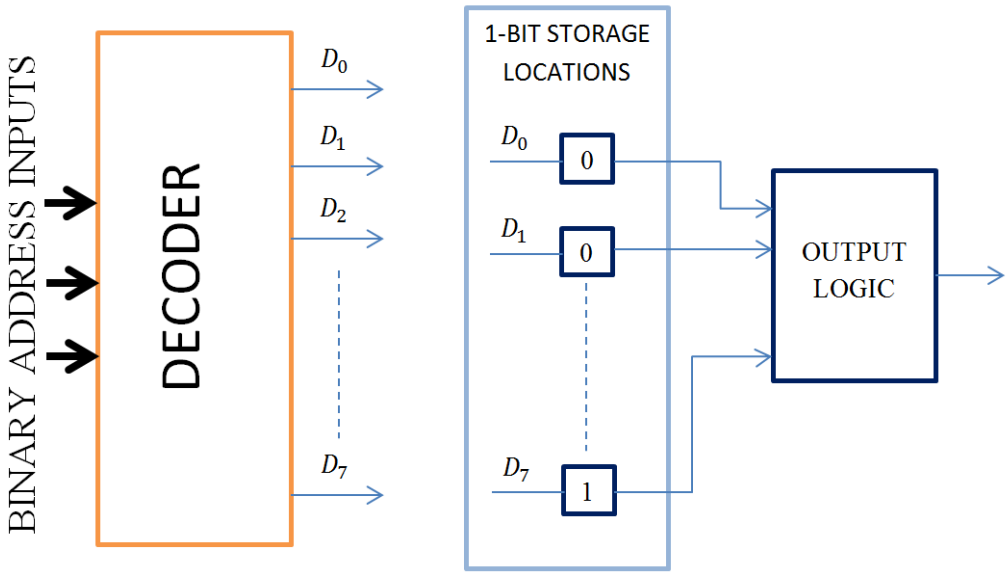


Figure 2.17 - General concept of Stored Logic Adaptive Microcircuit (SLAM)

Aleksander continued the research into SLAM in the 1970s with various other researchers, where first he and Fairhurst discussed the distance between classes that were trained using SLAM elements (Fairhurst and Aleksander, 1971), then together with Stonham considered chemical data recognition.(Stonham *et al.*, 1973) This led to the research by Stonham and Shaw, who used learning networks consisting of memory elements (RAM nodes) to automatically classify the mass spectra. (Stonham and Shaw, 1975)

In 1979, Aleksander and Stonham wrote a paper that gathered and reviewed 12 years of research into pattern-recognition using random-access memories, concluding it with the statement:

> "The technique can be used as a research tool because the recognition mechanism does not rely on any established theory of analysing of the data it is classifying. The opportunity for establishing new or hitherto unknown relationships between the measurements in the real-world on which the patterns are in some way based and the overall data description exists. The technique can be used to detect patterns or recognise known established patterns." (Aleksander and Stonham, 1979)

A common belief in the research prior to the WISARD system (Aleksander, Thomas and Bowden, 1984) being introduced was that the McCulloch & Pitts weighted theory produces systems that can generalise based on the training data, whereas a RAM node cannot. However, although a single RAM node cannot generalise without any external processing, by combining multiple nodes to create a RAM discriminators, generalisation was found to be possible.

A RAM Discriminator, displayed in Figure 2.18, consists of a single layer of RAM nodes, where each node is capable of storing 2^N bits. Each node has read and write capabilities, where the input vector is data extracted from the input image by the n-tuple, and the data in the input vector is used as the location in the relevant node. During training, in each node, the binary bit addressed by the input vector is set to '1', and during testing the values at the addressed locations in each of the nodes are summed, and are the final output from the discriminator.

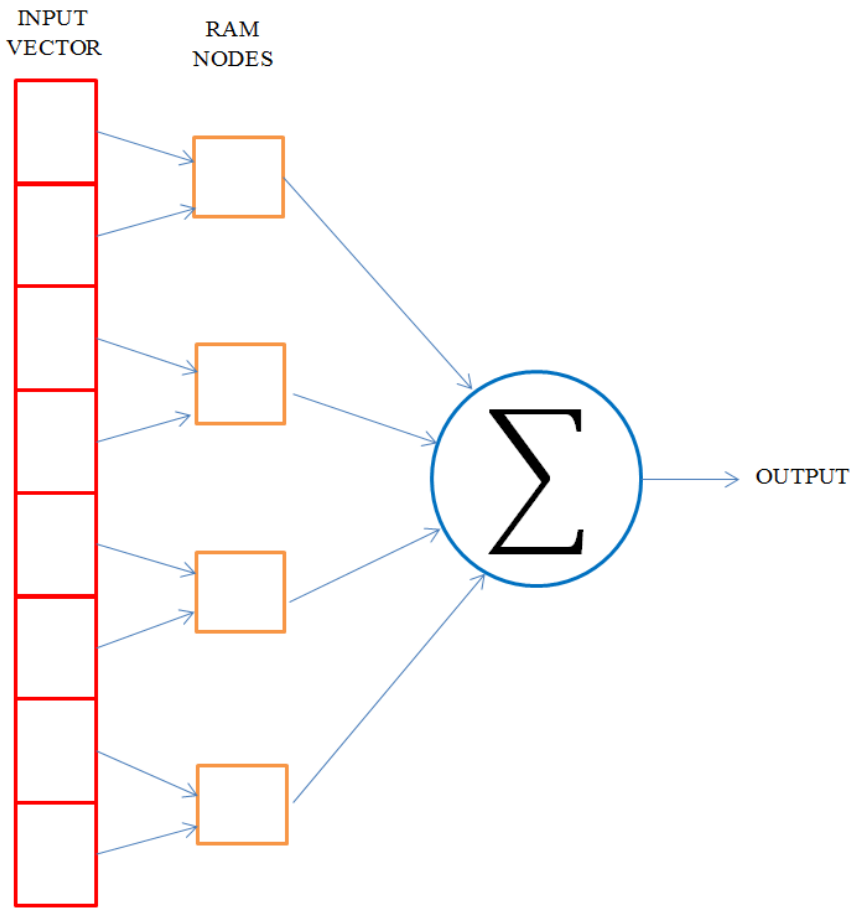


Figure 2.18 - General RAM Discriminator, consisting of multiple RAM nodes

The variations of states that occur during training individual patterns on the RAM discriminator are not required to exclusively occur in the pattern being tested on. Since an allowable state from one set can occur together with an allowable state from another set, generalisation occurs between training patterns.

2.2.2.3.3 The WISARD system

The RAM Discriminator was the basis of the design of the WISARD system, which was the first attempt to automatically recognise faces in real-time, and in the early 1980s, the first prototype of the WISARD (Wilkie, Stonham & Aleksander's Recognition Device) was

built, and was patented and sold commercially by 1984. (Aleksander and Morton, 1995; Aleksander, Thomas and Bowden, 1984)

The WISARD system was built by grouping together multiple RAM discriminators, where each discriminator was used to recognise a different class of patterns (Ludermir *et al.*, 1999). It was a breakthrough in pattern recognition, which used a 512x512 pixel image as an input, and up to 16 RAM discriminators, one for each class of patterns.

The structure of the hardware design, displayed in Figure 2.19, can be seen to be built of a discriminator unit that is contains the multiple RAM discriminators, each containing multiple RAM nodes, the main processing unit, which controls whether the system is training or testing on data, and a two part response calculator, one to calculate the individual response achieved by each discriminator, and another to calculate which response is the highest, which is then output by the system.

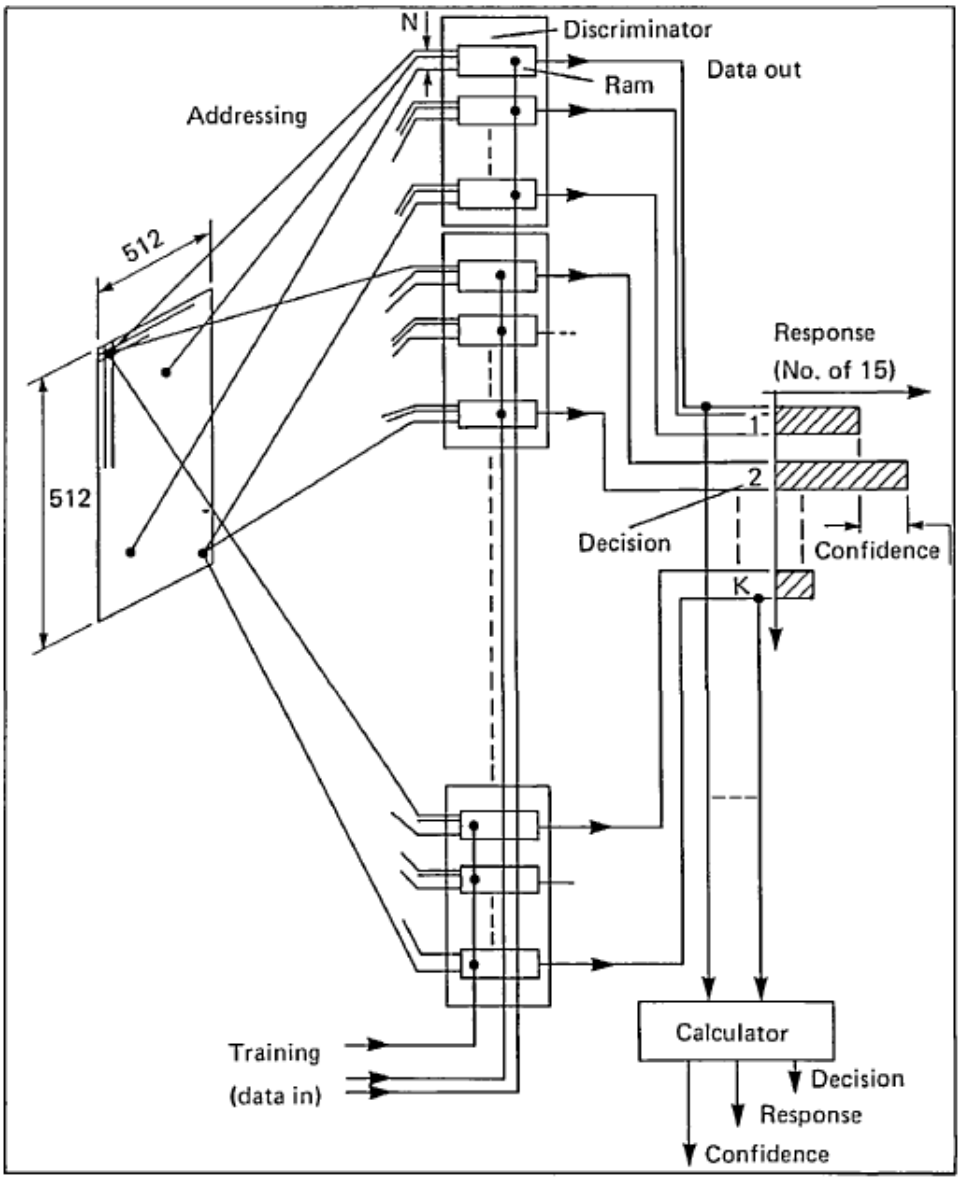


Figure 2.19 - Nominal structure of the WISARD System (Aleksander, Thomas and Bowden, 1984)

The success of the WISARD technique led to further research into other uses of this methodology, for example extracting edges contained in images (Wilson, 1986), scene analysis (James, 1986), texture discrimination (Patel and Stonham, 1991), electromyography (Pattichis *et al.*, 1994), hand-written character recognition (Tambouratzis, 1996), and even monitoring fish underwater (Chan *et al.*, 1999).

Research has also been carried out into the multiple aspects of the WISARD system, such as introducing unsupervised learning (Kerin and Stonham, 1990) using self-organisation techniques, and some attempts have also been made to modify and improve the WISARD architecture, like WIS-ART and AUTOWISARD (Wickert and Franca, 2002; Wickert and Franca, 2001; Fulcher, 1992).

Although the n-tuple based WISARD system was accurate when it encountered some variation in image lighting conditions, the fact remains that the accuracy is still dependant on the size of the chosen n-tuple (Aleksander and Stonham, 1979; Ullmann, 1969).

> "Experimental results, for instance, showed that a bigger size of N is better, until it approaches an impractical large size, though a value of N=8 turned out to be enough for many applications. … However, in general, the value of N is constrained by many considerations, where some of them might not be simultaneously satisfied in all situations. In such cases, a simple adjusting in N alone will not significantly improve the network performance." (Ludermir *et al.*, 1999)

2.3 Grey-level ranking methodology

Although the n-tuple process implemented in the WISARD (Aleksander, Thomas and Bowden, 1984) was a revolutionary step in neural network-based pattern recognition, the fact remains that this methodology, and the design it was based on (Bledsoe and Browning, 1959), are both limited by the ability to only process binary images. A problem arises when attempting to apply a threshold to the image being processed, as the threshold value depends on the lighting conditions present in the image.

In the past research has been undertaken in this field, leading to the design of various thresholding methods (Sezgin and Sankur, 2004), for example:

1. Histogram shape-based methods, which smooth the histogram of the image being processed, and then mathematically examine the peaks, valleys and curvatures present in the histogram (Sezan, 1990; Rosenfeld and de la Torre, 1983).
2. Clustering-based methods, where samples are taken from the image, and then clustered as either background or foreground (Otsu, 1979; Ridler and Calvard, 1978).
3. Spatial methods, which use higher-order probability distribution, and also correlation between pixels in the image (Kirby and Rosenfeld, 1979; Weszka and Rosenfeld, 1979).

4. Localized methods, which apply different threshold values to different areas in the same image, depending on the characteristics of the pixels located in that section of the image (Niblack, 1985; Nakagawa and Rosenfeld, 1979).

Four sample grayscale images are displayed in the top row of Figure 2.20, with the image results of various thresholding techniques in the rows below.

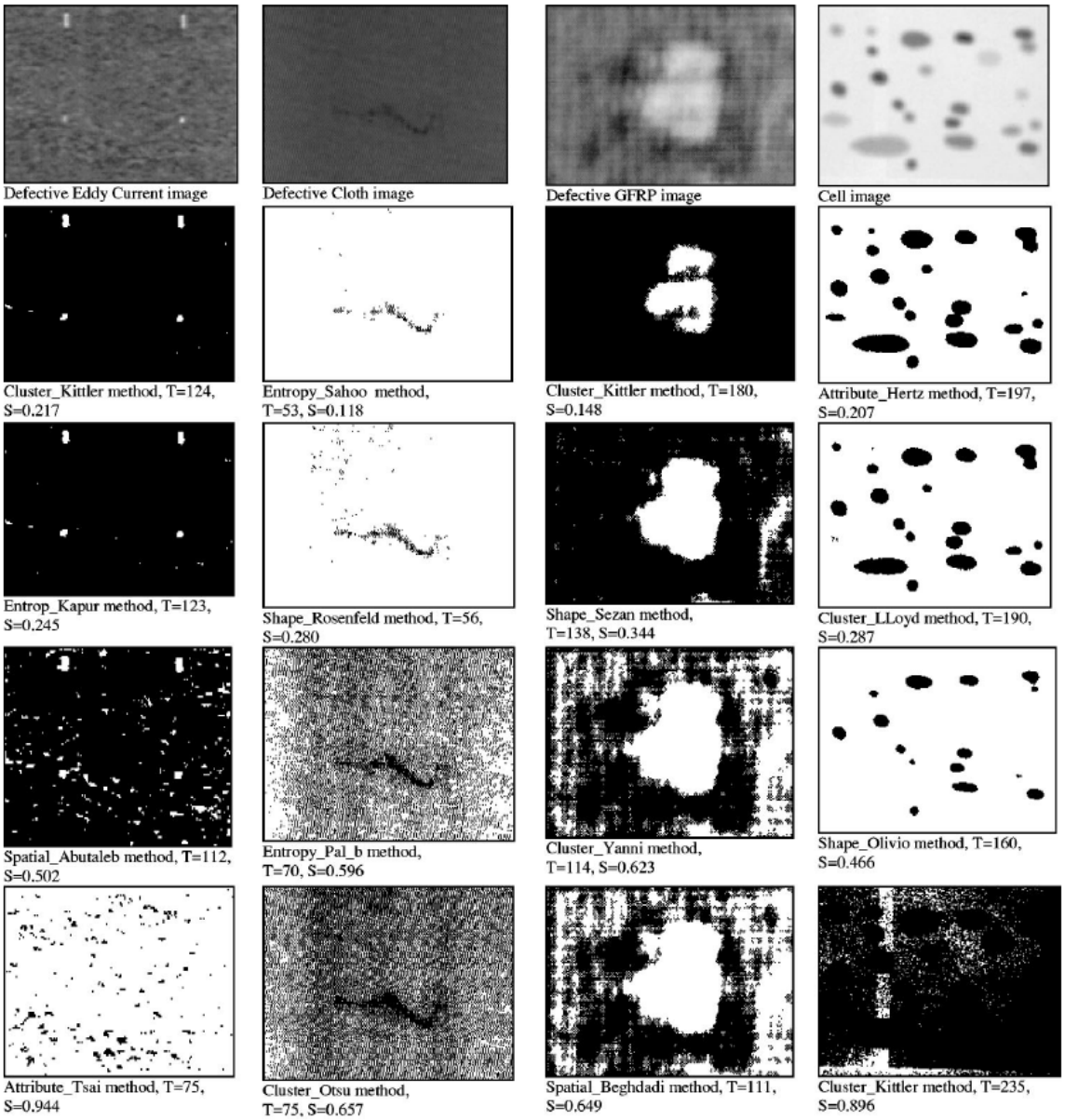


Figure 2.20 - Visual comparison of various thresholding techniques (Sezgin and Sankur, 2004)

Although intensive research has been carried out in this area, the problems encountered when attempting to threshold an image is best summed up by the following paragraph:

> "Automated image thresholding encounters difficulties when the foreground object constitutes a disproportionately small (large) area of the scene, or when the object and background grey levels possess substantially overlapping distributions, even resulting in an unimodal distribution. Furthermore, the histogram can be noisy if its estimate is based on only a small sample size, or it may have a comb-like structure due to histogram stretching/equalization efforts. Consequently, misclassified pixels and shape deformations of the object may adversely affect the quality-testing task in NDT applications. On the other hand, thresholded document images may end up with noise pixels both in the background and foreground, spoiling the original character bitmaps. Thresholding may also cause character deformations such as chipping away of character strokes or conversely adding bumps and merging of characters among themselves and/or with background objects. Spurious pixels as well as shape deformations are known to critically affect the character recognition rate." (Sezgin and Sankur, 2004)

Although the above statement is with regards to thresholding images of documents, the same is applicable to all images, especially those containing intensive detail (like faces), where the slightest bit of noise changes the contents of the image. In addition to this, thresholding is very costly with regards to computational intensity, and the speed of the algorithm, when applied as a pre-processing stage. These two factors are huge indicators that another simpler, faster, and more accurate solution can be found, hence the research into grey-level pixel ranking.

Grey-level ranking was first introduced by Austin, where 4-pixel n-tuples were used to divide the images, and for each n-tuple, after data was extracted from the image, the values were then sorted in descending order. For example, if the four pixels are labelled A, B, C and D, and the pixel values are 207, 169, 56 and 103 respectively, then the sorted n-tuple would be A, B, D and C, and this would be called a 'state'. Austin presented this algorithm as a pre-processing application for edge detection systems, however due to the cost of memory at the time, he applied a function to reduce the overall number of states. (Austin, 1988)

Patel used the same 4-pixel n-tuples for image classification and segmentation (Patel and Stonham, 1992), however he used all 24 states possible in a 4-pixel n-tuple, and allocated each state a unique number that he called 'ranks'. Patel used this method to classify images containing various texture samples, where he achieved an accuracy of 98%. Similar results were achieved by Hepplewhite in 1994, when he used the same methodology to inspect the surfaces of magnetic disks (Hepplewhite and Stonham, 1994).

Although further research has been performed (Hepplewhite and Stonham, 1996; Asselin de Beauville and Mraghni, 1996), and despite high accuracy levels that have been achieved in texture image classification and segmentation, this method has not been implemented in face recognition systems.

2.4 Summary

The algorithm presented in this thesis is a general image pattern recognition system based on weightless neural networks. The principle application is face recognition; however it can be used for any image-based operation where there is evidence of recurring local features.

In the past various techniques have been used in the design of pattern recognition systems, such as template matching, syntactic approaches, statistical, and neural networks (Jain, Duin and Jianchang Mao, 2000), however due to many algorithms being a combination of two or more techniques, it is difficult to classify them. The most frequent techniques used to build face recognition systems are either statistical or neural network-based approaches (Zhao *et al.*, 2003).

The statistical approaches that were reviewed were template matching, Eigenfaces, nearest feature line method, Fisherfaces, and Independent Component Analysis:

- Template matching was found to have, amongst others, problems if the test images varied in scale or shape (Ming-Hsuan Yang, Kriegman and Ahuja, 2002; Sakai, Nagao and Fujibayashi, 1969).
- Methodologies involving Eigenfaces also had a high error rate if the test image was smaller or larger than the training images, and these systems require a lot of time to optimise the initial eigenvectors that are used to build the Eigenfaces, in order to achieve high accuracy rates (Chengjun Liu and Wechsler, 2001).
- The nearest feature line method requires an optimised Eigen space, which (as mentioned above) requires a lot of time, and assumes that there are at least two prototype feature points (Lu 1999) in each class, which is not always guaranteed.
- Fisherfaces only achieves a lower error rate that Principal Component Analysis, however this is only the case when the patterns being tested on are very similar to the ones that have been trained on.
- Independent Component Analysis has a high computational intensity requirement, as it uses higher order calculations, and it also requires multiple image pre-processing to be applied before the image can be used.

Although statistical approaches have been used to design commercial pattern recognition systems in the past, they are plagued by the ‘curse of dimensionality’, and based on this and the above, the neural network approach was used to design the algorithm presented in this thesis.

Neural networks are divided into two distinct categories, analogue and digital. Classes used in analogue neural networks, sometimes known as weighted neural networks, are perceived by hyper surface, multi-dimensional planes, and most of these networks consists of an additional input to each neuron called the ‘bias’. Analogue neural network systems are trained to divide the multi-dimension pattern universe using a decision surface, in an attempt at classifying all patterns correctly.

During training, both patterns that belong to the class, and patterns that do not, are processed by the network, and the response is checked to see if it is correct. When the system outputs the wrong answer, the ‘bias’ on some (or all) of the neurons is adjusted

(supervised learning), and all the patterns are processed once again. Figure 2.2 and Figure 2.3 display examples of a 2-dimensional analogue neural network pattern space consisting of two classes, with a decision surface being trained.

Analogue neural networks were first discovered by McCulloch & Pitts in 1943 (McCulloch and Pitts, 1943), and led to the development of the perceptron in 1962 by Rosenblatt (Rosenblatt, 1962). Single layer neural networks consisting of perceptrons were found to be unable to solve linearly inseparable problems such as the Boolean XOR (Minsky and Papert, 1969), which led to the development of multi-layer perceptron solutions. Feed-forward multi-layer networks were found to be incapable of solving linearly inseparable problems, just as the single-layer networks, until back-propagation was fully discovered in the 1980s (Marsalli, 2006).

All analogue neural networks perform function approximation, and not data classification, and therefore there is no way to know in advance if the network being used to process the data is enough to arrive at an accurate solution. Also, whilst feed-forward networks cannot solve linearly inseparable problems, techniques that use back-propagation have “long learning times”, and they “cannot, in general, be shown to converge, nor are there well defined criteria for stopping” (Haykin, 1998).

Classes used in digital neural networks, also known as weightless neural networks, are perceived as clusters in a 2-dimensional universe, as shown in Figure 2.4 and Figure 2.5. Training these systems is fast and efficient, where once an image from the dataset is trained on, the system generalises based on the data extracted, and this generalisation does not decrease by training on another pattern (unsupervised learning), rather increases greatly.

Browning & Bledsoe designed the first n-tuple sampling system in the late 1950s (Bledsoe and Browning, 1959), which processed black and white 10x15 photocells consisting of hand-written characters. Aleksander built on their research, and by developing the SLAM node (Aleksander, 1966), allowed parallel access to binary data, decreasing the read/write time drastically. This led to the discovery of RAM nodes, and although individual RAM nodes did not have generalisation properties, a RAM discriminator, which consisted of a number of RAM nodes, could generalise.

Using multiple RAM discriminators, Aleksander went on to develop the WISARD system (Aleksander, Thomas and Bowden, 1984), alongside Wilkie and Stonham, which was the first automatic face recognition system to be developed. Although the n-tuple process implemented by WISARD was a revolutionary step in weightless neural network-based pattern recognition, the system is limited by the ability to process only binary images. Due to inaccuracies of thresholding, the algorithm in this thesis is a development based on the original WISARD system that uses grayscale ranking methodologies to process pixel data extracted from images, in order to recognise patterns in the images.

This research was inspired partly by the distinct lack of success being achieved by commercial pattern recognition systems, which have used standard analytical techniques and have failed to deliver sufficiently accurate performances for commercial

applications in for example, airports. It is believed that the digital neural network approach is capable of far superior performance at low cost, and one of the aims of this thesis is to establish this fact beyond doubt by using open source databases where comparison is possible.

Chapter 3 :

System Overview & Implementation

3. SYSTEM OVERVIEW & IMPLEMENTATION

3.1 Introduction

This chapter defines the methodology of weightless neural networks, and identifies the factors that can be optimised. The effects of changing various system parameters are also introduced. The chapter succinctly introduces the overall system, and the various system parameters will be investigated further in Chapter 6 of this thesis.

As discussed in Chapter 1, the methodology presented in this thesis is an adaptation of the WISARD system (Aleksander, Thomas and Bowden, 1984), which was based on the method proposed by Browning and Bledsoe in 1959 (Bledsoe and Browning, 1959). The methodology was first presented by the author in 2012 (Khaki and Stonham, 2012).

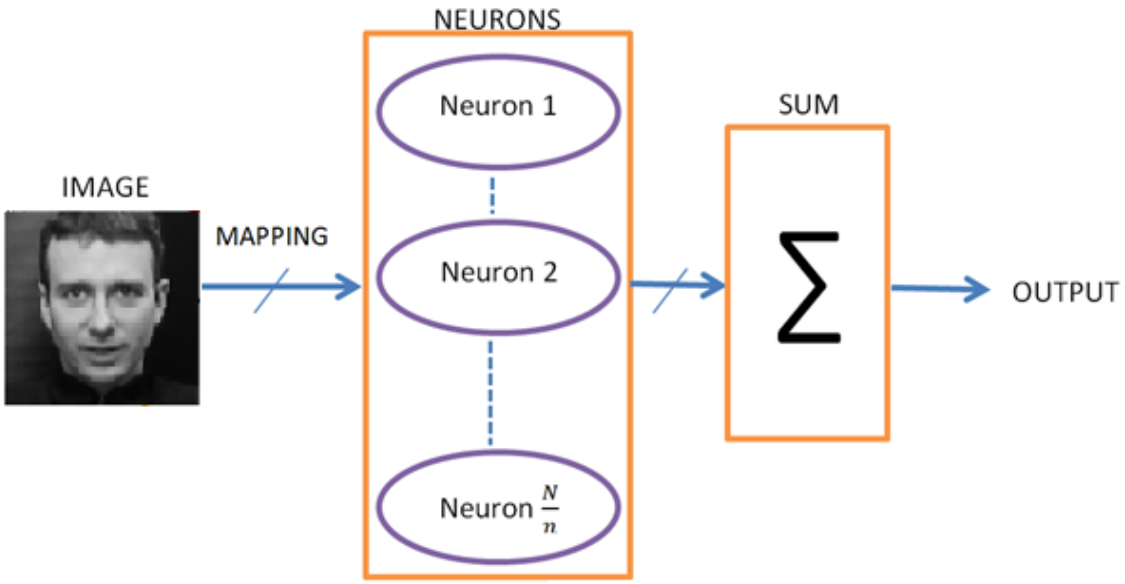


Figure 3.1 - Visual representation of overall methodology

The system (an overview of which is displayed in Figure 3.1) performs the following steps:

- Image is received for training or testing
- A pixel location table known as the 'mapping table' is used to extract pixel values from the image.
- These pixel values are grouped into sets called n-tuples; the number of pixels per n-tuple depends on the n-tuple size 'n'.
- Each n-tuple is the input to one neuron, and the neurons perform a 'ranking algorithm' on the pixel values.
- During training:
  - The state in binary becomes a term of the logic function implemented by the neuron.
  - Another image is then processed.

- During testing:
  - The function of each neuron determines whether the input (rank) causes the logic function to output '1', i.e. the test pattern n-tuple is a state that has occurred during training.
  - Positive or zero response is output by each neuron
  - Positive responses are summed, and an overall response is calculated
  - Response is output, and another image is then processed

As each neuron contains an individual storage facility, the combined storage space of the neurons is called the 'net', and it is this memory that holds the information of the trained images. There are three main parts of the system:

1. The pixel value extraction
2. The neuron computation
3. The output calculation

It is important to note that all the images being processed by the system are in grayscale, due to the reasons provided against thresholding in Chapter 2, and if the image is 8-bit grayscale, then:

$$Images \xrightarrow{Threshold} Input\ Vector: I_i \dots I_{i*j} \qquad where: I_i \in (0:255)$$

3.2 Pixel value extraction

The sequence according to which pixel values are extracted from the image is known as image mapping, and can be of three types:

1. Linear
2. Random
3. Specific

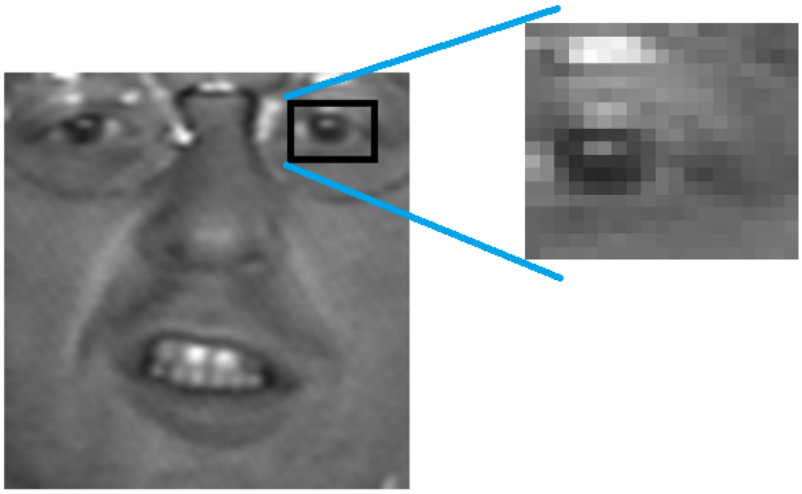


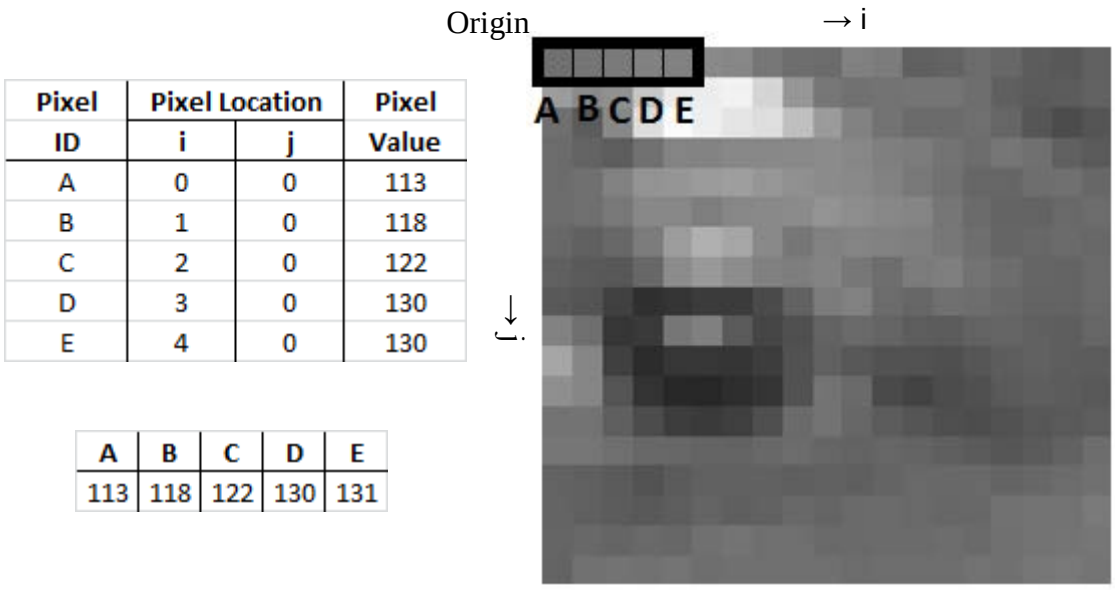
Figure 3.2 - Sub-section of face image

It is important to note, that the same mapping that is used to train the net should be used when testing images against it, otherwise the results obtained will be false.

A sub-section of a face image (displayed in Figure 3.2) is used to explain the different mapping techniques below.

3.2.1 Linear mapping

Linear mapping can either be horizontal or vertical-based mapping, an example of which is found in Figure 3.3, where a 5-pixel n-tuple is used to extract the pixels using a horizontal local operator. The pixel values that are extracted from the image are then used as the input to the neuron.



Pixel ID	Pixel Location i	Pixel Location j	Pixel Value
A	0	0	113
B	1	0	118
C	2	0	122
D	3	0	130
E	4	0	130

A	B	C	D	E
113	118	122	130	131

Figure 3.3 - Example of linear mapping

3.2.2 Random mapping

Random mapping is when the extraction of pixel values occurs without any obvious structure, however as stated above, the sequence of pixel locations has to be repeatable in order for it to be used to train and test any images. An example of random mapping can be found in Figure 3.4, where a 5-pixel n-tuple was used to extract pixel values from the image.

After the 5-pixel values are extracted from the image, they are then input into the neuron to determine the rank.

Pixel ID	Pixel Location i	Pixel Location j	Pixel Value
A	8	1	151
B	6	6	166
C	14	6	110
D	11	9	101
E	1	11	134

A	B	C	D	E
151	166	110	101	134

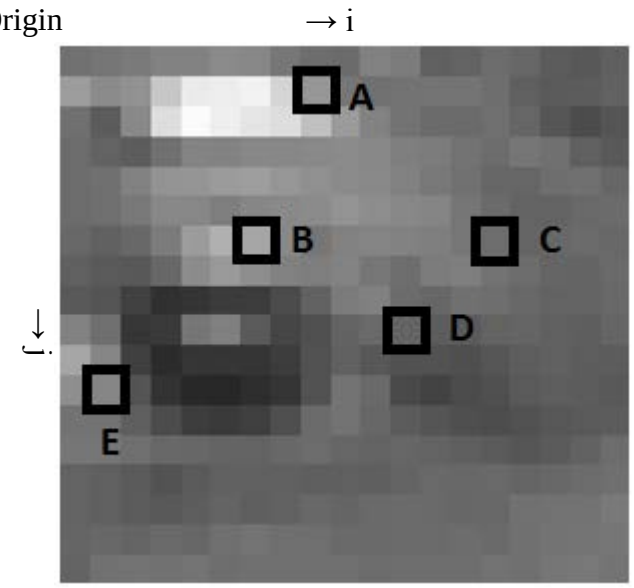


Figure 3.4 - Example of random mapping

3.2.3 Specific mapping

Specific pixel mapping is a mapping that has some form of structure, it can denote a specific shape on the image (a square, cross, etc.), or can be non-regular mapping (image is divided into four quadrants, and n-tuples are formed by one pixel from each quadrant, etc.). An example of specific-shape n-tuple mapping is displayed in Figure 3.5, where the n-tuple mapping forms the shape of a cross.

Pixel ID	Pixel Location i	Pixel Location j	Pixel Value
A	1	0	118
B	0	1	157
C	1	1	139
D	2	1	146
E	1	2	92

A	B	C	D	E
118	157	139	146	92

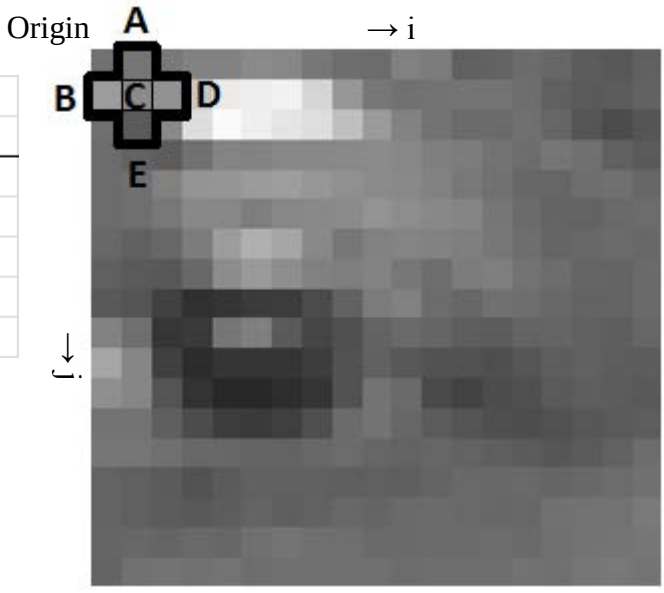


Figure 3.5 - Example of specific mapping

These pixel values that are extracted are then input into the neuron to determine the n-tuple rank.

The advantages and disadvantages of using these different mapping techniques is discussed later in Chapter 6, however it is important to reiterate that the mapping that is used whilst training the net should be used during testing, otherwise the results obtained will be false.

3.3 Neuron computation

As explained in the previous section of this chapter, pixel mapping is used to extract pixel values from the image being processed, and transfer those values into neurons that determine the n-tuple rank, and subsequently provide an input state to the neuron function. Each n-tuple connects only to one neuron, and the total number of neurons depends on two factors – the size of the image, and the number of pixels per n-tuple. The formula is as follows:

$$Number\ of\ Neurons = \frac{N}{n} \qquad where: N = i * j$$

$$n = \#\ of\ pixels\ per\ ntuple$$

Since each neuron carries out a specific function, the overall function of the system can be depicted as:

$$Function = \sum \left(F_1, F_2, F_3 \dots \dots, F_{\frac{N}{n}}\right) \qquad where: F_i - i^{th}\ function$$

The main difference between the methodology presented in this thesis, and the WISARD system developed in 1984 (Aleksander, Thomas and Bowden, 1984), is the type of pixels being processed. As explained in Chapter 2, the WISARD system processed images consisting of only black and white pixels, whereas the methodology presented in this thesis does not perform any thresholding algorithm as a pre-processing step. Instead, a grey-level ranking algorithm is performed on pixel values extracted from the image, as discussed in Chapter 2, where from the set of pixels in the Image (8-bit grayscale):

$$Images \xrightarrow{Threshold} Input\ Vector: I_i \dots I_{i*j} \qquad where: I_i \in (0: 2^8)$$

3.3.1 The ranking algorithm

The n-tuple extracts a set number of pixel values:

$$ntuple\ sample \rightarrow n\ values\ from\ subset\ of\ I - I_i \dots I_{j*k}$$

Since the image being processed is in 8-bit grayscale, each pixel has a value between 0 and 255, where 0 is absolute black, and 255 is absolute white. This set of pixel values is

then sorted into descending order, and each possible combination of the 'n' number of pixels is allocated an integer state, which is called a 'rank'.

$$Ranks\ I_k \geq I_k \ldots \geq I_n - \ state\ 0 \qquad to \qquad I_k \leq I_k \ldots \leq I_n - \ state\ (n! - 1)$$

For example, if the n-tuple size was 5, the ranks would be allocated as such:

Order									Rank
Pixel A	==	Pixel B	==	Pixel C	==	Pixel D	==	Pixel E	0
Pixel A	≥	Pixel B	≥	Pixel C	≥	Pixel D	≥	Pixel E	1
Pixel A	≥	Pixel B	≥	Pixel C	≥	Pixel E	≥	Pixel D	2
Pixel A	≥	Pixel B	≥	Pixel D	≥	Pixel C	≥	Pixel E	3
Pixel A	≥	Pixel B	≥	Pixel D	≥	Pixel E	≥	Pixel C	4
Pixel A	≥	Pixel B	≥	Pixel E	≥	Pixel C	≥	Pixel D	5
Pixel A	≥	Pixel B	≥	Pixel E	≥	Pixel D	≥	Pixel C	6
				↓					
Pixel E	≥	Pixel D	≥	Pixel C	≥	Pixel B	≥	Pixel A	120

Table 2.1 - Sample ranking table for n=5

Therefore, in the example presented in Figure 3.4 in the previous section of this chapter, the rank would be:

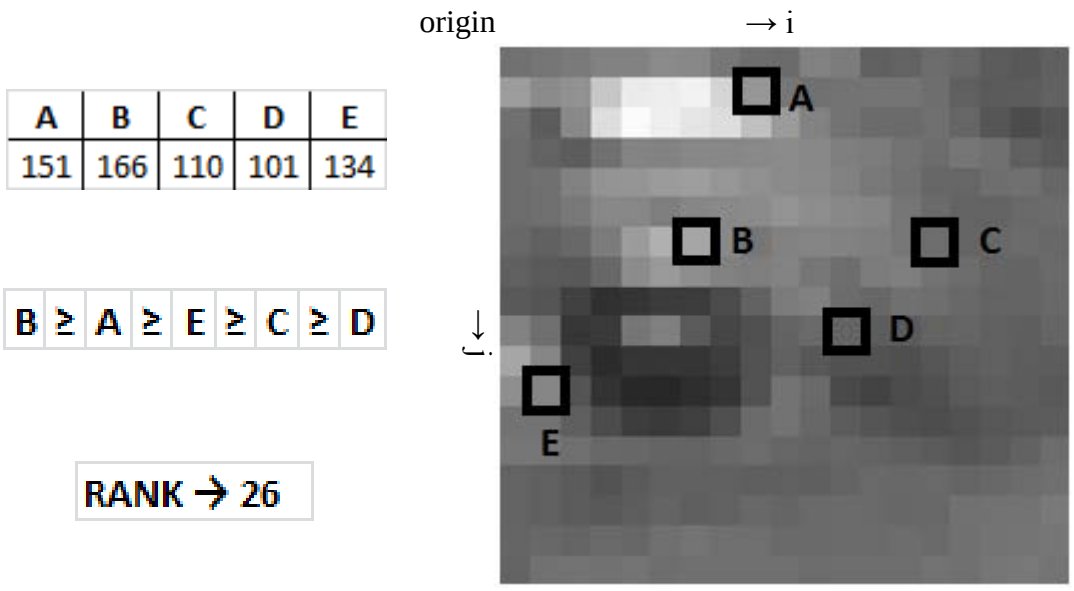


Figure 3.6 - An example of ranking algorithm implementation

The rank that is calculated by each neuron is stored within the neuron itself. Therefore, each neuron contains a specific number of binary bits that enable it to store the ranks

achieved during the training phase, where the address of each bit corresponds to a rank achievable by the n-tuple. Therefore:

$$Function\ F_i = (B_0, B_1, B_2 \dots\dots, B_{n!-1}) \qquad where: B_i \in (0,1)$$

When the 'net' is first created, each binary bit (whose address corresponds to the rank) in each function is set to a value of '0', and during the training phase, when a rank is computed from the pixel values extracted, the binary bit in that location is set to '1'. During testing, any binary bit corresponding to the rank that is computed is checked to see if it is set to '1', and if so, the neuron emits a positive response.

3.3.2 The storage requirements

The neurons will perform combinational logic functions, these will be implemented using programmable logic technology based on memory. The size of the system therefore can be measured based on the storage requirements for the logic functions in the networks.

Due to the binary nature of the system, if a 'net' contains of too many bits set to '1', then it will result in a high response when tested against images that do not belong to the same class, and the net is then said to have reached 'saturation'. It should be noted that the algorithm presented in this thesis is less susceptible to saturation due to the use of grayscale ranking; this will be discussed at length in Chapter 6 of this thesis.

Just as the number of neurons required by the system depends on the image size and the number of pixels used per n-tuple, so does the storage, where:

$$Storage\ (F_i) \rightarrow \left(\frac{(w * h)}{n}\right) * (n!) \qquad where: w - Image\ width$$

$$h - Image\ height$$
$$n - ntuple\ size$$
$$n! - total\ \#\ of\ ranks\ possible$$

For example, if the input image is 100x100 pixels:

n-tuple size	Total # of n-tuples	Total storage requirement
4	2500	58.6 KB
5	2000	234.4 KB
6	1666	1.14 MB
7	1428	6.86 MB

Table 3.2 - Effect of n-tuple size on storage requirement

It is observed in table above that if the image size remains the same, increasing the n-tuple size by 1 pixel decreases the total number of n-tuples slightly, however there is a

great increase in the total storage requirement; this will be discussed further in Chapter 6 of this thesis.

It should be noted that in some cases due to the size of the n-tuples being used and the size of the image, not all pixels will be processed in an image. For example, in Table 3.2, when the n-tuple size is 6, the total number of functions being used is 1666, however:

$$(100x100) - (1666x6) = 4$$

Hence 4 out of 10,000 pixels in each image being processed are ignored; since this is only 0.04% of the total image, it is deemed an acceptable loss.

3.4 Output calculation

During the testing phase, each individual neuron emits a positive response if the rank calculated from the pixel values obtained is present in the training data. These responses are all summed to calculate the total response of the image when tested against the trained 'net', where:

$$Overall\ response = \left(\frac{total\ \#\ of\ positive\ responses}{Number\ Of\ Functions}\right) * 100$$

It should be noted that although this is the overall response of the system, an extra thresholding step can be performed at the output, where a certain minimum percentage value is required for the image to be classified as being part of the trained 'net'.

3.5 Software implementation

A hardware/software representation of the system is shown below, where the neurons are replaced by programmable logic functions.

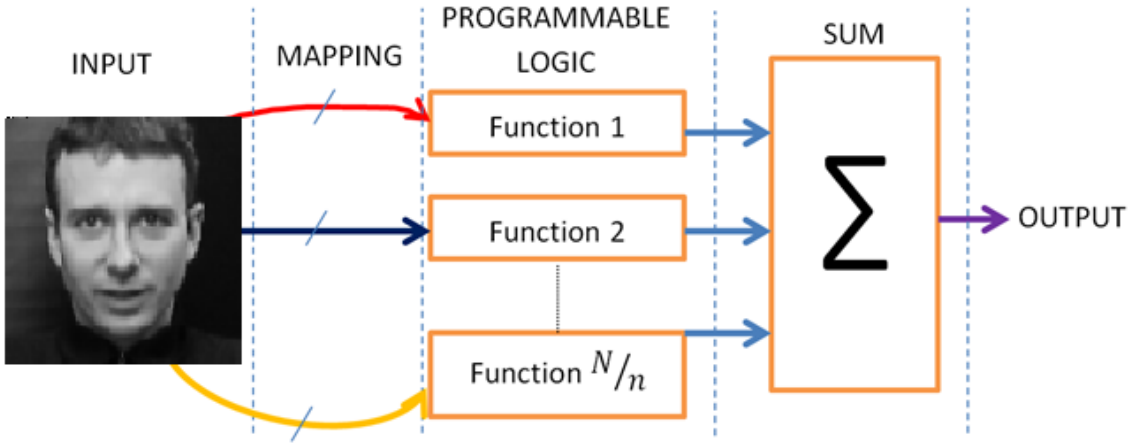


Figure 3.7 - Hardware/Software representation of the system

The methodology presented in this thesis was implemented in C++, based on the system overview (Figure 3.7). Over the course of the research, the following developing platforms were used to design and implement the system:

1. Qt Creator (Digia, 2013) and OpenCV (Windows, Linux & Mac OSX Versions) (OpenCV, 2013)
2. Qt Creator (Nokia N900 mobile implementation)
3. C++ Builder XE3 (Embarcadero, 2013) and Mitov VideoLab for Windows (Mitov Software, 2013)

It should be noted that the software implementation was designed and implemented from scratch in each of the three cases above, and that the OpenCV and Mitov VideoLab components were used only for their video-frame capturing abilities.

The programming language C++ was chosen only due to the researcher student's experience with the development platforms, and the methodology was implemented in multiple Operating Systems such as Windows (XP, Vista & 7), Linux (Ubuntu) and Mac OSX (Lion), and was also tested on the N900 mobile operating system (Maemo).

Although the system design consists of parallel neuron computation, single-threaded software was designed for its simplicity. Although the system processes approximately 36 frames a second at a 160x120 pixel resolution, this speed will increase dramatically if the system is implemented in hardware.

The tests results presented in the following chapters were all performed on the system designed on the C++ Builder XE3 development platform, with Mitov VideoLab being used to capture individual frames from video segments.

3.6 Summary

This chapter was an overview of the methodology being presented in this thesis, and also introduced the two major aspects of the system that can be optimised – the pixel mapping method and the n-tuple size. The effect that image size has on the storage requirements of the system was also discussed, together with the various software implementations that were designed during the course of the research.

The aspects of the system that can be used to optimise its implementation will be investigated further in Chapters 6 and 7. The accuracy of different pixel mapping methods and a an investigation into the effects of changing the n-tuple size will be discussed in Chapter 6, whilst Chapter 7 will focus on the effect different image conditions, such as size, lighting and zoom, have on its overall accuracy.

Chapter 4 :

Weightless Neural Networks as Image Filters

4. WEIGHTLESS NEURAL NETWORKS AS IMAGE FILTERS

The methodology presented in Chapter 3 of this thesis is a pattern recognition algorithm that can be implemented in various aspects of the real-world, such as simple image filtering, real-world scene understanding, face localization, and face recognition. The objective is to design a system that will recognize consistent scenes, or objects within a scene; however there is a problem with recognizing objects within a scene. For example, if the algorithm was to recognize a bookcase, although human perception states when the bookcase or part of one is present in an image, to the system the bookcase is made up of millions of pixels which have to be recognized, and any slight change in position or viewing distance would immensely affect the data input into the algorithm. Another main objective is to design a self-evolving system, and therefore prior analysis of the data is not allowed.

This chapter applies the methodology presented in Chapter 3 of this thesis firstly on a simple example of recognising a component within a scene, i.e. a bookcase. Then the algorithm is extended to a more challenging problem, where it is required to recognize consistent scenes within a video sequence. The video sequence used is a news transmission that was found online. This video has not been pre-processed in any way, and the objective was to recognise the scenes that have been trained on.

4.1 Results formatting

The system consists of nets, each one detecting a perceived class of data. During the training stage, each net is trained on images belonging to it, and during the testing stage, unique (not used during training) images are tested on, and a response is output by the system on each image tested, versus each trained net.

When an image is tested, a resulting accuracy is arrived at, based on the percentage response it achieved on each net that existed in the database. The response of an image when tested is calculated as follows:

$$Response\ on\ a\ net = \frac{Ranks\ that\ are\ both\ in\ the\ testing\ image\ and\ the\ trained\ net}{Total\ number\ of\ functions}$$

After all the images have been tested on, the accuracy of the system is calculated:

$$Accuracy\ of\ the\ System = \left(\frac{Number\ of\ images\ correctly\ classified}{Total\ number\ of\ images\ tested}\right) * 100\%$$

Although this is the overall accuracy of the system, two other parameters are widely used to assess the performance of the system, namely the False Acceptance Ratio and the False Rejection Ratio.

4.1.1 False Acceptance Ratio (FAR)

The False Acceptance Ratio depicts the percentage of images that have been falsely recognised as being part of specific classes, over the overall number of images that the system has tested.

$$FAR = \left(\frac{Number\ of\ images\ falsely\ recognised}{Total\ numbner\ of\ images\ tested}\right) * 100\%$$

4.1.2 False Rejection Ratio (FRR)

The False Rejection Ratio depicts the percentage of images that have been falsely rejected from being part of specific classes, over the total number of images that the system has been tested on.

$$FRR = \left(\frac{Number\ of\ images\ falsely\ rejected}{Total\ number\ of\ images\ tested}\right) * 100\%$$

4.1.3 Response Graphs

For each test that is performed in this chapter of the thesis, and the following chapters, the results obtained are presented in a series of graphs. In order to introduce these graphs, sample test data (displayed in Table 4.1) is used below.

	Net 0	Net 1	Net 2	Net 3
1	52	28	23	20
2	16	68	22	20
3	17	21	64	34
4	17	21	38	63

Table 4.1 - Sample test data

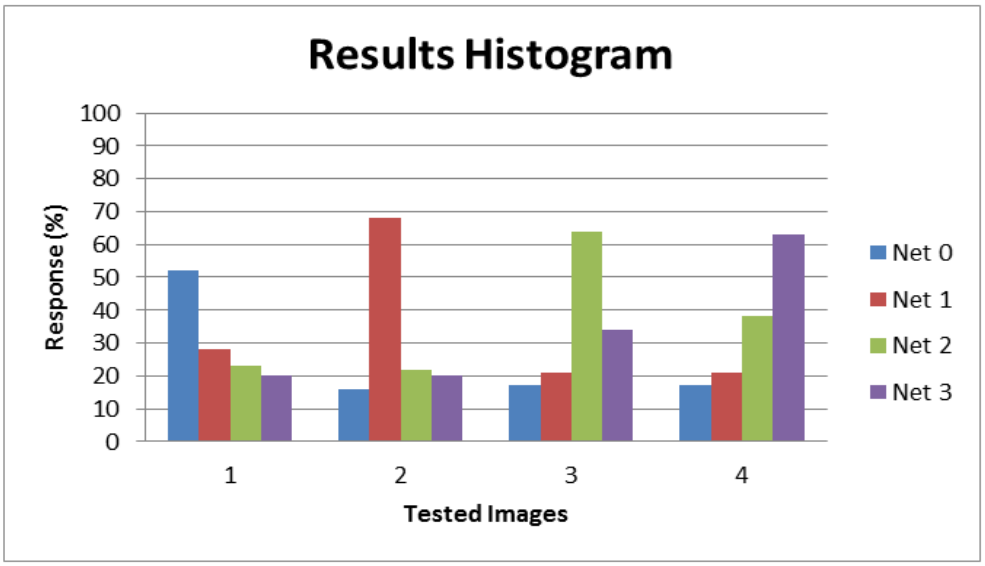


Figure 4.1 - Histogram of sample results

Table 4.1 displays the results of four images when tested against four nets. Each image belongs to one net, and therefore it is expected that each image will produce the highest response when tested on the net it belongs to. These results are then displayed in a visual format in Figure 4.1, where the results that each net produces for a single image are displayed as a group, in order to accurately visualise the accuracy of the system.

Although this is the most efficient method of displaying the responses achieved during each test, due to the number of images tested and the number of nets that were tested on, it is not always possible to clearly display the results in the above format. Therefore, the results that are obtained in each test will be presented as displayed in Figure 4.2 below:

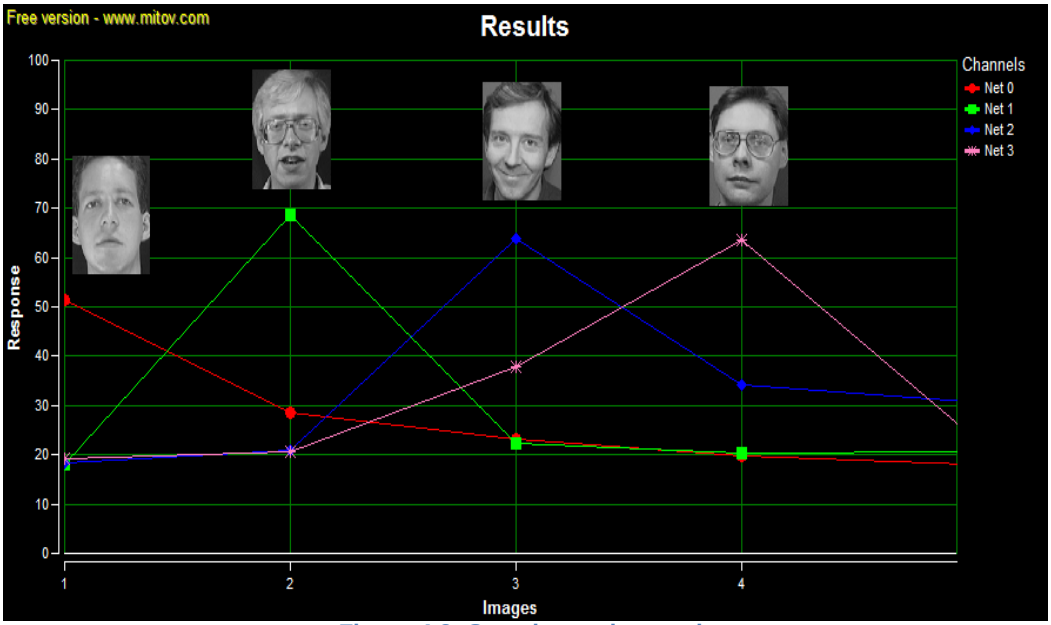


Figure 4.2- Sample results graph

It should be noted that although the graph above represents the results in a continuous form, the responses of each of the images are independent, as displayed in Figure 4.1. In the graph, the X-Axis depicts the images that have been tested on, and the Y-Axis presents the response achieved. The 'Channels', on the top right corner, is a key for the graph; it lists the number of nets that the system has been trained on. Each net has a colour code, and this colour allows the results of the images on each individual net to be followed on the graph.

On this sample graph, it can be observed that image 1 is classified as being part of Net 0, since testing against that net achieves the highest response, compared to the other nets. Image 2 is classified as being part of Net 1, and so on.

The False Acceptance Ratio (FAR), False Rejection Ratio (FRR), and the overall accuracy of the system is then calculated using the formulas presented earlier in this

chapter. Since each image is classified correctly in the sample data, the overall accuracy of the system is 100%, and both the FAR and FRR are 0%.

It should be noted that in all the tests performed in this thesis, the full range of responses are used without thresholding. This aspect will be discussed later in Chapter 6.

When any test is performed, another aspect of the results that is investigated is the confidence achieved by the system. Confidence relates to the percentage difference between the response of the correct net on an image and the maximum response of the other nets on the same image. Like the results, the confidence achieved on each image is independent to other images tested by the system (displayed in Figure 4.3), however due to the number of images tested upon, it is easier to display the confidence in a continuous form, as shown in Figure 4.4.

Figure 4.3 - Histogram of confidence achieved on the sample data

Figure 4.4 - Sample confidence graph

The X-Axis of the confidence graph represents the same images that have been tested on as Figure 4.2, however the Y-Axis is the percentage difference between the response of the correct net and the maximum false response of that image on the other nets it has been tested on. It is important to note that confidence graphs are only useful if more than one net is tested, otherwise there is no net to compare responses against.

4.2 Random image filter

In order to test the accuracy of the system on image recognition, a database of images from a video taken of a room was created, the objective being to detect when a bird cage was in the field of view of the camera. A horizontal pan and tilt IP camera with a 640x480 pixel resolution was used to take images for the database. Although the camera allows both pan and tilt manipulation, it was placed in the middle of the room with the tilt at a specific position throughout the image capturing process. For the training set, images of the bird cage were used, and sixteen images were taken at different horizontal positions, with the cage fully in the scene.

Sample images from the training set are displayed in Figure 4.5 to Figure 4.7 below:

Figure 4.5 - Sample image from the Random Image Filter Training Set (Bird cage on left)

Figure 4.6 - Sample image from the Random Image Filter Training Set (Bird cage at the centre)

Figure 4.7 - Sample image from the Random Image Filter Training Set (Bird cage on right)

Once the training set was created, the pan was adjusted to the minimum rotation angle, and a video was taken at the same resolution, whilst the camera rotated to the maximum horizontal angle, and then back to the starting position. A total of 232 images were captured, samples of which are shown below in Figure 4.8 to Figure 4.12:

Figure 4.8 - Sample image from Scene 1 and 2of the Random Image Filter Testing Set

Figure 4.9 - Sample image from Scene 3 and 4 of from the Random Image Filter Testing Set

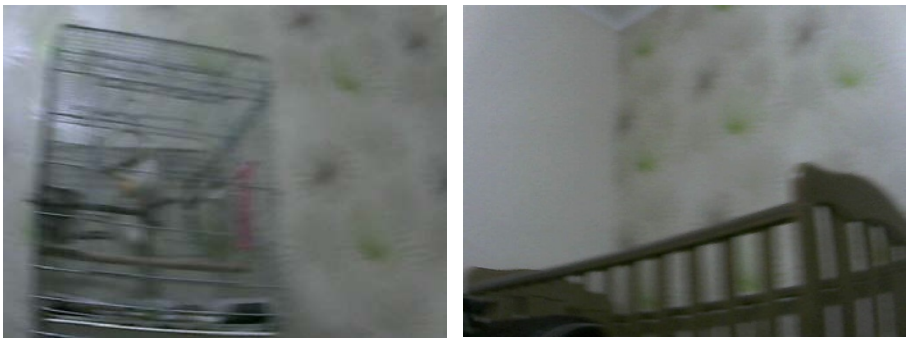


Figure 4.10 - Sample image from Scene 5 and 6 of from the Random Image Filter Testing Set

Figure 4.11 - Sample image from Scene 7 and 8 of from the Random Image Filter Testing Set

Figure 4.12 - Sample image from Scene 9 of from the Random Image Filter Testing Set

As mentioned earlier, the images used in this test were of a 640x480 pixel resolution, and 16 images were used to train the net, and then 232 unique (previously unseen) images were tested against, with 15 images containing the bird cage in its entirety. The 5-pixel n-tuple system was used to test this database, and the results obtained are shown in Figure 4.13.

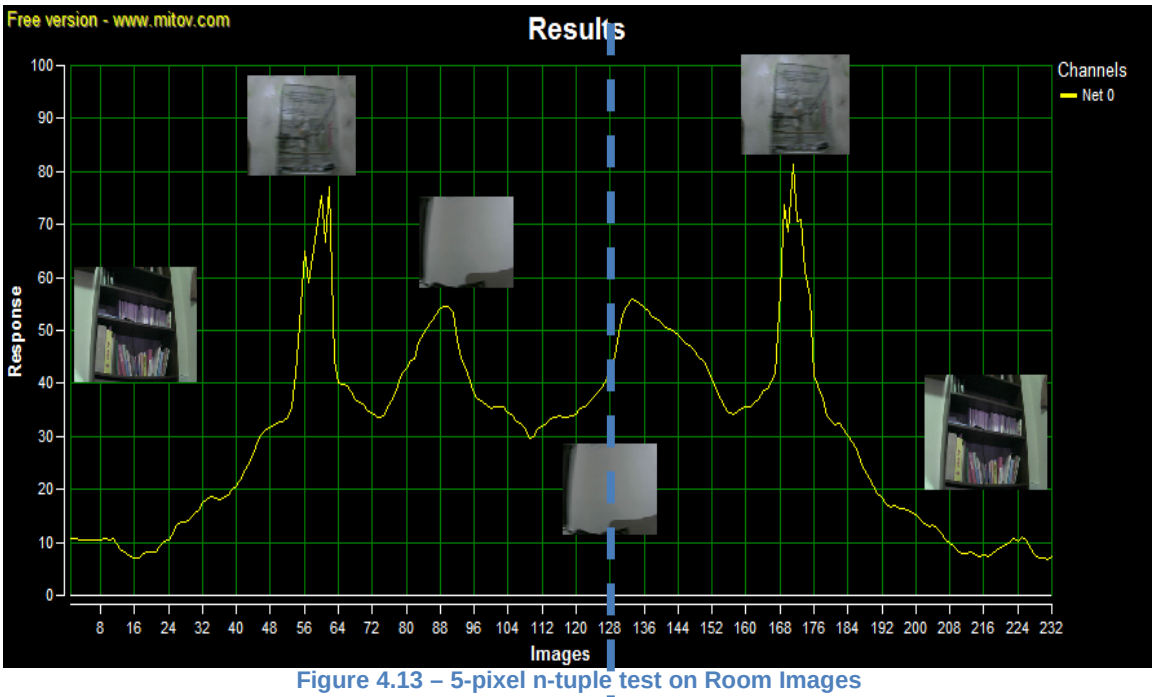


Figure 4.13 – 5-pixel n-tuple test on Room Images

The above graph displays the results obtained on the 232 images, and the graph can clearly be divided into two, with the halves being mirror images. This is because the camera pans from one side to the other and then back to the original position. Minute differences are observed between the results in the two halves, and this confirms that images were not taken in the exact positions both times.

It can be observed in the above graph that as the camera pans horizontally, starting from the scene in Figure 4.8, the lowest overall response is achieved by the system from image 1 to image 16. This is due to the immense differences between the trained images and the tested images, which is pointed out in Figure 4.14.

Figure 4.14 – Sample image from trained net (left) and tested scene (right)

As the camera passes the bookcase shown in the first couple of images, and changes to the light coloured images of the wall and the door, pictured in Figure 4.9, the response rises slightly, See images 24-35 in Figure 4.13.

When the camera pans to capture the door shown in Figure 4.9, the wallpaper changes to that shown in the trained images; resulting in a rise in response on images 36 to 46.

The cage then begins to appear in the frame from image 47 onwards, and once it is entirely present in the captured images, the response is the highest - between images 57 and 63 (see Figure 4.13). It then drops drastically as the camera turns to display scene 6, and then rises once again due to the similar features observed in scene 7 and 8, displayed in Figure 4.11.

The response curve then drops off, as the camera approaches the maximum rotation angle, and the entire result is repeated again backwards, as the camera then turns slowly back to its starting position.

It can clearly be observed in the results shown in Figure 4.13 that the scene trained on is recognized confidently, 100% accuracy is achieved, with the False Acceptance and False Rejection Ratios being 0%.

4.3 Scene detection in live video

The adaptive image filtering methodology was applied to a real-world media scenario. A news video was downloaded off the internet that consisted of 25 different scenes, each scene made up of multiple frames. The video is called Jenkem (SeriousBizness123 and Fox News, 2007) as it is a news clip regarding a type of drug (Jenkem) being abused by American children; the video had a 480x320 pixel resolution, a frame rate of approximately 30 frames per second, and is 1 minute and 55 seconds long. Figure 4.15 to Figure 4.23 below display sample images from each scene in the video file.

Figure 4.15 - Sample image from scenes 1 - 3

Figure 4.16 - Sample image from scenes 4 - 6

Figure 4.17 - Sample image from scenes 7 - 9

Figure 4.18 - Sample image from scenes 10 - 12

Figure 4.19 - Sample image from scenes 13 - 15

Figure 4.20 - Sample image from scenes 16 - 18

Figure 4.21 - Sample image from scenes 19 - 21

Figure 4.22 - Sample image from scenes 22 - 24

Figure 4.23 - Sample image from scene 25

Due to the video being less than 2 minutes long, and to assist in ensuring that the frames in the training set are unique to the frames being tested on, all the frames were extracted from the video without any change in the original 480x320 pixel resolution. The total number of frames that were extracted was 3465. Having captured these frames, the data in the 10 scenes that were chosen to be detected by the system were divided into two, half of which were used in training, leaving a unique set of images to be used to test the system; Table 4.2 below displays the number of frames used for training and testing per scene. All scenes containing people were chosen to train the system, which resulted in 10 trained nets. These were scenes 1 and 2, 10 to 13, 18, and 23 to 25; sample images from these scenes are displayed below:

Figure 4.24 - Sample images from Net 0 to Net 2

Figure 4.25 - Sample images from Net 3 to Net 5

Figure 4.26 - Sample images from Net 6 to Net 8

Figure 4.27 - Sample images from Net 9

Scene	Train	Test	
1	21	21	***Net 0***
2	160	160	***Net 1***
3		38	
4		65	
5		28	
6		57	
7		35	
8		246	
9		124	
10	119	120	***Net 2***
11	65	65	***Net 3***
12	137	137	***Net 4***
13	73	73	***Net 5***
14		102	
15		127	
16		144	
17		192	
18	30	31	***Net 6***
19		103	
20		105	
21		33	
22		115	
23	263	264	***Net 7***
24	85	86	***Net 8***
25	20	21	***Net 9***

Table 4.2 - Number of frames used in Training/Testing Jenkem

The system that was used to train and test the frames used 5-pixel n-tuples and was trained on the first half of the frames of each scene; the frames that were trained on were then omitted from the testing data set, hence the total number of unique frames tested on was 2492. The results graph of the test is shown below in Figure 4.28:

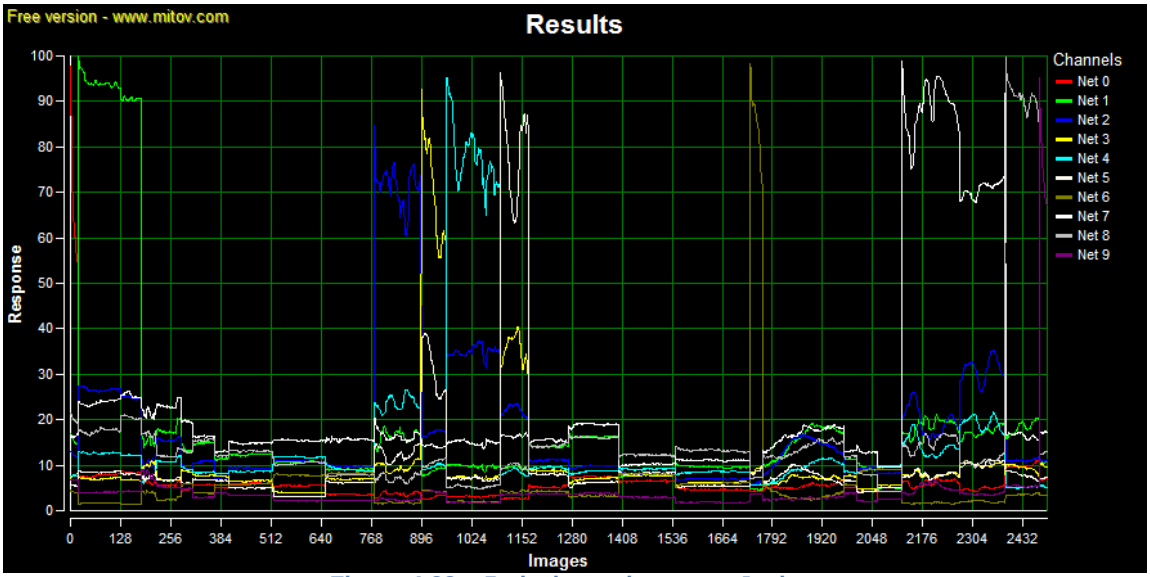


Figure 4.28 – 5-pixel n-tuple test on Jenkem

As can be noticed from this graph, the initial image(s) of each scene have the highest correct response due to the maximum similarity to the training data. The slight drop in percentage response achieved by the other frames in the same scene is due to the fact that the frames from each scene vary from other frames in the same scene, with regards to lighting, angle, and camera position. The average correct response of each net when tested on images from another net is shown in Table 4.3.

		Train On:									
		Net 0	**Net 1**	**Net 2**	**Net 3**	**Net 4**	**Net 5**	**Net 6**	**Net 7**	**Net 8**	**Net 9**
T	**Net 0**	65.56	15.46	12.25	7.74	7.67	5.55	4.04	16.43	20.03	6.75
E	**Net 1**	7.98	93.05	26.08	6.98	12.50	8.38	1.58	24.32	18.53	4.09
S	**Net 2**	3.67	16.29	71.40	9.67	23.63	12.51	1.72	16.49	7.15	2.37
T	**Net 3**	2.98	8.71	17.14	70.34	9.67	31.31	4.26	14.22	10.65	4.18
E	**Net 4**	3.17	9.75	34.99	7.31	77.85	7.01	1.98	14.98	5.26	1.80
D	**Net 5**	2.65	9.40	22.11	35.52	8.48	79.16	4.23	16.43	9.70	3.62
	Net 6	4.24	4.69	6.05	9.03	5.65	8.57	85.07	7.30	7.26	3.62
O	**Net 7**	5.58	18.09	24.71	8.88	16.42	9.01	1.80	80.93	13.18	4.43
N	**Net 8**	9.86	18.30	11.20	9.48	5.21	7.86	3.51	17.35	90.14	5.39
	Net 9	6.77	10.57	11.02	9.98	5.35	7.16	3.40	17.21	12.66	76.13

Table 4.3 - Average responses of 5-pixel n-tuple test on Jenkem video file

In Figure 4.28, it is noticed that both the FAR (False Acceptance Ratio) and FRR (False Rejection Ratio) is 0%, with each scene being recognized accurately, and the graph below in Figure 4.29 displays the data from Table 4.3 proving this. It can also be observed that there is a clear difference between the average correct response and the average false responses in each set of images. The average correct responses can be

seen to range from 70% to above 90%, with false responses being at a maximum of 35%.

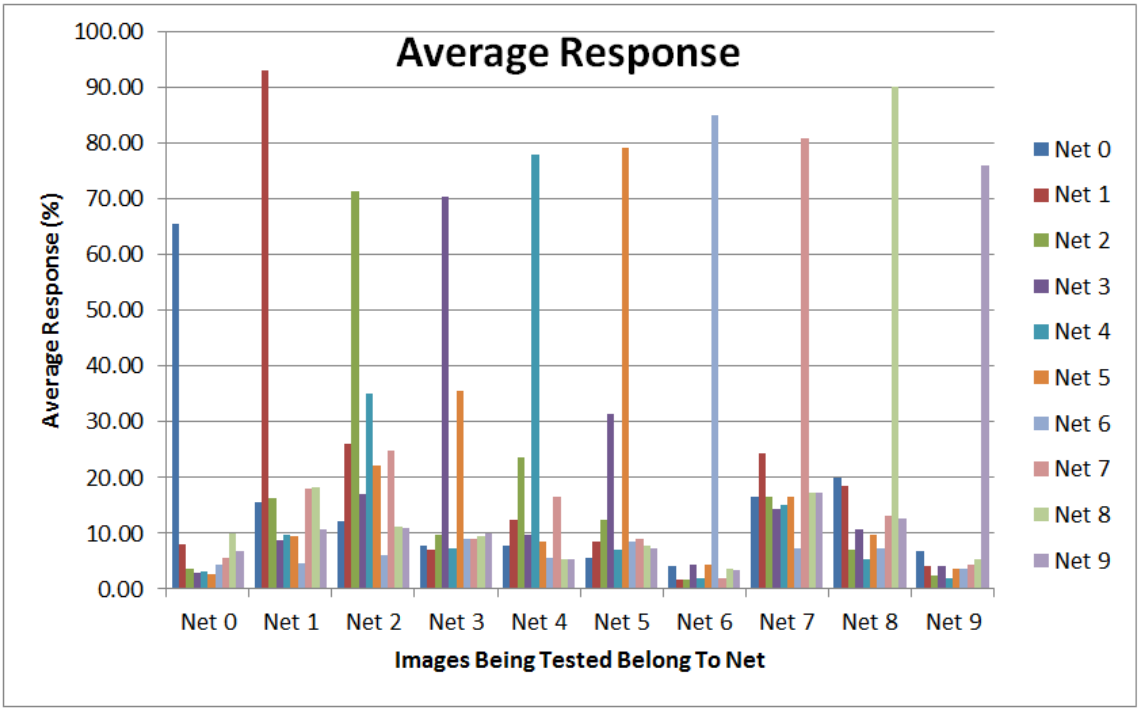


Figure 4.29 - Average responses of 5-pixel n-tuple test on Jenkem video file

The overall average correct response is 81.07%, and the minimum correct response is 54.08%. Regardless of this, the average confidence (the correct response minus the maximum false response) is 54.7%. Take a closer look at the background similarities between Net 3 and Net 5, as shown below in Figure 4.30:

Figure 4.30 - Sample images from Net 3 and Net 5

Even though the similarities between the background pixels in the two images and the two individuals themselves are large, the system still achieves a minimum confidence of 24.92% between these two scenes!

4.4 Conclusion

In this chapter image filter properties of the methodology presented in this thesis have been demonstrated, and the following important properties have been discussed:

1. The system can operate on any data form where similar sets of images occur.
2. No analysis of the images is required.
3. The system can be configured in real-time.
4. Error-free performance has been established on two real-world applications.
5. Real-time evaluation can be achieved on any image classification problem.
6. The system is implementable in hardware or software in real-time.

The chapter also presented the method with which the results will be presented in the later chapters of this thesis, and two real-time tests were conducted. Both tests were performed on frames extracted from real-world video scenarios, and resulted in an overall accuracy of 100%, with the FAR and FRR being 0% in both tests. The next chapter assesses the accuracy of this methodology when tested against face image databases.

Chapter 5 :

Face Recognition

5. FACE RECOGNITION

Face recognition is the most common form of personal identification, and is performed by humans subconsciously on a daily basis. Research has been conducted on this topic for many decades endeavouring to develop a system that is fast, effective and error-free. The scale of the problem is immense (Sinha *et al.*, 2006), and many attempts in the past have failed (Zhao *et al.*, 2003). The nature of the failure is that it is an attempt to provide analysis on data when the actual features are not known.

Normal mathematics might not apply to images, and there is evidence that this is a justifiable statement (Bremermann, 1971). Humans perform face recognition to a high degree of accuracy, and there is no evidence that the formal mathematics that is used in image analysis takes place in natural systems. This is not to say that similar functions such as frequency sensitivity do not take place, however natural systems do not directly implement Fourier transforms.

There are various problems with regards to real-world implementation, like lighting, camera angle, viewing distance, etc. Another main problem with real-world implementation is the computational intensity required by these systems, and the amount of pre-processing required to be performed on images before they can be tested.

Even though the algorithm presented in the methodology chapter (Chapter 3) of this thesis is one of pattern recognition, and the previous chapter investigated video scene filtering, this algorithm can be applied to faces. As seen in the previous chapter, the methodology presented in this thesis is best suited for video sequences. This enables the system to acquire, at video frame rates, the variation of a perceived category. A prime aim of this thesis is to contrast the face recognition results with those obtained by other researchers, therefore it was decided to use image databases that are available for research purposes, in order to make performance comparisons.

5.1 Databases

Due to the lack of face detection as a pre-processing step, the databases used by the system should consist of images containing few background pixels. Hence, a number of the multiple databases that are available to researchers have been investigated, in order to decide whether or not they fulfil this requirement.

5.1.1 The AT&T (ORL) Database

The AT&T Database of Faces (Samaria and Harter, 1994), formerly known as the ORL Database of Faces consists of 10 different images of 40 distinct individuals and all the images were taken at the AT&T Lab in Cambridge.

The images were taken at different times, with a variety of lighting, facial expressions and facial details, but with a dark background. Each image has a size of 92x112 pixel resolution, and is grayscale with 256 grey levels per pixel.

Figure 5.1 - Some images from the AT&T Database

The original results published by Samaria and Harter had a maximum overall accuracy of approximately 99.8% (Samaria and Harter, 1994), with other results ranging from 90% to 95.4% (Wang *et al.*, 2011; Thamizharasi, 2010; Vatsa, Singh and Gupta, 2004). Sample images from five of the individuals in the AT&T Database are in Figure 5.1 above.

This database was deemed appropriate for training and testing the system, as the images consisted of few homogeneous background pixels, and the results of this test is in the results section of this chapter below.

5.1.2 Caltech Faces Database

The Caltech Faces Database is a fully frontal face dataset collected by Markus Weber at the California Institute of Technology. It consists of 450 images of approximately 27 unique individuals under different lighting conditions, with various expressions and backgrounds. (Weber and Caltech, 2005)

Each image is 896x592 pixels in size, and four sample images from the dataset are below in Figure 5.2:

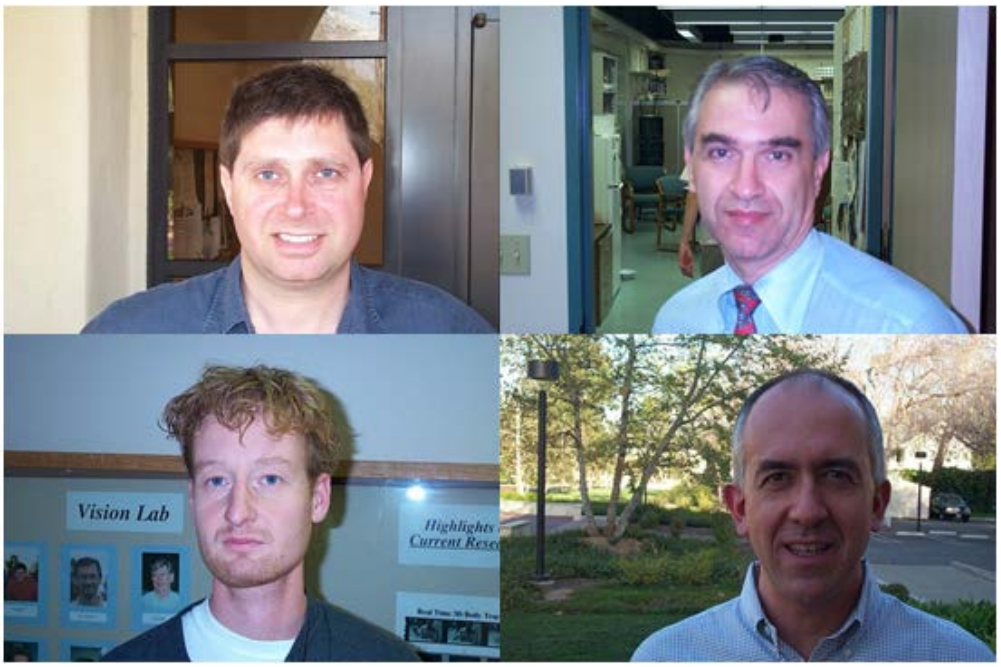


Figure 5.2 - Some images from the Caltech Faces Database

> "The Caltech database is again geared more toward face detection and alignment rather than face recognition. It provides the position of four facial features, but does not give the identity of individuals. Thus, it is not particularly suitable for face recognition experiments." (Huang *et al.*, 2007)

Although the number of images is high, the quality of each image is good, and that all the images are frontal images, a problem arises with regards to the background. As each individual has multiple backgrounds in their individual datasets, and the background makes up of approximately half of each image, this causes a problem for the methodology, and so this database is not going to be used in this research.

5.1.3 Colour FERET

The Colour Facial Recognition Technology (FERET) database (Phillips *et al.*, 2000; Phillips *et al.*, 1998) contains 14126 images of 1199 individuals, and 365 duplicate sets of images. Each image was taken in a semi-controlled environment, however for some individuals images were taken over a two year time period, and so there are minor variations between the camera setups.

Each image is 512x768 pixels in size, and sample images from three of the image sets are in Figure 5.3 below:

Figure 5.3 - Some images from the Colour FERET Database

Although the images are detailed and high in quality, the number of images in each set varies, where the minimum number of images in a single set is 5. This causes a problem with regards to training and testing, and the fact that the number of background pixels in each image also varies greatly leads to the conclusion that this database will not be used in this research as it requires face detection as a pre-processing step.

5.1.4 Face Recognition Data, University of Essex

The Essex Face Recognition Database consists of four databases – Faces94, Faces95, Faces96, and Grimace.

5.1.4.1 Faces94

Faces94 consists of 153 individuals, 18 to 20 unique images per individual, with each image containing 180x200 pixels. Each individual sat at a fixed distance from the camera, with a standard background behind them, and spoke whilst the images were taken in order to introduce facial expression variation. (Spacek, 2007a) Some statistical approaches to facial recognition have achieved an overall accuracy ranging from 85% to 96.67% (Karim *et al.*, 2010; Ding and Feng, 2009).

Sample images from three of the subjects in the database are below in Figure 5.4:

Figure 5.4 – Some images from the Faces94 Essex Database

Since the images have very little background, and the background itself is fairly similar, therefore this database will be used to gauge the accuracy of the algorithm. The results of this test are in the results chapter.

5.1.4.2 Faces95

This database consists of 72 individuals, with the same (180x200 pixel) size as the faces94 database with 20 images per individual; however, there are multiple variations of zoom in the images of each individual. (Spacek, 2007b) Sample images from three of the individuals in the database are below in Figure 5.5:

Figure 5.5 - Some images from the Faces95 Essex Database

Even though the background is the same throughout the database, because of the difference in camera position in each set of images, this database is not to be used for initial testing of the algorithm.

5.1.4.3 Faces96

The faces96 database consists of 152 individuals, with 20 images per individual, however the image resolution is now 196x196 pixels. As in the faces95 database, multiple variations of zoom are produced by changing the camera position within each set of images. (Spacek, 2007c)

Sample images from this database are in Figure 5.6 below:

Figure 5.6 - Some images from the Faces96 Essex Database

This database is similar to the faces95 database but with a larger background in some images. Since the number of background pixels changes between images, this database was deemed inappropriate for initial testing of the algorithm.

5.1.4.4 Grimace

This database only contains 18 distinct individuals, with 20 images per individual, each with a 180x200 pixel resolution. The images in this database have an even lower background pixel percentage than the faces94 database, and it was created to display the difference between facial expressions. (Spacek, 2007d) Results achieved on this database using transformed shape features range from 73.3% to 100% overall accuracy (Biswas and Biswas, 2011).

Sample images from the database are below in Figure 5.7:

Figure 5.7 - Some images from the Grimace Essex Database

Even though this database was designed to test for facial expression recognition, since the camera is at a fixed position throughout the database, and there are 20 images per individual, it was decided that this database would also be used to gauge the accuracy of the algorithm for face recognition purposes.

5.1.5 Georgia Tech Face Database

The Georgia Tech Face Database contains 15 images per individual, and a total of 50 people. Each image has a resolution of 640x480 pixels, and has various zoom, lighting conditions and tilt. (Nefian, 2005)

Sample images from two of the subjects in the database are in Figure 5.8 below:

Figure 5.8 - Some images from the Georgia Tech Face Database

It can clearly be seen from the images above that the faces in the image make up around 30% of the overall image, and the background is dominant in each image. Due to this, and the variability in zoom, this database was deemed inappropriate for face recognition testing without a face detection algorithm.

5.1.6 Japanese Female Face Database

The Japanese Female Facial Expression (JAFFE) database contains 213 images of 10 female individuals, and was created to facilitate the research into detection of a subject's facial expression. (Lyons, Budynek and Akamatsu, 1999; Lyons *et al.*, 1998) Each image has a resolution of 256x256 pixels, is in grayscale, and the background is consistent throughout the database.

Sample images from 4 of the individuals in the database are shown below in Figure 5.9:

Figure 5.9 - Some images from the JAFFE Database

The database has been used for facial recognition independent of facial expression, with 96.55% overall accuracy (Boughrara *et al.*, 2012), and also to detect facial expressions with an average accuracy of 92% (Tan and Zhang, 2008). As can be observed above, the faces make up approximately 90% of the overall image, and since there is no variation is camera angle and viewing distance, this database was appropriate to test the algorithm on. The results of this test are in the next section of this chapter.

5.1.7 Labelled Faces in the Wild

Labelled Faces in the Wild is a database consisting of more than 13000 images collected from the internet, with only 1680 individuals having two or more images in the dataset. It was designed as a database for studying face recognition in unconstrained environment, providing a large set of faces that vary in pose, lighting, expression, race, age, gender and background, as seen in everyday life (Huang *et al.*, 2007); thus the images vary in size, zoom, background, lighting and quality.

Sample images from this database are below in Figure 5.10:

Figure 5.10 - Some images from the Labelled Faces in the Wild Database

As this database contains non-standard images, with many individuals having only one image in their dataset, this database is not meant to be used to test a face recognition that does not implement any form of face detection, hence it will not be used to test the algorithm presented in this thesis.

5.1.8 MIT-CBCL Face Recognition Database

The MIT-CBCL database contains face images of 10 individuals, and was designed for methodologies that construct 3D morphable models of the faces being trained on. In the paper used to present this database, the training faces are decomposed "into a set of components that are interconnected by a flexible geometric model. Changes in the head pose mainly lead to changes in the position of the facial components which could be accounted for by the flexibility of the geometric model." (Weyrauch *et al.*, 2004)

As 3D models are created during training, the process is slow and involves manual interaction, where 3 images of a person's face are used to compute the model, and a large number of synthetic faces are then generated to train the system. A face detection process is used prior to the recognition phase, and Weyrauch and Heisele achieved approximately 88% overall accuracy.

The dataset is divided into three categories.

5.1.8.1 Training Original Folder

This category contains 59 high resolution images of 10 individuals, with 5 to 7 images per person. Sample images from two of the individuals are displayed in Figure 5.11.

Figure 5.11 - Some images from the MIT-CBCL Training Original Database

These images are meant to be full facial profiles of the individuals, and were used by MIT to develop synthetic images of each individual from 3D models, on which they then trained their system. These images are not going to be used for any tests.

5.1.8.2 Training Synthetic Folder

324 synthetic images per subject were rendered from the 3D head models, each with a variation in pose and/or illumination. These images had a resolution of 200x200 pixels each.

A sample image from four of the individuals is shown below in Figure 5.12:

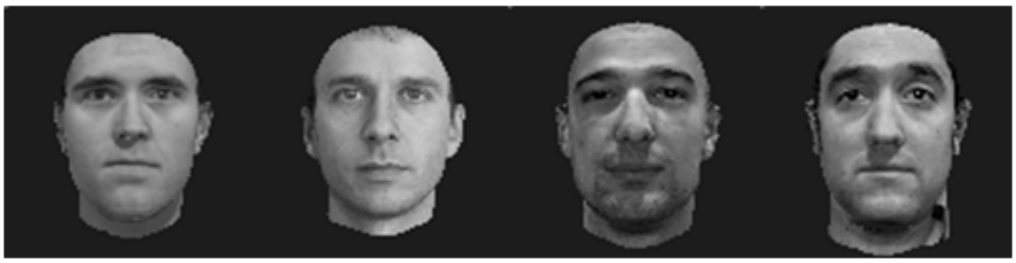


Figure 5.12 – Some images from the MIT-CBCL Training Synthetic Database

These images are not going to be used in any of the tests conducted on the algorithm presented in this thesis based on the fact that they are synthetic, and have been produced from 3D computer models of the individuals.

5.1.8.3 Test Folder

The test database of the MIT-CBCL consisted of 200 images of 100x100 to 115x115 pixel resolution of each individual, 10 subjects in total. These images vary slightly in background, zoom, illumination, and camera angle; however the face in each image consists of 90% of the image, as seen in the examples in Figure 5.13 below:

Figure 5.13 - Some images from the MIT-CBCL Test Database

This database, although meant only for testing was deemed to be a challenge for the system due to the constant change in the image conditions. Therefore it was decided that the algorithm would be tested on this database to test the limits of flexibility of the system on the training data.

5.1.9 Yale Faces B

The Yale Face Database B contains 5850 images of 10 individuals, each with 65 different illumination conditions of 9 poses, including ambient illumination. The images are all high resolution, at 640x480 pixels, and are in grayscale. The database was created to test a methodology that produced illumination cone models for face recognition under variable lighting and pose, using a small number of training images to "synthesize novel images under changes in lighting and viewpoint." (Georghiades, Belhumeur and Kriegman, 2001)

Sample images from two of the subjects in the database are shown below in Figure 5.14:

Figure 5.14 - Some images from the Yale Faces B Database

This database is not meant for the algorithm presented in this thesis as each image captures the face in a different angle, or with a different illumination, and so valid results cannot be achieved by training on half of the images per individual. Georghiades and Belhumeur manually cropped the images for training, to include only the faces with no hair or background; hence, this database was not used to gauge the accuracy of the system.

5.2 Results

The 5 databases chosen for training and testing are:

1. The AT&T Database
2. The Essex Faces94 Database
3. The Essex Grimace Database
4. The JAFFE Database
5. The MIT-CBCL Test Database

The system used to train/test these databases consisted of 5-pixel n-tuples and used random mapping to capture individual grayscale pixel values, as explained in chapter 3 of this thesis. The results of these individual tests are explained in the sub-sections below.

5.2.1 AT&T Database

The AT&T Database consisted of 40 individuals, with 10 92x112 pixel images per individual. As each set is too small to adequately train a class, each of the 40 nets were trained on 9 unique images of that subject, which meant the test set contained 1 unique image per individual, 40 images in total.

The overall results obtained from this test are displayed in Figure 5.16, however to explain these results better, the response of the first four nets on the first four images are shown in Figure 5.15.

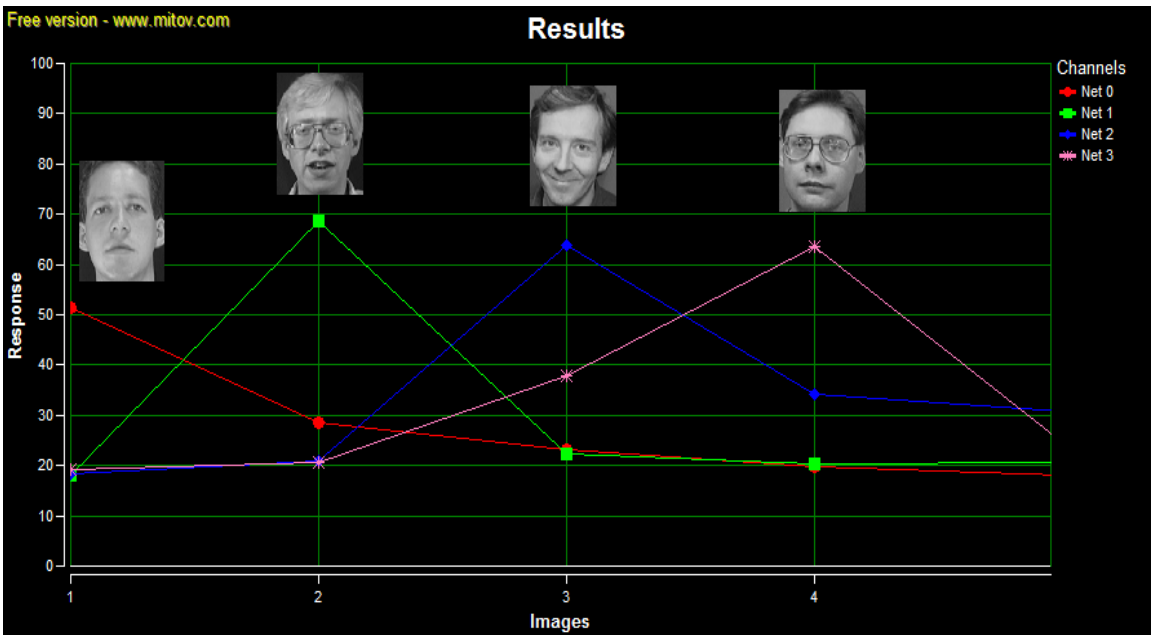


Figure 5.15 - Sample results from the 5-pixel n-tuple test on AT&T Database

As seen in the above graph, out of the four nets the highest response on image 1 is from Net 0, to which it belongs, and the highest response on image 2 is from Net 1, and so

on. This means that none of the images are falsely accepted as being part of another class. The graph containing all the results is below:

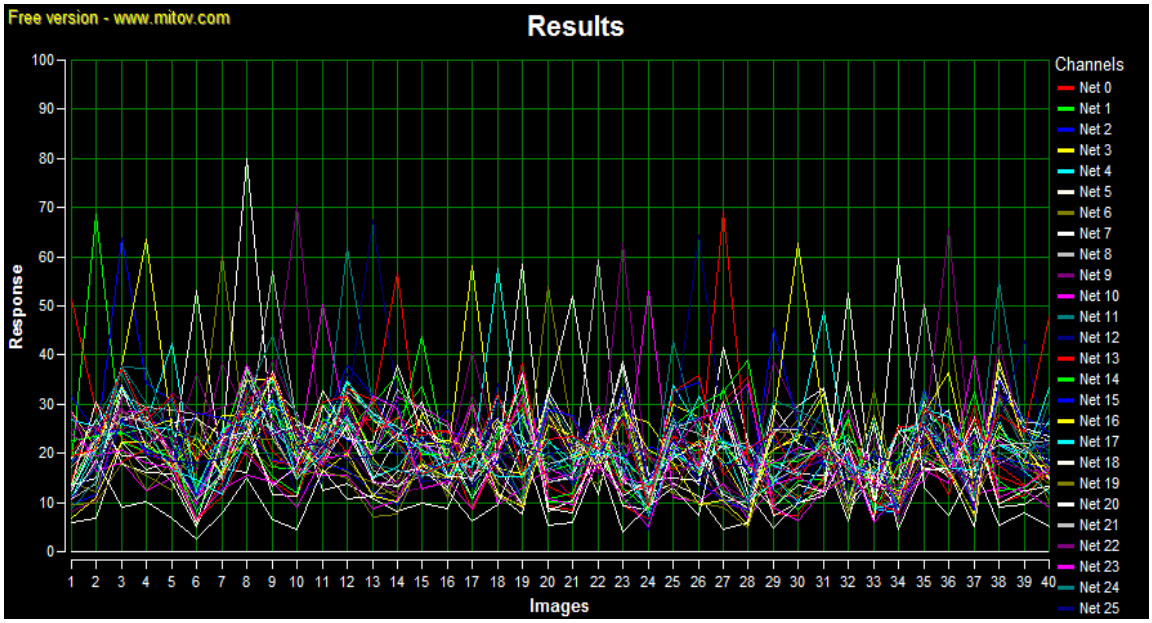


Figure 5.16 – 5-pixel n-tuple test on AT&T Database

Each of the 40 images tested on belong to a different class, hence the desired result consists of a single maximum peak per image, which is the result achieved in the above graph. The average correct response is at 54.9% due to the lack of images to train on, and therefore generalize on, for each class.

The system achieves a 100% accuracy rate, with 0% FAR (False acceptance Ratio) and 100% FRR (False Rejection Ratio), compared to the 99.8% achieved by Samaria and Harter (Samaria and Harter, 1994).

The confidence of the system is the difference between the response of the net that the pattern belongs to, and the highest false response. Therefore if the confidence value is negative, it means that the image being tested has not been correctly classified. Despite the lack of training images, the confidence graph below reveals the system achieves a maximum confidence of 42.18%, an average confidence of 20.5%, and a minimum confidence of 2.76%.

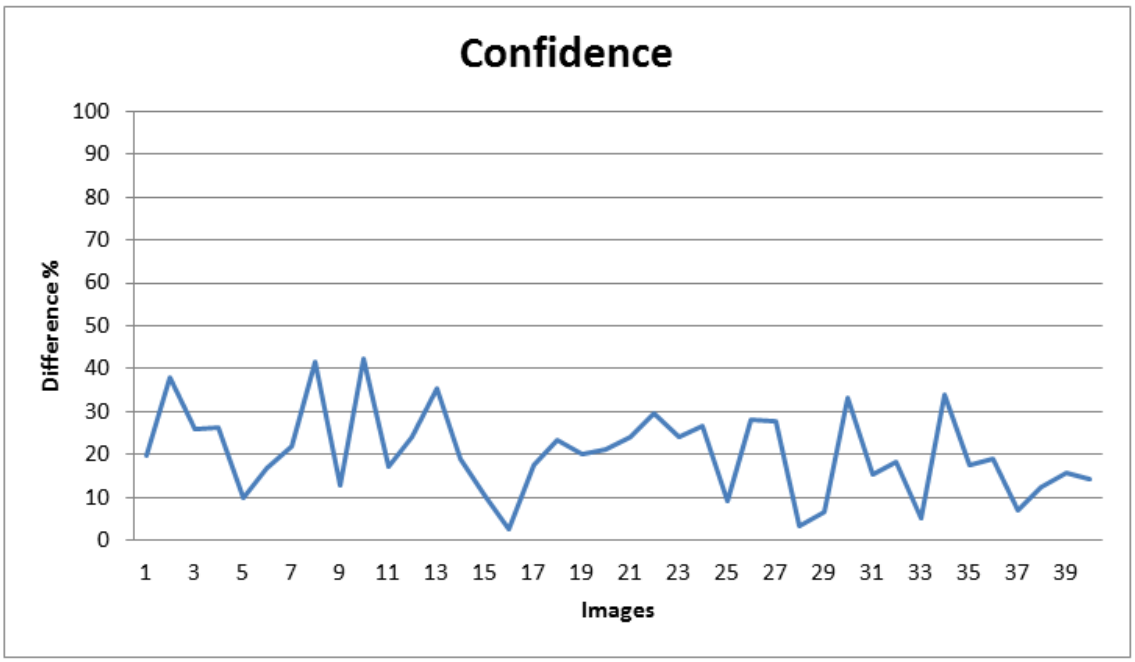


Figure 5.17 - Confidence Graph of test on AT&T Database

The positive confidence rate shows that even when the few images are available to train the system, it is able to generalise within its own class, and differentiate between other classes, if the images in the training and testing set have standard image conditions such as camera angle, background, lighting conditions, etc.

5.2.2 Essex Faces94 Database

The Essex94 Database consists of 152 subjects, with 18 to 20 (180x200 pixel) images per individual. Due to the variability in the size of the sets of images, and in order to standardise the size of the training set, 15 unique images were used to train on each subject, and the rest of the images (3 or 5 depending on the size) were used to test the algorithm.

As with the AT&T Database, the system achieved 100% accuracy on the Essex Faces94 database, with 0% FAR and FRR using randomly mapped, 5-pixel operators. The system achieves an average correct response of 85.6%, with the maximum correct response being 97.93% on the tested images.

The overall results obtained in this test are displayed in Figure 5.19, however as there are over 500 images being tested; Figure 5.18 displays the results obtained for the first 20 images on their nets.

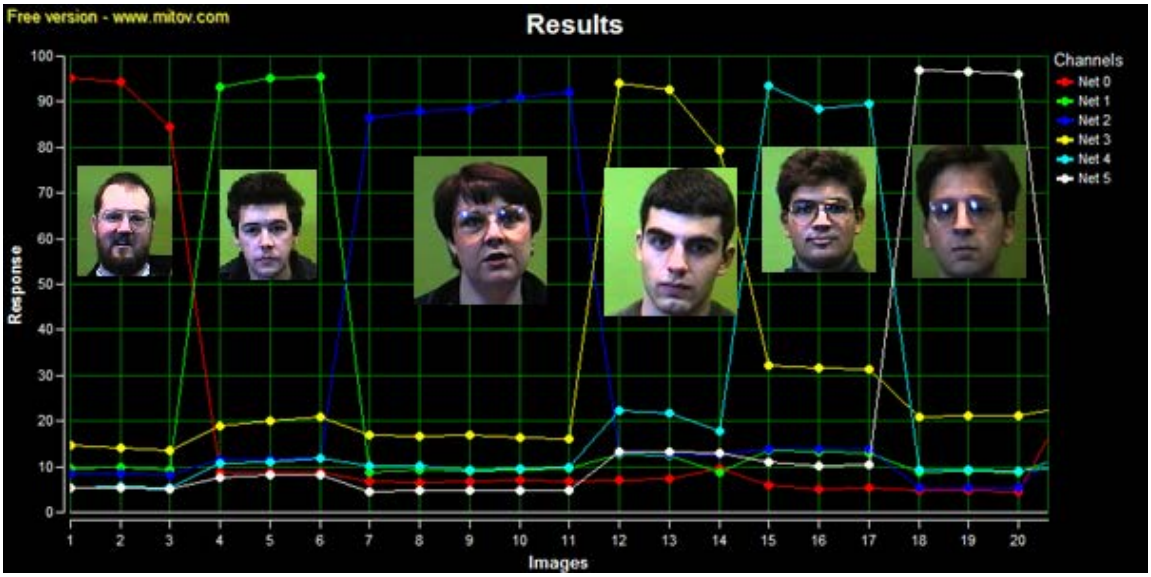


Figure 5.18 - Sample results from the 5-pixel n-tuple test on Faces94 Essex Database

As can be seen in the above graph, for each image being tested, the response from the class it belongs to is the highest, and this is true throughout the test, as shown in Figure 5.19.

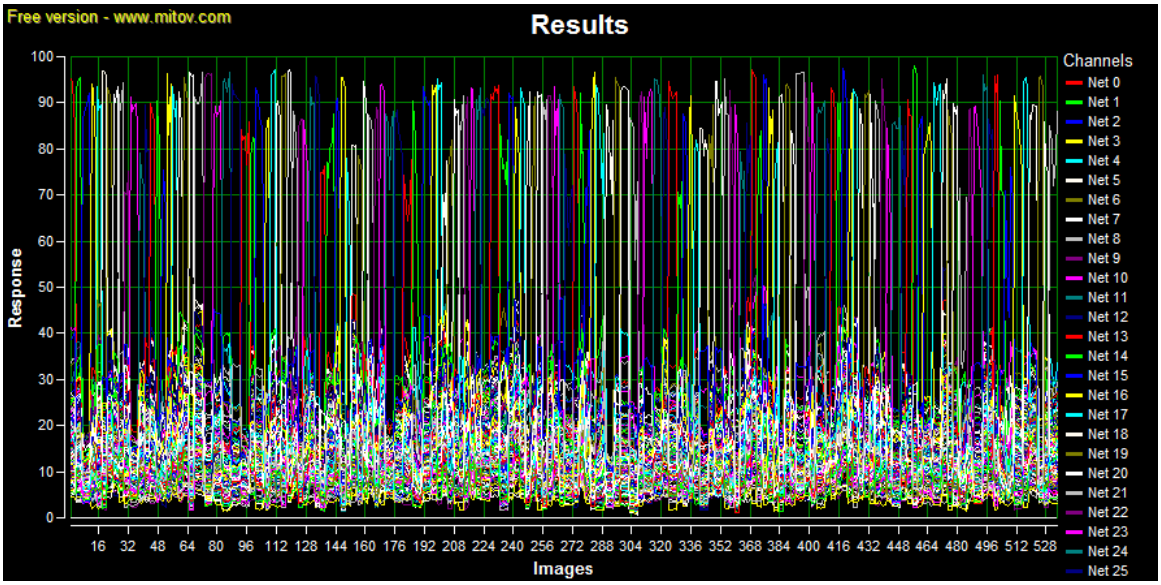


Figure 5.19 – 5-pixel n-tuple test on Faces94 Database

Due to the congestive nature of the results graph above, the correct responses for all the tested images are displayed in Figure 5.20, and the average false responses per image are shown in Figure 5.21.

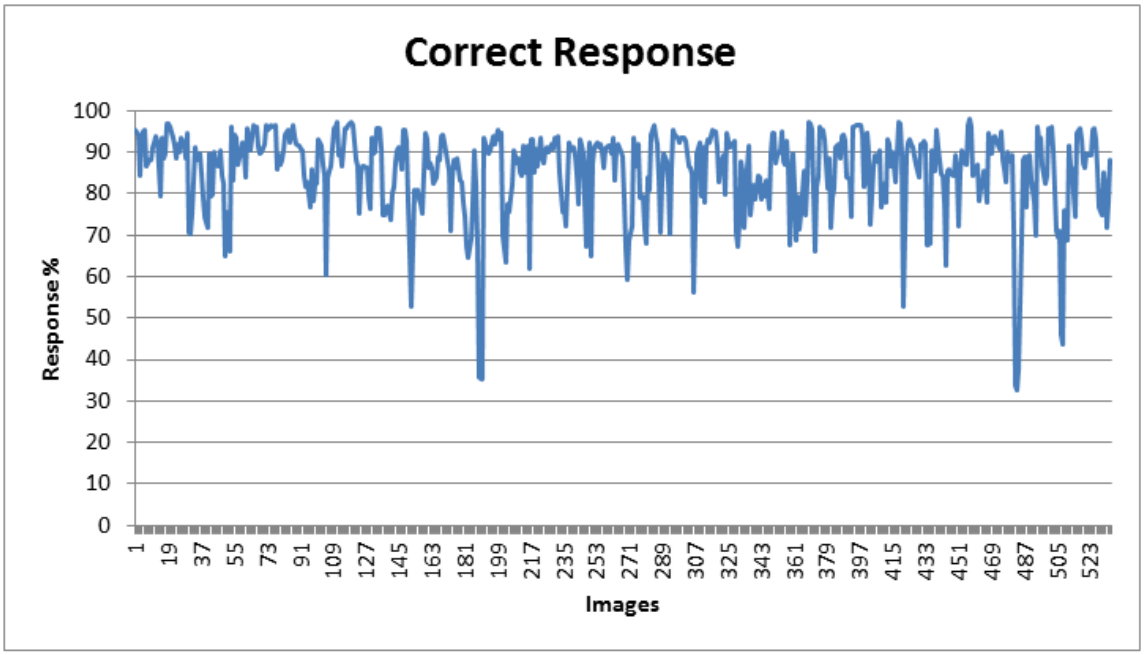


Figure 5.20 - Correct Response across the images in the Faces94 Essex Database

As observed in the two graphs, the correct response is mostly between 60% and 100%, and the average false response is between 10% and 20%.

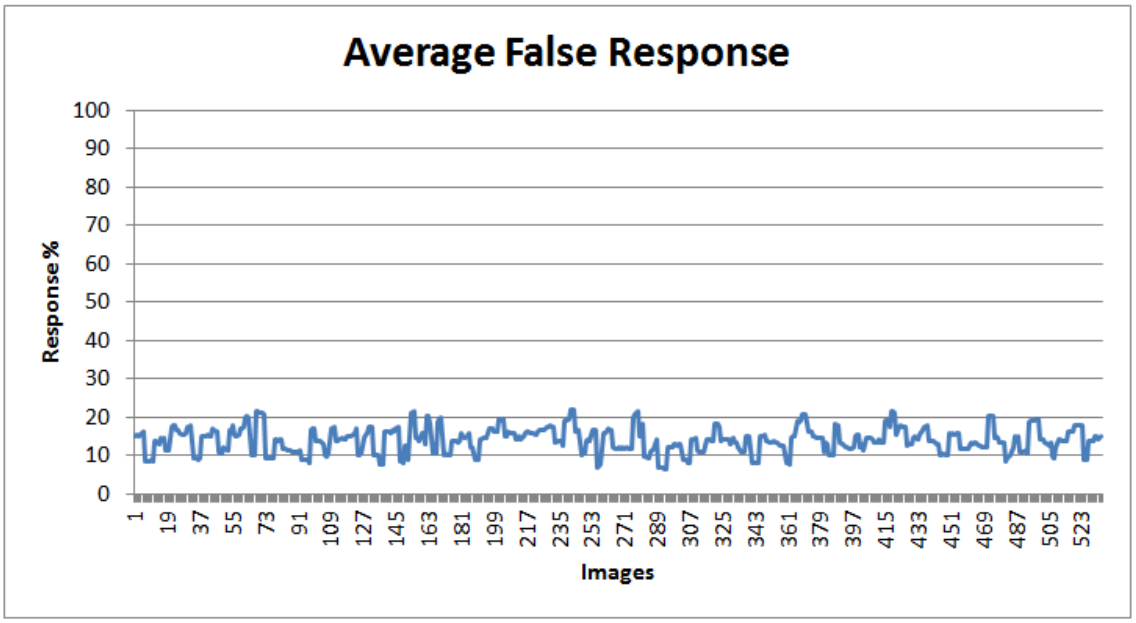


Figure 5.21 - Average False Response across the Faces94 Essex Database

The average confidence of the system is expected to be higher than that of the AT&T test, as it consists of more images in the training set. This expectation is realised, as the system achieves an average difference of 34.5% between the correct maximum response and the maximum false response compared to the 20.5% achieved on the AT&T test, with a maximum confidence of 54.1% compared to the 42.18% on the AT&T test. The overall graph displaying these results is shown below:

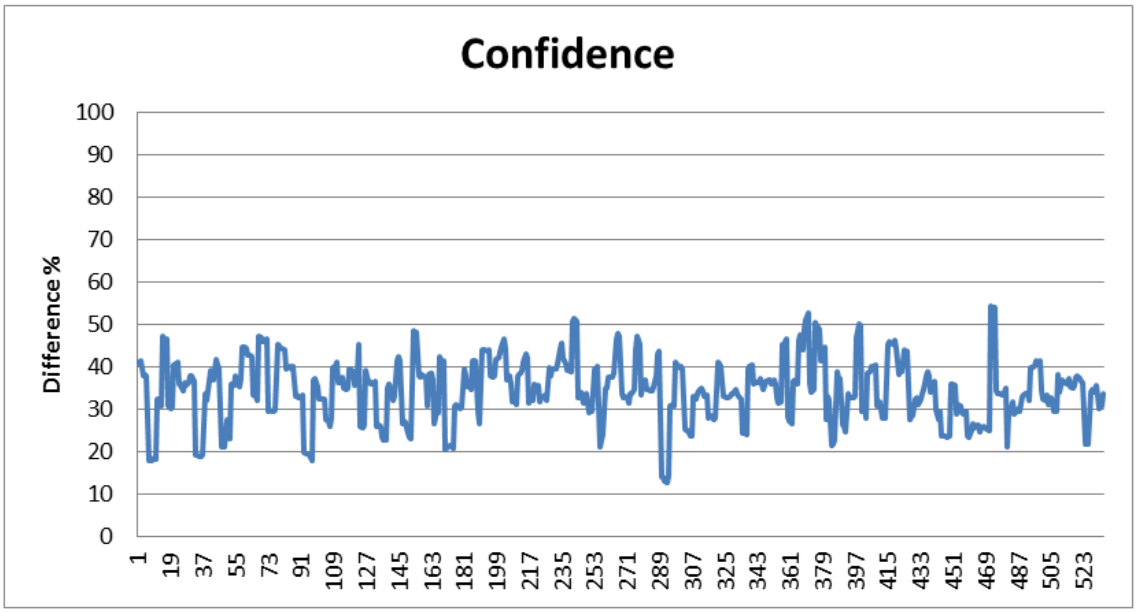


Figure 5.22 - Confidence Graph of test on Faces94 Database

It should be noted that the minimum confidence achieved is 12.61%, compared to the 2.6% achieved on the AT&T test, because more trained images result in greater system generalisation. The overall accuracy of the system is 100%, compared to other methodologies that achieve 85% to 96.6% on the same database (Karim *et al.*, 2010; Ding and Feng, 2009).

5.2.3 Essex Grimace Database

The Essex Grimace Database consists of 18 subjects, 20 180x200 pixel images per individual. Although this database was developed to test for facial expression, hence the name 'Grimace Database', by training and testing on these images, it is being proved that the algorithm can generalize between facial expressions, thus recognising subjects regardless of facial expression.

As each class consists of 20 images, the training data set consisted of 15 unique images per subject, and the nets were tested on 5 unique images per individual. The entire result of this test is below in Figure 5.24, and Figure 5.23 displays results of the first 20 images when tested on the first four nets.

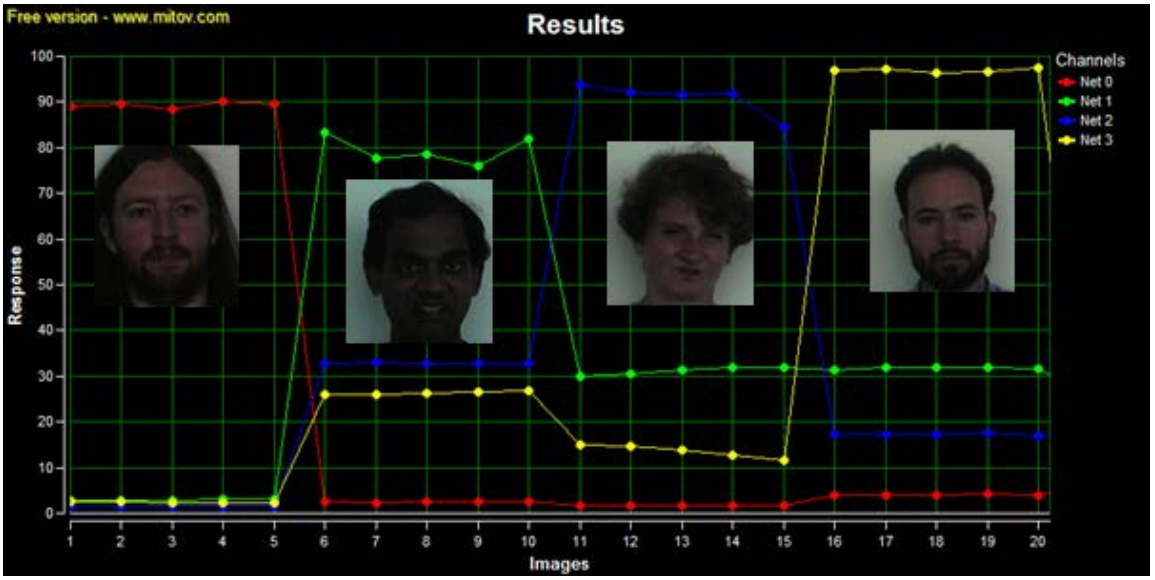


Figure 5.23 - Sample results from the 5-pixel n-tuple test on Essex Grimace Database

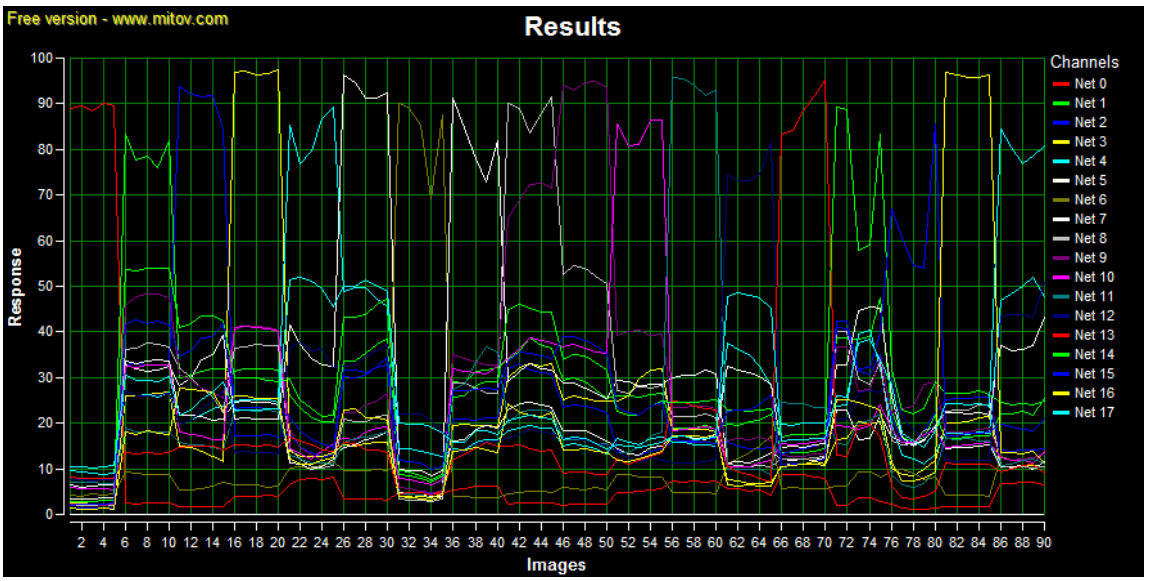


Figure 5.24 – 5-pixel n-tuple test on Grimace Database

As with the previous two tests, the system achieves a 0% error rate in both the FAR and FRR, resulting in an overall accuracy of 100%; each image is correctly recognised as being part of its class, with a maximum correct response of 97.54%, and an average response of 85.6%.

As the images are larger, and there are more images in the training set, this database is expected to have a higher average confidence than the AT&T Database; however, due to a lesser amount of background pixels in the images compared to the Essex Faces94 Database, the maximum and minimum confidence is expected to be higher.

The confidence graph displaying these results is below:

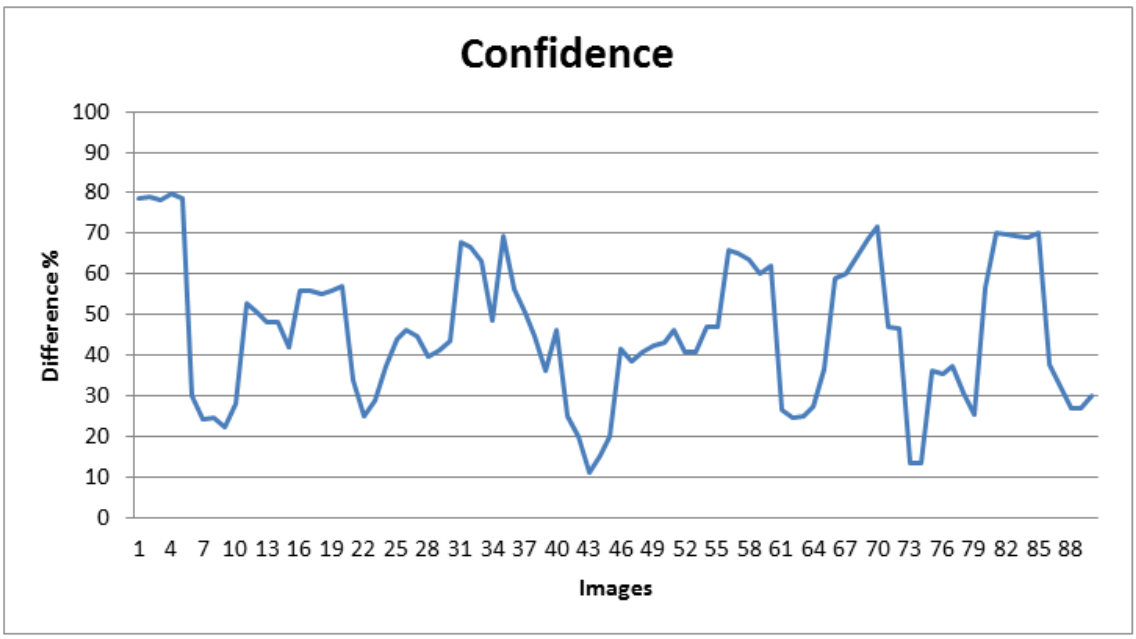


Figure 5.25 - Confidence Graph of test on Grimace Database

These expectations are achieved in the graph shown above, as it can be seen that the maximum confidence achieved is 79.8%, the average is 45.8% and the minimum is 11.06%, whereas the maximum confidence in the AT&T test was 42.18%, the average was 20.5% and the minimum was 2.56%.

5.2.4 JAFFE Database

The Japanese Female Facial Expression Database consists of 20 to 23 256x256 pixel images per class, with 10 subjects in total. Each image has approximately 10% of background pixels, all standard throughout the database. These grayscale image sets were divided into two, with 10 to 12 unique images being used in training, and the rest in testing.

The results graph of this test displays the flawless results of the test on the JAFFE database, once again achieving 0% False Acceptance and False Rejection Ratios, resulting in 100% accuracy on the database. Due to the standardised conditions of the data set, especially the positioning of the subjects in each of the images, and the background pixels, the average and maximum correct response is expected to be less than the previous two tests; the results obtained are displayed in Figure 5.26.

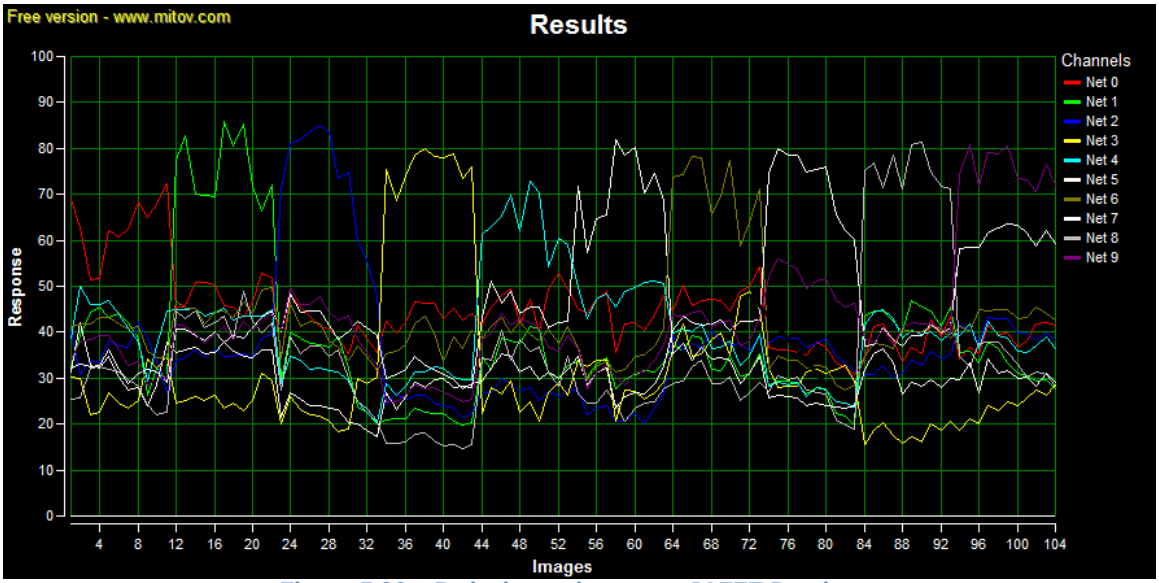


Figure 5.26 – 5-pixel n-tuple test on JAFFE Database

The average correct response is 71.7%, and the maximum correct response is 86%, and both values are approximately 10% lower than those achieved by the system on the Essex Faces94 and Grimace Databases.

The confidence graph displaying the difference between the maximum correct response and the maximum false response per image is shown below:

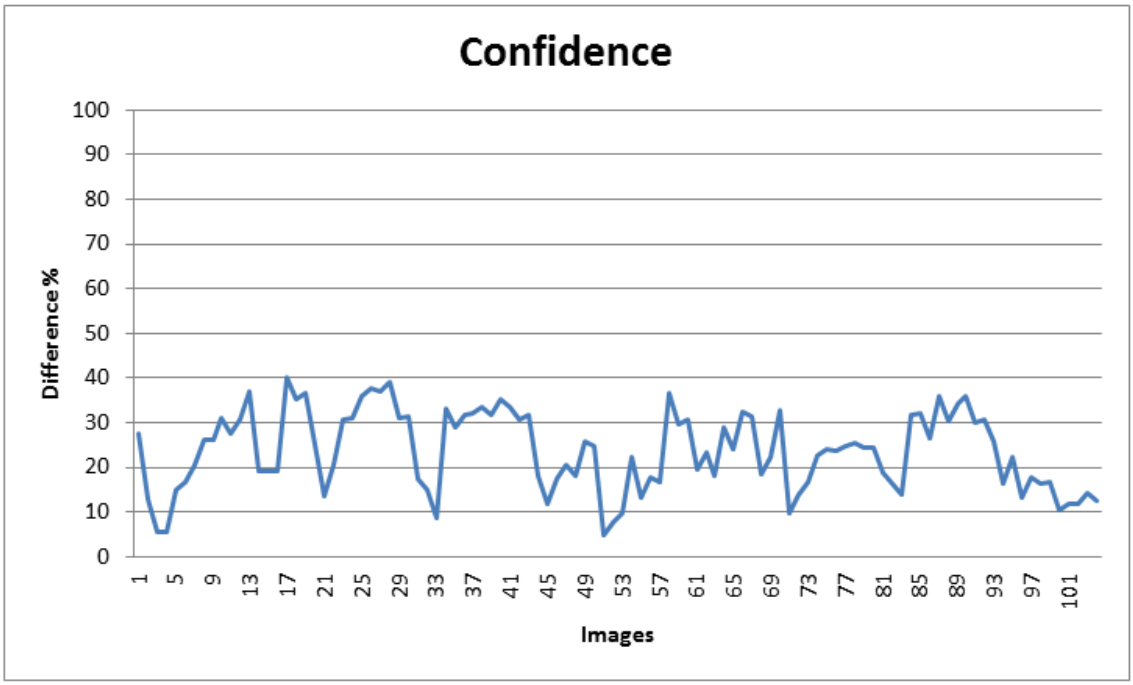


Figure 5.27 - Confidence Graph of test on JAFFE Database

As the images in the database are very similar with regards to camera position, angle, zoom and background, a low minimum confidence is expected, and the system achieves a minimum confidence of 4.71%. The maximum confidence is 40.18%, whereas the average difference is 23.65%. This average is high when compared to the visual similarity of the images in the database.

5.2.5 MIT-CBCL Test Database

The database was originally designed to be used as a test set for the MIT-CBCL database, however due to the training set of the MIT-CBCL database being unusable, as explained in the previous section of this chapter, and the database contained 200 images of 10 individuals, this database was selected to be used to train and test the system.

This dataset was selected to test the limits of the algorithm, as the images within contain variations in illumination, zoom and camera position, and also used different cameras to take the images. An example of this is subject 3 in the training set:

Figure 5.28 - Same images from Subject 3 in the MIT-CBCL Test Database

The images vary in size, ranging from 100x100 pixels to 115x115 pixels. Since all the images being trained on and tested on have to be equal in size due to the size of the random mapping, all the images were resized to 100x100 pixels instead of increasing the size of the other images to 115x115 pixels. The resizing was done by subsample normalization, which does not change the values of any of the pixels in the images, rather it resizes by deleting rows and columns of pixels. Therefore, all the pixels in all the images were exactly the same value as the original images, without any computer manipulation.

The database was then divided into two sets, one for training and the other for testing. Each set contained 100 unique images per individual, therefore there were 1000 images used in training and 1000 in testing. A sample image from each net is displayed below:

Figure 5.29 - Sample images from Subjects 1 - 4 from the MIT Database

Figure 5.30 - Sample images from Subjects 5 - 8 from the MIT Database

Figure 5.31 - Sample images from Subjects 9 and 10 from the MIT Database

The graph of results obtained from this test is shown below:

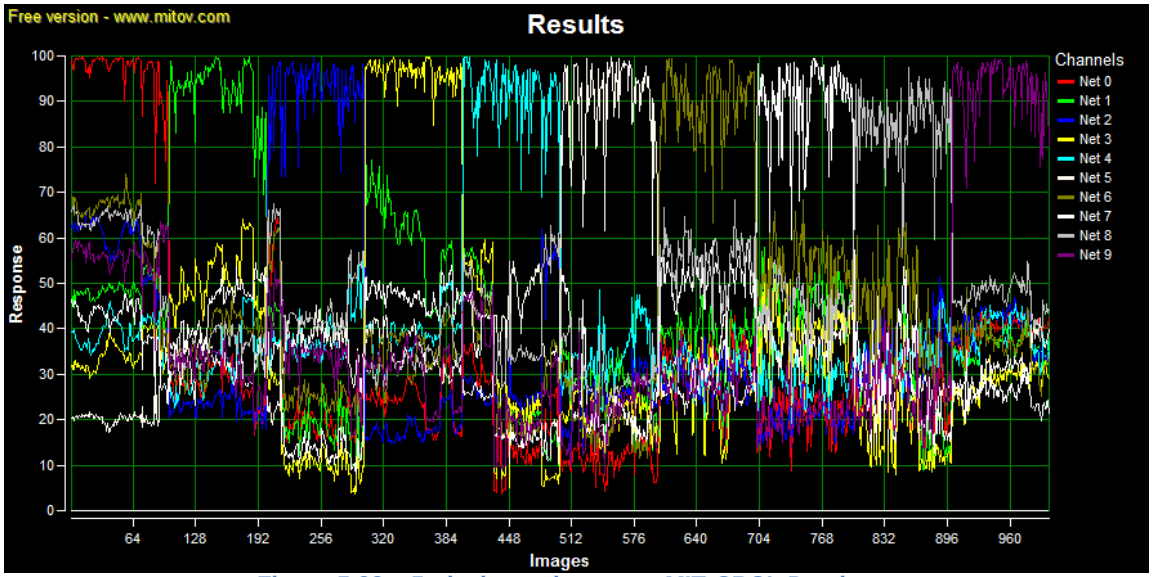


Figure 5.32 – 5-pixel n-tuple test on MIT-CBCL Database

The system achieves a 99.9% accuracy rate, with 0% FRR and a 0.1% FAR, compared to the 88.8% overall accuracy achieved by Weyrauch and Heisele (Weyrauch *et al.*, 2004). The image that is falsely recognised is from subject 3, which is the most diverse dataset in the database. Overall, the average correct response is 92%, with the maximum being 99.8% and the minimum is 56.35%

The confidence achieved by the system is displayed in the graph below:

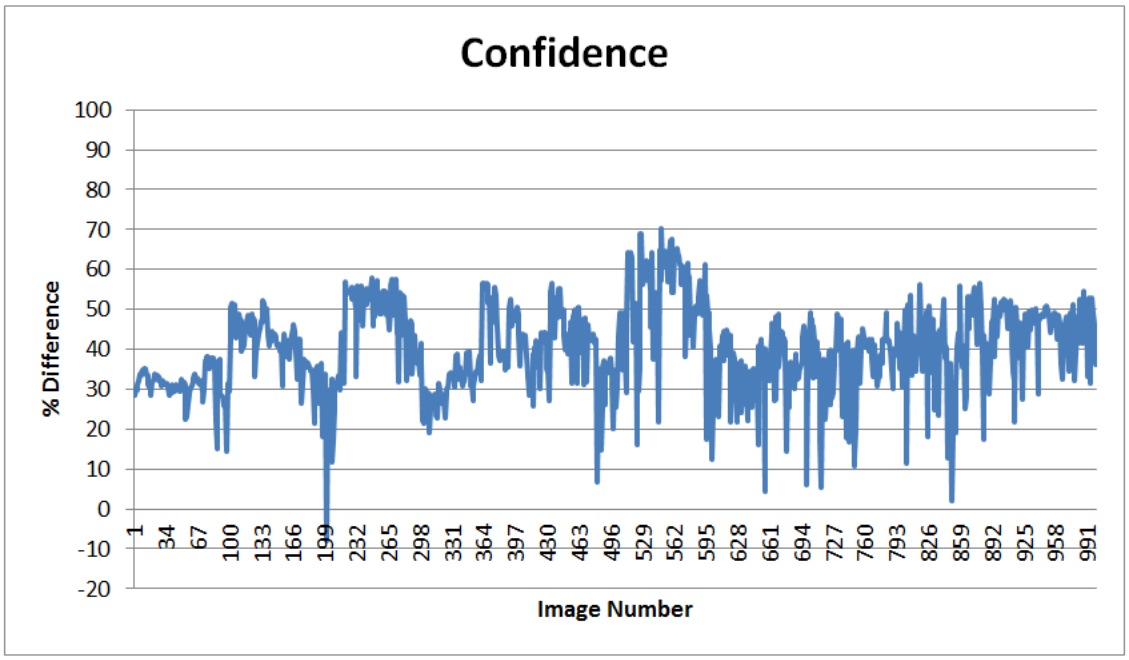


Figure 5.33 - Confidence Graph of test on MIT-CBCL Database

The average confidence of the system is 40.3%, and the maximum is 70.35%. The minimum positive confidence is the lowest of all five tests conducted at 1.95%, and this is due to the multiple variations in zoom, lighting and camera angle. The image that is falsely accepted is image 201, which has a negative confidence of -8.15%.

5.3 Conclusion

The algorithm presented in this thesis is a weightless neural network-based pattern recognition system, hence many face recognition databases cannot be used to test the accuracy of the methodology as they require face detection to be performed on the images before training and testing on the extracted faces.

Regardless of this fact, databases that consist of images with standard or semi-standard conditions, namely, lighting, zoom and camera angle have been trained and tested against, with all the images used in each test being unique and previously unseen by the trained nets. The system achieved an accuracy of 100%, with a 0% False Acceptance Ratio and a 0% False Rejection Ratio on the AT&T, Essex Faces94, Essex Grimace and the JAFFE databases.

After achieving zero error rates in all the previous tests, a database was chosen that contained images of faces in a variety of positions, with various different backgrounds, lighting conditions, camera angle and also with varied zoom. The individual classes in this database were split into two, with half the images being used to train the nets, and the other unseen images being used to test the accuracy of the system.

The chosen dataset was from the MIT-CBCL Database, and the resulting accuracy of 99.9% was achieved by the system, with a 0.1% False Acceptance Ratio. Based on the varied conditions of the images in the database, this result proves the generalization and discrimination capabilities of the algorithm; these two aspects of the system will be discussed in depth in the next chapter.

Chapter 6 :

System Optimization

6. SYSTEM OPTIMIZATION

The algorithm presented in this thesis uses pixel mapping to extract grayscale values of individual pixels, which are then transferred to n-tuples that implement a ranking algorithm. During training, the ranks that are achieved are then stored in the functions that make up the net, and in testing, these ranks are tested against the net, which results in the overall response of the image.

The tests conducted in Chapter 4 and Chapter 5 of this thesis were performed on different image sizes, however all the tests use the same system parameters with regards to the pixel mapping and the n-tuple size. This chapter investigates how adjusting the mapping, varying the n-tuple size, and increasing or decreasing training set size effects the overall accuracy of the system.

6.1 Mapping

Mapping is the process which assigns each n-tuple 'n' pixels from an image, and is of two main types:

1. Linear
2. Random

In order to assess the difference between the two types of mapping, the MIT-CBCL Test folder was trained and tested on by two 5-pixel n-tuple systems, one using linear and the other random mapping. Since the training/testing set images, the image sizes, and the n-tuple size was the same in both systems, this would allow for a direct comparison of the two results.

The MIT-CBCL Test Database is explained in detail in Chapter 5, and consists of 10 individuals, 200 unique images per individual, out of which 100 unique images were used to train each class, and the testing set consisted of 100 unique images per subject.

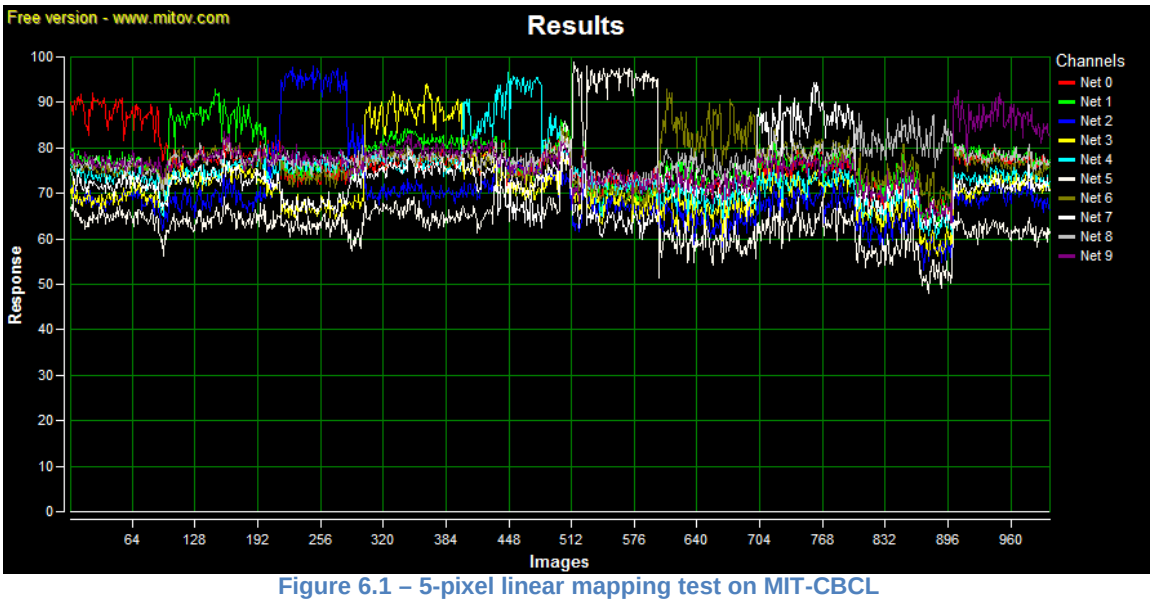


Figure 6.1 – 5-pixel linear mapping test on MIT-CBCL

The results graph above displays the results of the MIT-CBCL test using linear mapping, and it can be observed from the graph that although the correct response per image is between 70% and 100%, the maximum false response is between 70% and 80% which is very high!

		TESTED ON:									
		Net 0	**Net 1**	**Net 2**	**Net 3**	**Net 4**	**Net 5**	**Net 6**	**Net 7**	**Net 8**	**Net 9**
T	**Net 0**	87.14	76.59	68.31	69.14	73.30	64.42	75.35	71.57	75.76	76.53
E	**Net 1**	78.01	86.73	68.88	73.88	75.38	64.38	78.32	73.85	78.26	78.06
S	**Net 2**	75.51	75.88	90.18	67.55	75.76	62.68	74.62	69.16	77.10	77.27
T	**Net 3**	76.91	81.61	70.19	88.29	77.04	65.66	77.81	75.43	78.09	79.55
E	**Net 4**	77.58	78.64	73.54	73.47	88.73	68.44	76.60	71.62	78.07	77.59
D	**Net 5**	74.13	71.90	67.58	69.43	72.86	92.52	71.97	66.19	74.59	73.49
	Net 6	71.86	73.36	63.76	66.45	69.06	59.19	83.96	71.33	76.05	72.06
O	**Net 7**	76.29	77.27	68.54	72.50	72.87	63.59	79.25	86.73	79.01	76.42
N	**Net 8**	69.47	70.30	61.32	63.66	66.36	55.83	72.93	67.75	81.86	69.30
	Net 9	77.55	78.36	69.46	71.93	73.62	61.94	76.83	71.82	78.04	86.51

Table 6.1 - Average response 5-pixel n-tuple linear mapping on MIT-CBCL

Table 6.1 displays the average response of each net across the 100 images tested on, per net. For example, the average response of Net 0, when tested on images from the same class is 87.14%, but when tested on images from Net 1, the average response of Net 0 is 76.59%. From this table it is observed that the highest average response on an image from any net is from the net it belongs to. The average results in Table 6.1 are displayed in Figure 6.2 below.

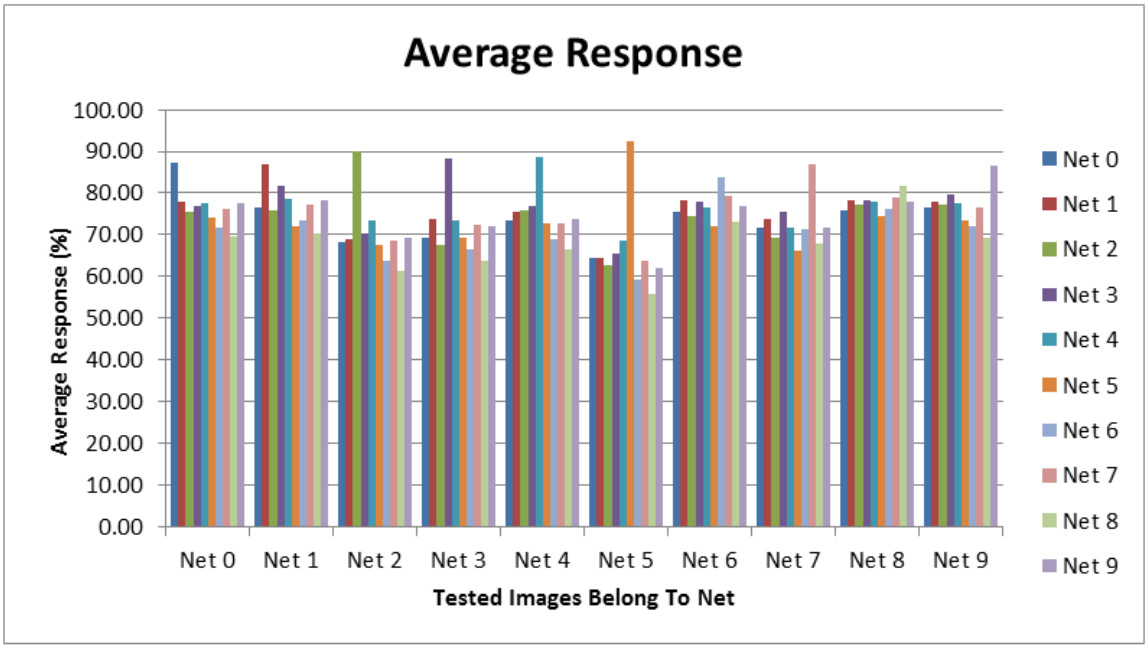


Figure 6.2 - Average response 5-pixel n-tuple linear mapping on MIT-CBCL

The average correct response overall is 87.7%, and the maximum correct response is 99%, however even though these values are high, the confidence graph displayed in Figure 6.3 proves that linear mapping cannot be used in this algorithm.

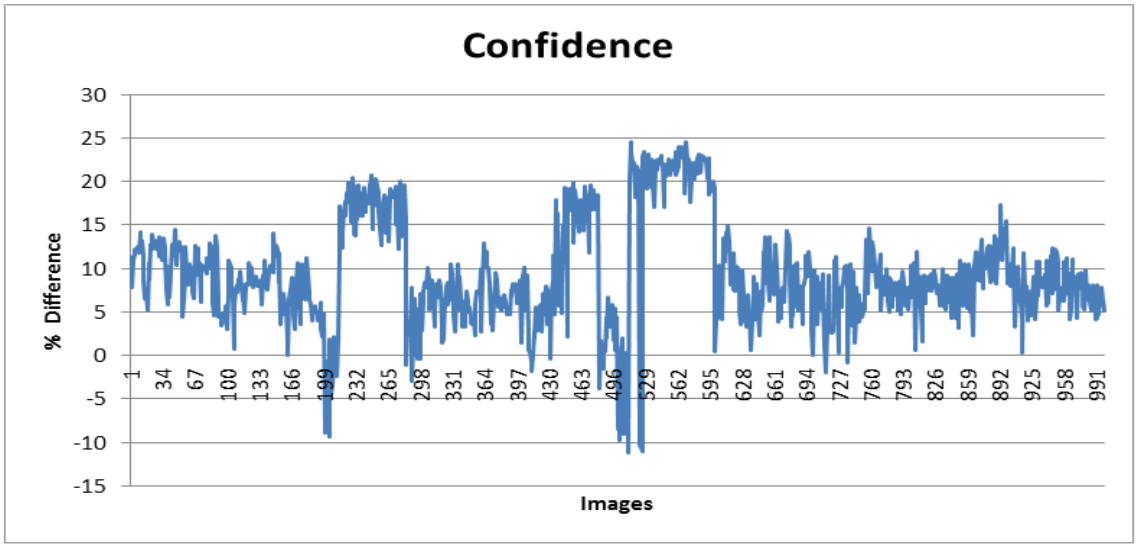


Figure 6.3 - Confidence graph of 5-pixel n-tuple linear mapping test on MIT-CBCL

As can be observed in the confidence graph, there are 36 falsely accepted images, which results in an overall accuracy of 96.4%. The average system confidence is 9.9%, and the maximum difference between the correct response and the maximum false response is 24.5%.

When the same test is run but with random mapping instead of linear mapping, the results in Figure 6.4 below are achieved, with an average correct response of 92%, and a maximum of 99.8%.

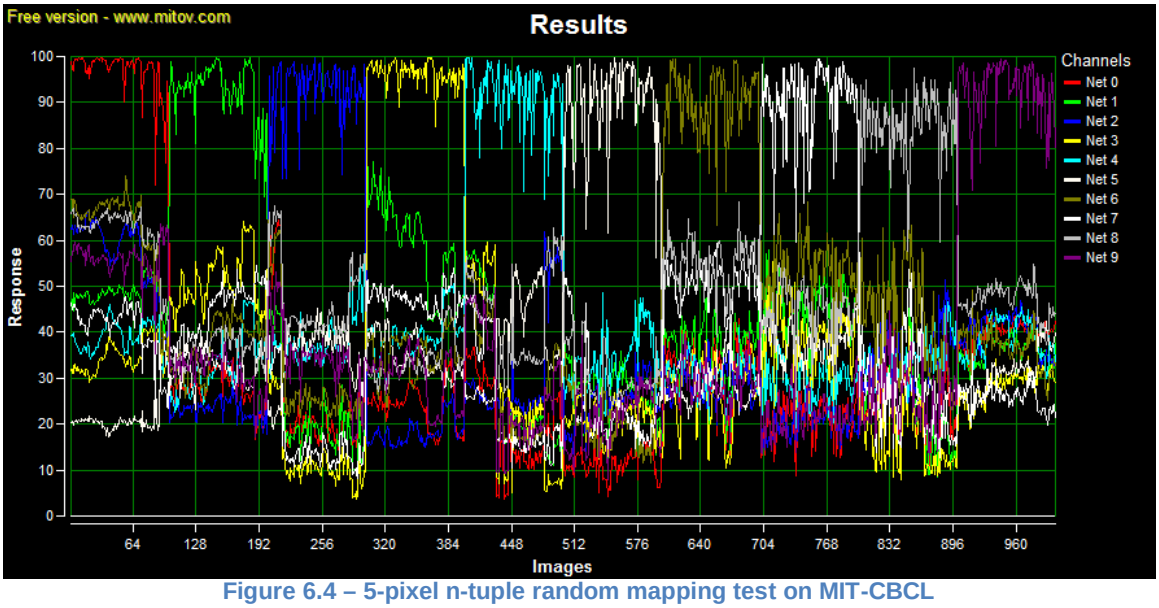


Figure 6.4 – 5-pixel n-tuple random mapping test on MIT-CBCL

The average correct response overall is higher than that in the linear mapping test, due to the average response of each net, when tested on images that belong to it, being higher, as shown in Table 6.2.

		TESTED ON:									
		Net 0	**Net 1**	**Net 2**	**Net 3**	**Net 4**	**Net 5**	**Net 6**	**Net 7**	**Net 8**	**Net 9**
T	**Net 0**	96.82	47.97	57.26	34.46	39.09	21.61	62.77	40.88	61.99	55.95
E	**Net 1**	28.38	92.34	23.39	50.92	32.82	32.54	37.49	44.10	35.93	31.75
S	**Net 2**	25.62	22.38	91.74	12.96	38.09	34.67	30.01	16.93	46.78	34.50
T	**Net 3**	24.03	59.06	18.89	95.67	39.43	34.97	35.37	47.13	33.98	30.33
E	**Net 4**	19.90	29.56	30.92	29.24	91.54	37.45	29.32	24.59	42.56	27.95
D	**Net 5**	11.67	26.53	25.54	21.27	34.25	89.86	21.28	24.33	30.35	21.39
	Net 6	31.87	36.24	28.92	27.75	30.11	28.66	89.67	50.33	54.82	28.87
O	**Net 7**	21.42	44.90	20.45	38.34	28.91	37.72	52.91	89.86	44.89	22.28
N	**Net 8**	24.43	23.42	34.54	17.45	34.48	26.12	42.05	28.08	85.00	26.97
	Net 9	39.79	35.86	40.51	28.38	38.60	31.19	36.71	25.86	47.53	92.40

Table 6.2 - Average response 5-pixel n-tuple random mapping on MIT-CBCL

When the average responses are plotted into a graph in Figure 6.5, it is visible that the average false response is much lower than that attained in the linear mapping test, being within the range of 10% to 60%, whilst the average correct response is nearly always at or above 90%.

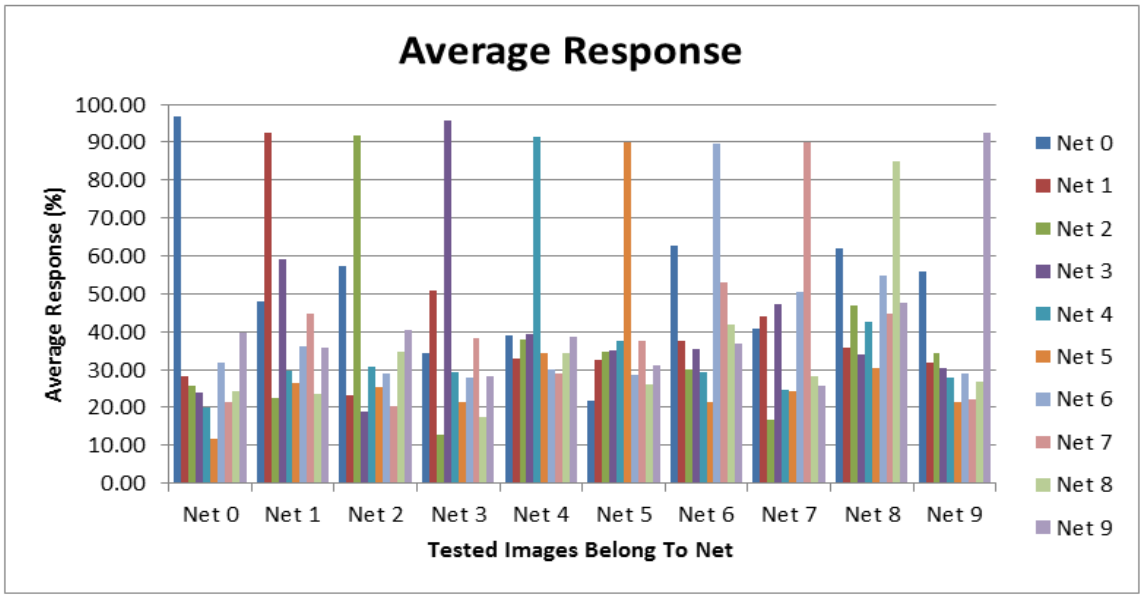


Figure 6.5 - Average response 5-pixel n-tuple random mapping on MIT-CBCL

This is also proved in the confidence graph in Figure 6.6 below, where it can clearly be seen that there is only one image falsely accepted by the system compared to the 36 falsely accepted in the linear mapping test, and that the average confidence is 40.3% whilst the maximum confidence is 70.4%. This is much higher than that of linear mapping, where the average confidence was 9.9%, and the maximum was 24.5%.

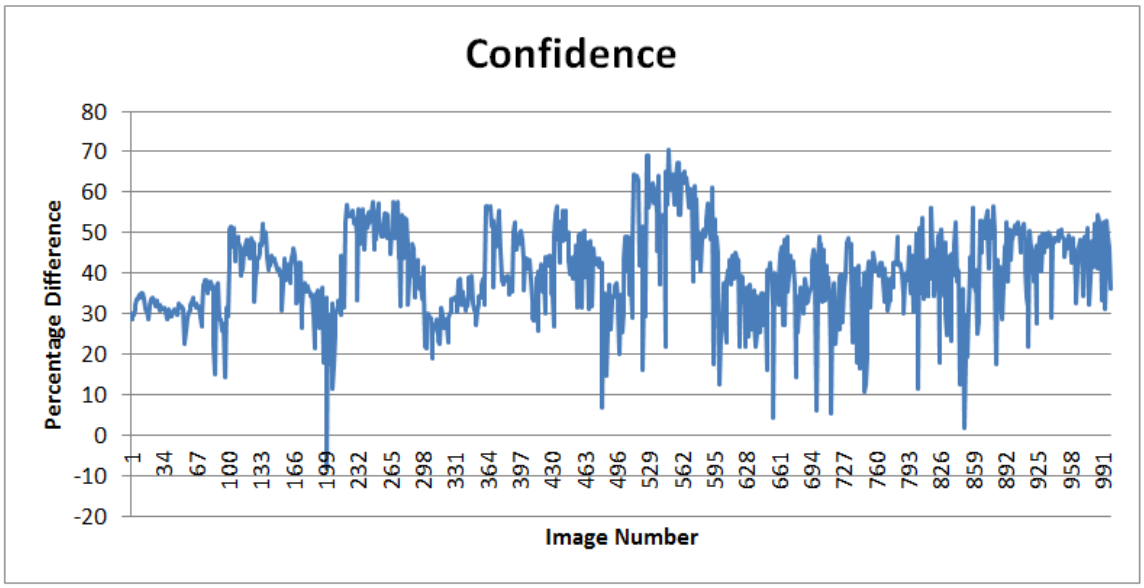


Figure 6.6 - Confidence graph of 5-pixel n-tuple random mapping test on MIT-CBCL

Although the overall system accuracy of the linear mapping system is above 95%, this is 4.9% lower than the random mapping system, and coupled with the low average confidence, linear mapping is not appropriate for pattern recognition purposes.

6.2 The n-tuple size

One of the three main aspects of the algorithm presented in this thesis is the n-tuple size. The effect a change in n-tuple size has on the amount of memory the system requires, has been discussed in Chapter 3, however it is important to investigate what effect changing the n-tuple size has on the responses obtained on the testing data.

To understand how the n-tuple size affects the system, it is first necessary to understand the effect training on one pattern from the class has on the accuracy of the net. Figure 6.7 displays a visual representation of all the data that forms class 'A' (D_A) in the universe.

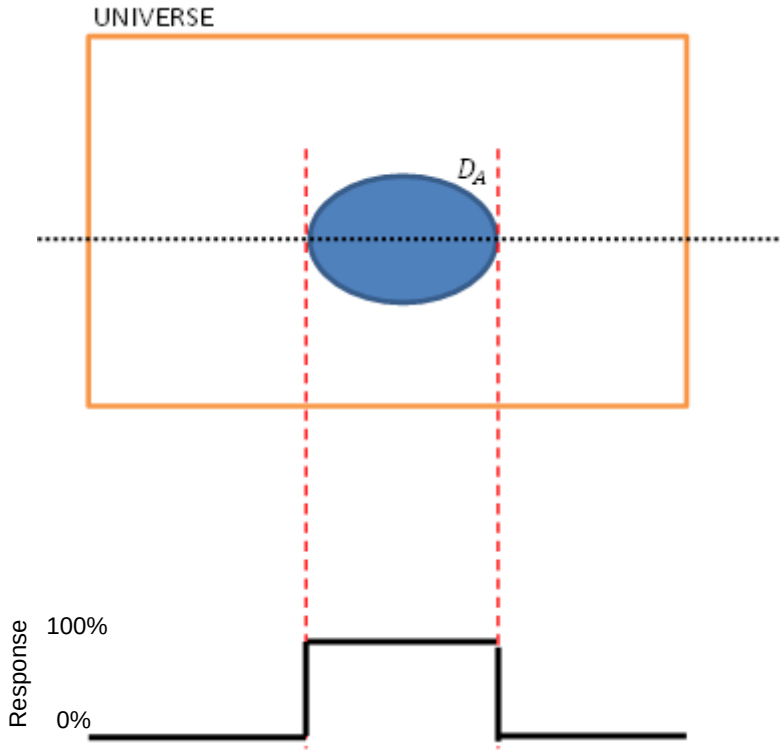


Figure 6.7 - Data set of class 'A' in the Universe, and 1-D representation

The cluster D_A is then represented in one dimension by taking a horizontal profile of the cluster, and representing it by a square wave, as seen in Figure 6.7. In an ideal world, a fully trained net on class 'A' would be represented by the square wave, which would mean any data that is part of D_A would be recognised, and any data that is not part of D_A would be rejected.

However, in a real-world scenario, the effect of training on 1 pattern from D_A is that shown in Figure 6.8, where 100% response is only achieved by the pattern that has been trained on, and a few other (similar) patterns have a lower response.

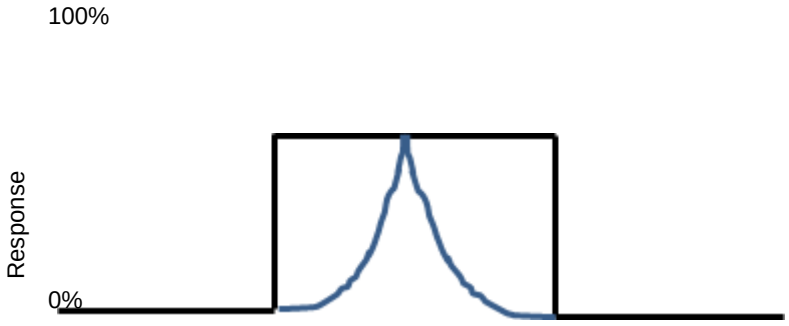


Figure 6.8 - Effect of training one image from class 'A'

Hence by training on multiple patterns (Figure 6.9), the system generalises between all the data, resulting in the graph shown in Figure 6.10.

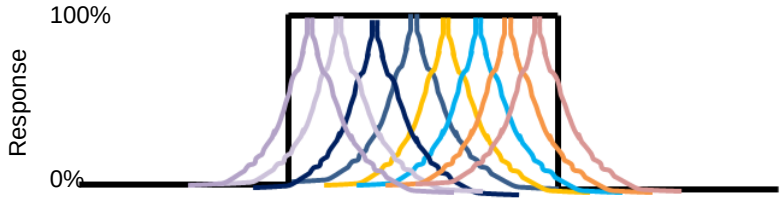


Figure 6.9 - Effect of training multiple images from class 'A'

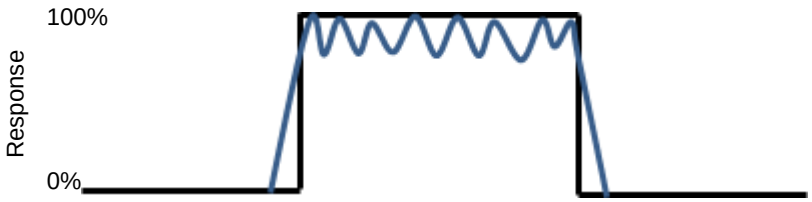


Figure 6.10 – Generalising effect of training multiple images from class 'A'

Although training on more patterns results in a better generalised net, in a real-world scenario, data is sometimes falsely accepted as being part of the class, or false rejected also. This is depicted in Figure 6.11, where pink shading depicts patterns that are falsely accepted, and the yellow shading depicts patterns that have a lower response, and that are, in some situations, falsely rejected.

Figure 6.11 - Accuracy, FAR and FRR areas after training on multiple images from class 'A'

The above graphs depict the generalization effect of training on data from one class, when only one class exists in the universe. If two classes exist in the universe, it is important to calculate the probability of a pattern from one class being classified as being part of the other class. For example if class 'A' and 'B', displayed in Figure 6.12, existed, the similarity between the two classes would be as shown in the shaded area in Figure 6.13, and the difference would be as shown in the shaded area in Figure 6.14 below.

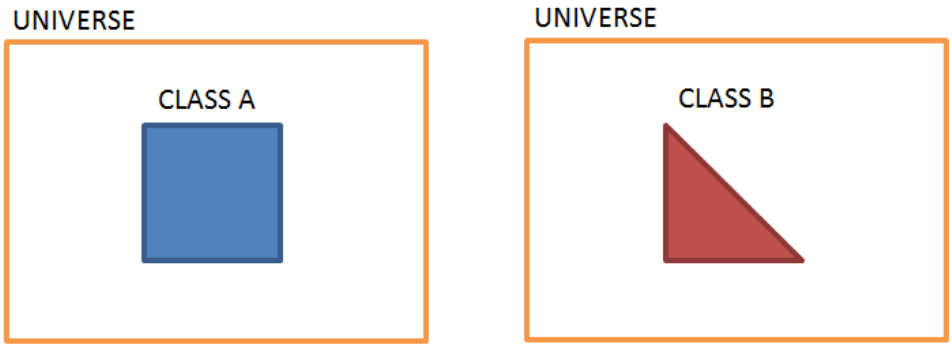


Figure 6.12 - Class 'A' and 'B' in the universe

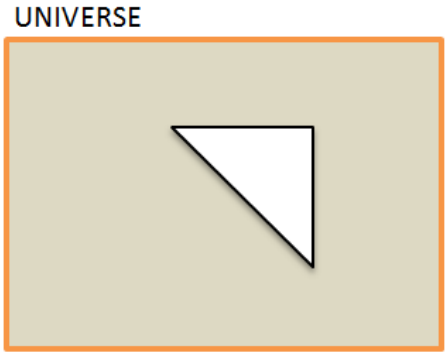


Figure 6.13 - Similarity between Class 'A' and 'B'

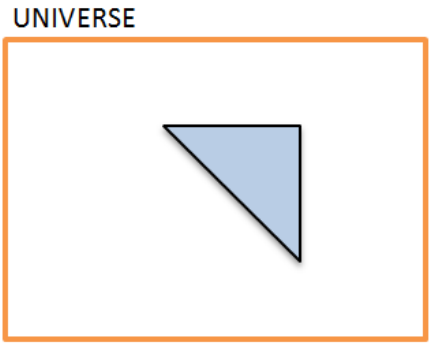


Figure 6.14 - Difference between Class 'A' and 'B'

If the n-tuple size ‘n’ is 1, the probability of a pattern from Class ‘B’ being classified as being part of Class ‘A’ is:

$$P_{\frac{B}{A}} = \frac{S}{N}\ x\ 100\% \qquad where: S \rightarrow \#\ of\ similar\ pixels$$

$$N \rightarrow total\ \#\ of\ pixels$$

If the n-tuple size ‘n’ is 2, the probability of a pattern from Class ‘B’ being classified as being part of Class ‘A’ is:

$$P_{\frac{B}{A}} = (\frac{S}{N})x(\frac{S-1}{N-1})$$

Since $P_{\frac{B}{A}}$ is required to be as small as possible, and ‘S’ and ‘N’ are very much larger than 1, it is possible to calculate the probability of a pattern from Class ‘B’ being classified as being part of Class ‘A’ when the n-tuple size is 2 by the following equation:

$$P_{\frac{B}{A}} = \left(\frac{S}{N}\right)^2$$

Using the theory presented above, in general the probability of a pattern from Class ‘B’ being classified as being part of Class ‘A’ is calculated as follows:

$$P_{\frac{B}{A}} = \left(\frac{S}{N}\right)^n$$

The effect of a change in n-tuple size has on the response of the system on a pattern therefore depends on how similar or different the pattern is with regards to the training data, as displayed in the graph below:

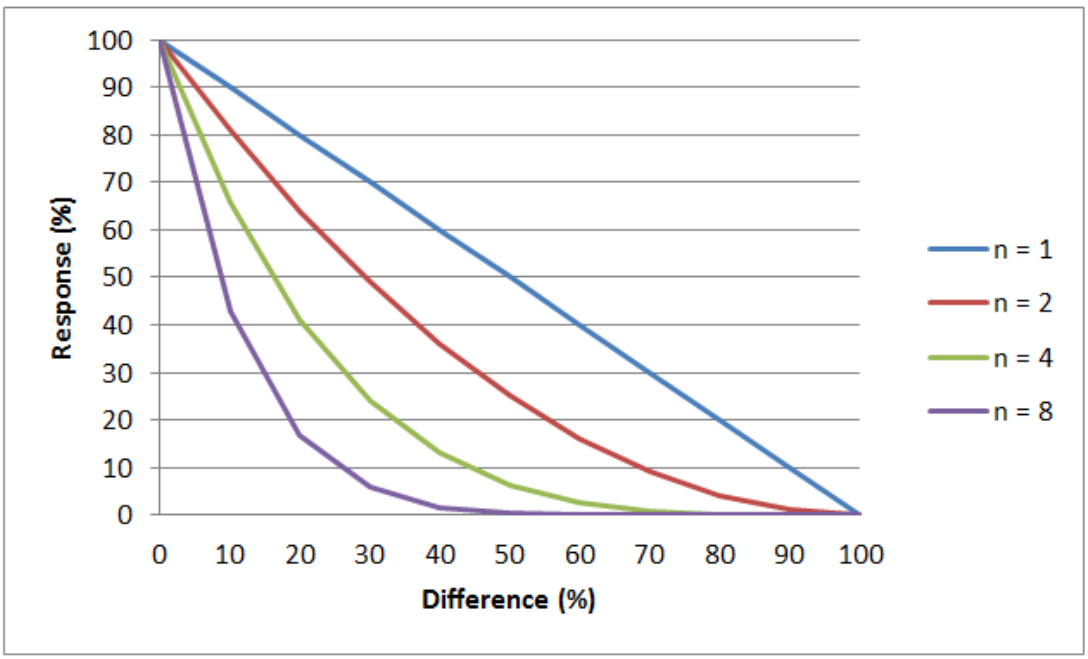


Figure 6.15 - Graph of response vs. difference for different n-tuple sizes

It is observed from the graph that as the n-tuple size increases, a small increase in the difference results in a large decrease in the response. This graph proposes that a lower

n-tuple size requires lesser number of patterns to generalise the net, and that since a higher n-tuple size discriminates more between patterns, more training data is required for the net to adequately generalise.

In order to investigate the impact a change in n-tuple size has on the net, the aforementioned MIT-CBCL database was used to train and test the system, with four different n-tuple sizes – 4, 5, 6 and 7-pixel n-tuples. As explained in Chapter 5, the MIT-CBCL Test dataset consists of 10 individuals, each with 200 unique images. Since the images vary between 115x115 pixels to 100x100 pixels, all the images were resized to 100x100 pixels using subsample normalization, in order to preserve the individual pixel values. The results obtained from each of the four tests are presented below, and the difference between each result is analysed.

6.2.1 Four-pixel n-tuple

Figure 6.16 displays the results of the MIT-CBCL database when a 4-pixel n-tuple system is tested on the database. The overall accuracy of the system is 99.9% and the correct responses vary mostly between 90% and 100%.

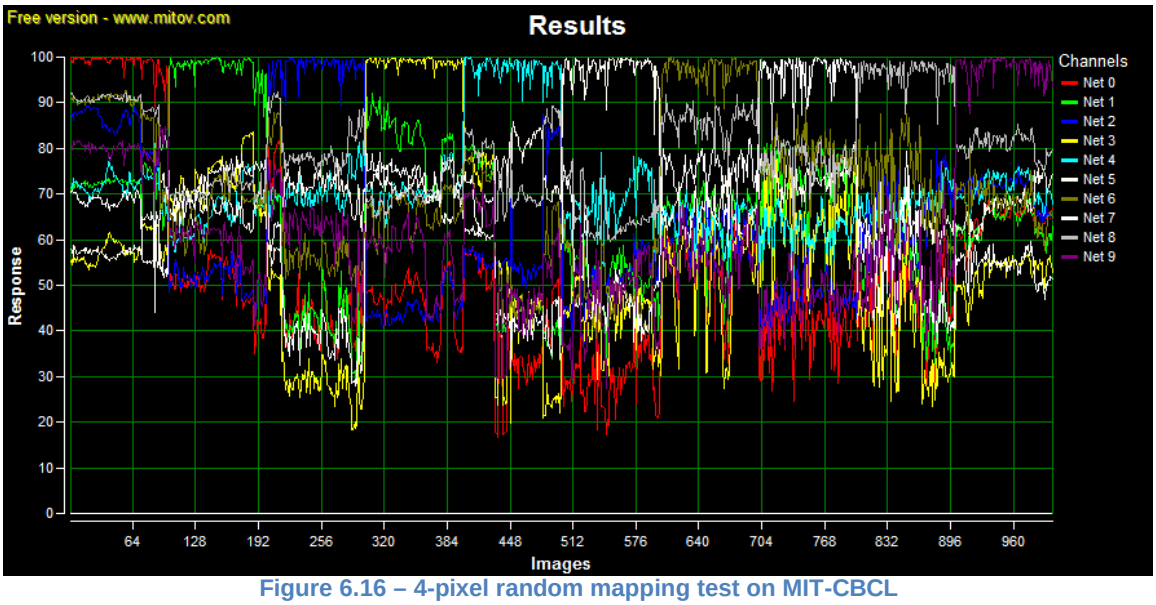


Figure 6.16 – 4-pixel random mapping test on MIT-CBCL

Although the majority of correct responses are in the 90% to 100% range, Table 6.3 shows the average response of each net when tested on another, and it is noticed that the majority of average false responses are between 50% and 90%.

		TESTED ON:									
		Net 0	**Net 1**	**Net 2**	**Net 3**	**Net 4**	**Net 5**	**Net 6**	**Net 7**	**Net 8**	**Net 9**
T	**Net 0**	98.82	71.94	83.74	57.73	72.10	58.30	87.95	66.41	89.72	79.81
E	**Net 1**	50.64	97.72	51.46	72.70	65.98	67.54	67.32	72.19	71.36	58.61
S	**Net 2**	47.70	46.30	97.33	31.88	70.83	69.39	60.18	41.98	80.48	62.43
T	**Net 3**	45.97	80.92	45.18	98.32	70.95	70.62	65.83	73.70	70.10	56.36
E	**Net 4**	40.81	54.24	59.82	50.48	97.52	72.21	57.50	50.61	76.12	53.59
D	**Net 5**	29.79	51.46	52.34	42.87	67.96	97.42	49.65	49.84	65.74	46.67
	Net 6	55.30	63.68	61.15	52.64	63.75	65.92	97.06	74.51	85.77	56.15
O	**Net 7**	40.81	72.10	47.37	62.68	62.15	73.87	77.68	96.65	77.59	47.53
N	**Net 8**	46.39	50.02	65.63	38.94	67.72	62.92	71.71	54.02	96.80	53.87
	Net 9	63.94	63.40	71.08	52.59	71.42	67.43	68.20	54.28	80.96	97.38

Table 6.3 - Average response 4-pixel random mapping on MIT-CBCL

When the average results are mapped out on a graph, displayed in Figure 6.17, it is clear that whilst the overall average correct response is above 95%, the nets generalise too much, resulting in a high false response value.

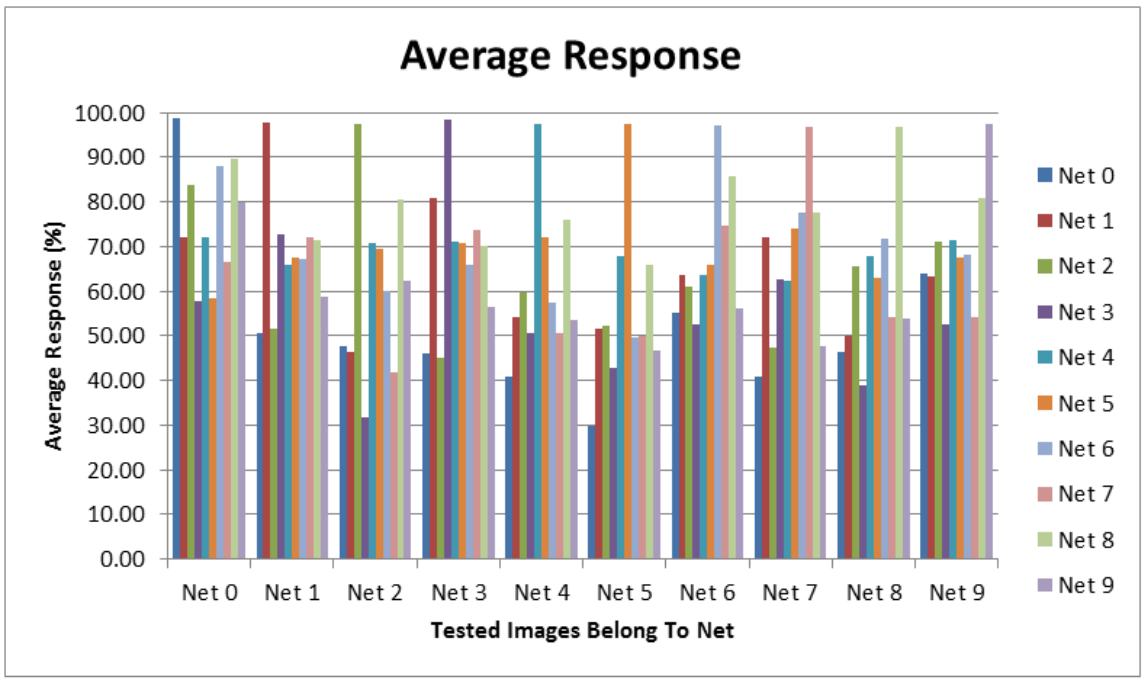


Figure 6.17 - Average response 4-pixel random mapping on MIT-CBCL

This is also displayed on the confidence graph displayed below in Figure 6.18, where it shown that the confidence of the system varies mostly between 1% and 20%.

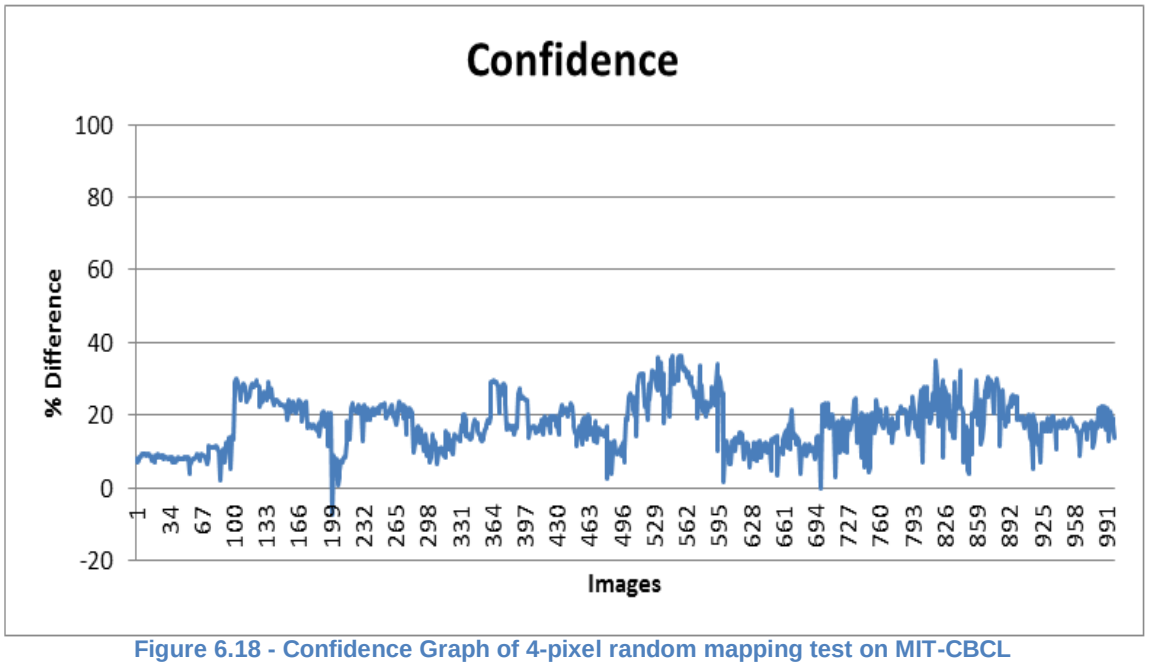


Figure 6.18 - Confidence Graph of 4-pixel random mapping test on MIT-CBCL

6.2.2 Five-pixel n-tuple

When the n-tuple size is increased to 5-pixel n-tuples, Figure 6.19 displays the test results achieved, and once again the system achieves an accuracy of 99.9%. The results graph shows a decrease in the correct response percentage compared with the 4-pixel n-tuple system, and this is proved to be the case in the average response table shown in Table 6.4, which shows that the average correct response is now between 85% and 97%.

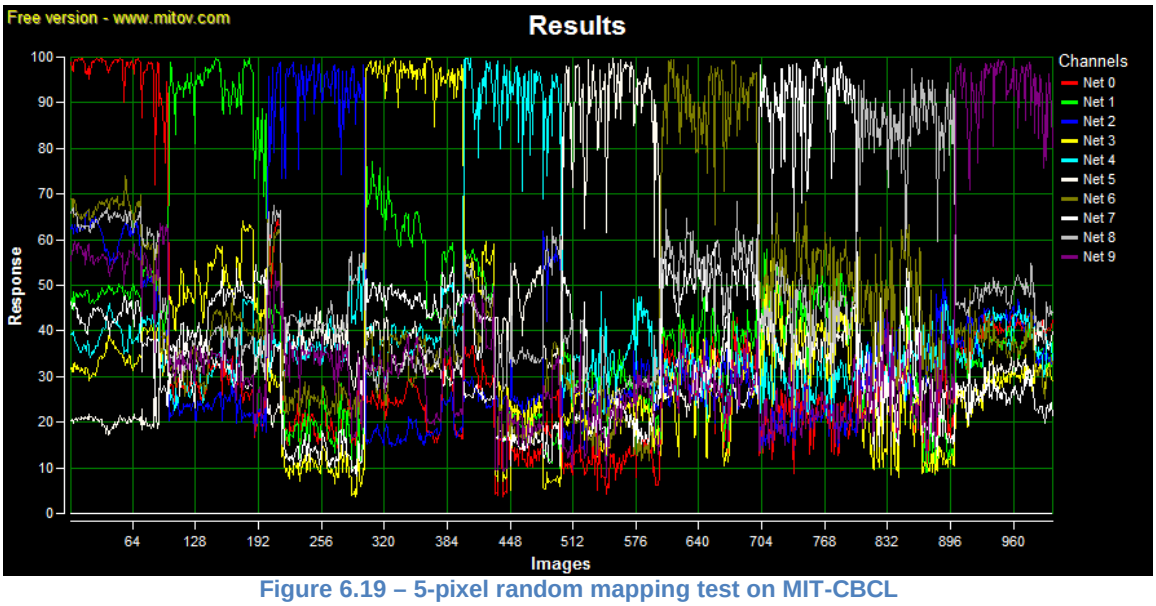


Figure 6.19 – 5-pixel random mapping test on MIT-CBCL

		TESTED ON:									
		Net 0	**Net 1**	**Net 2**	**Net 3**	**Net 4**	**Net 5**	**Net 6**	**Net 7**	**Net 8**	**Net 9**
T	**Net 0**	96.82	47.97	57.26	34.46	39.09	21.61	62.77	40.88	61.99	55.95
E	**Net 1**	28.38	92.34	23.39	50.92	32.82	32.54	37.49	44.10	35.93	31.75
S	**Net 2**	25.62	22.38	91.74	12.96	38.09	34.67	30.01	16.93	46.78	34.50
T	**Net 3**	24.03	59.06	18.89	95.67	39.43	34.97	35.37	47.13	33.98	30.33
E	**Net 4**	19.90	29.56	30.92	29.24	91.54	37.45	29.32	24.59	42.56	27.95
D	**Net 5**	11.67	26.53	25.54	21.27	34.25	89.86	21.28	24.33	30.35	21.39
	Net 6	31.87	36.24	28.92	27.75	30.11	28.66	89.67	50.33	54.82	28.87
O	**Net 7**	21.42	44.90	20.45	38.34	28.91	37.72	52.91	89.86	44.89	22.28
N	**Net 8**	24.43	23.42	34.54	17.45	34.48	26.12	42.05	28.08	85.00	26.97
	Net 9	39.79	35.86	40.51	28.38	38.60	31.19	36.71	25.86	47.53	92.40

Table 6.4 - Average response 5-pixel random mapping on MIT-CBCL

Although the average correct response decreases by around 6%, when the average responses are plotted in Figure 6.20, the average false responses decrease drastically when compared with the 4-pixel n-tuple test, with the majority being between 20% and 60%.

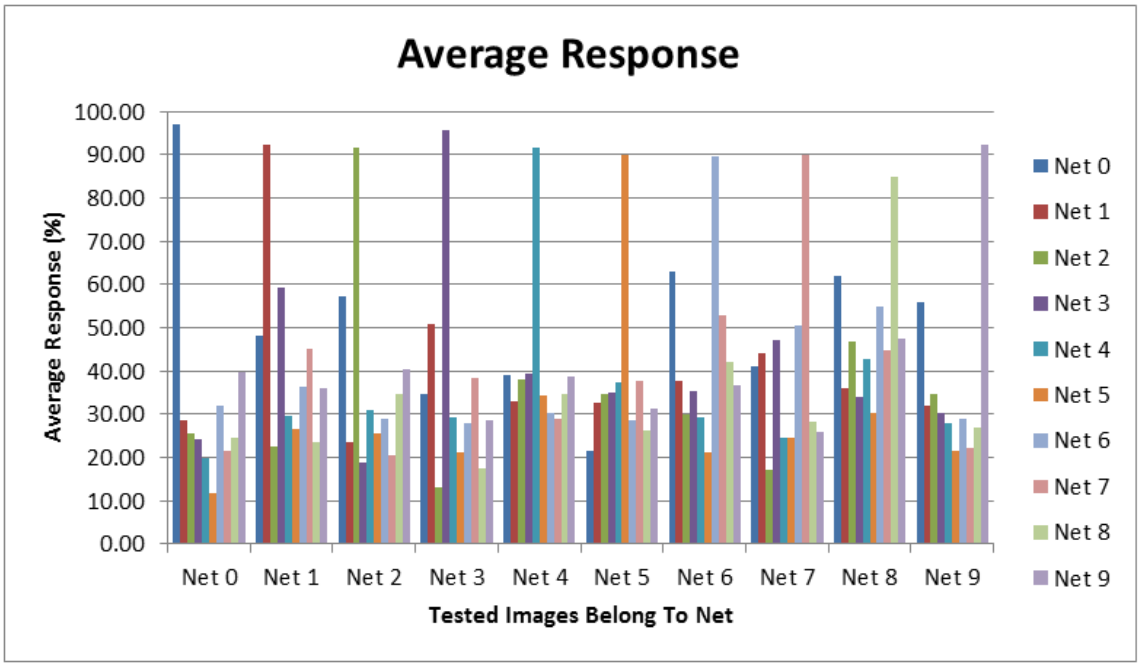


Figure 6.20 - Average response 5-pixel random mapping on MIT-CBCL

This is also noticed in the confidence graph of the 5-pixel n-tuple test (Figure 6.21) where the majority of confidence values lie between 20% and 60%, compared to the 1%-20% achieved by the 4-pixel n-tuple system.

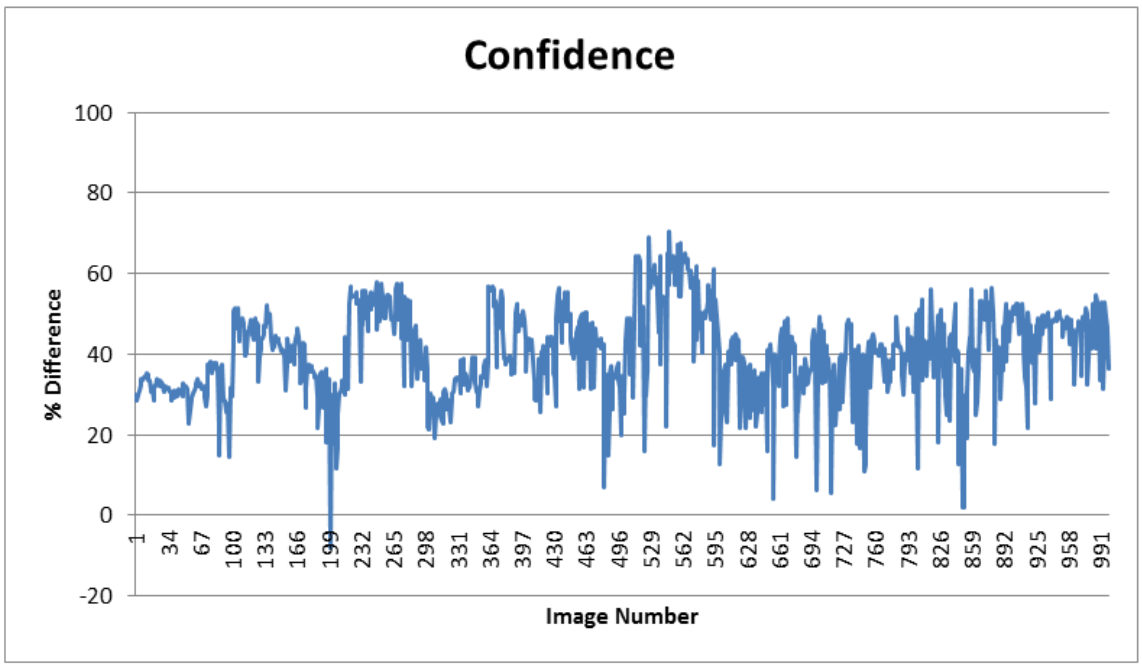


Figure 6.21 - Confidence Graph of 5-pixel random mapping test on MIT-CBCL

6.2.3 Six-pixel n-tuple

When the MIT-CBCL database is used to test the 6-pixel n-tuple system, the overall accuracy of the system now falls to 99.7%, and the results obtained by the system are displayed in Figure 6.22. There is a noticeable decrease in the value of the correct response of each image, and this is proved in the average response table (Table 6.5), where the values are shown to be between 63% and 93%.

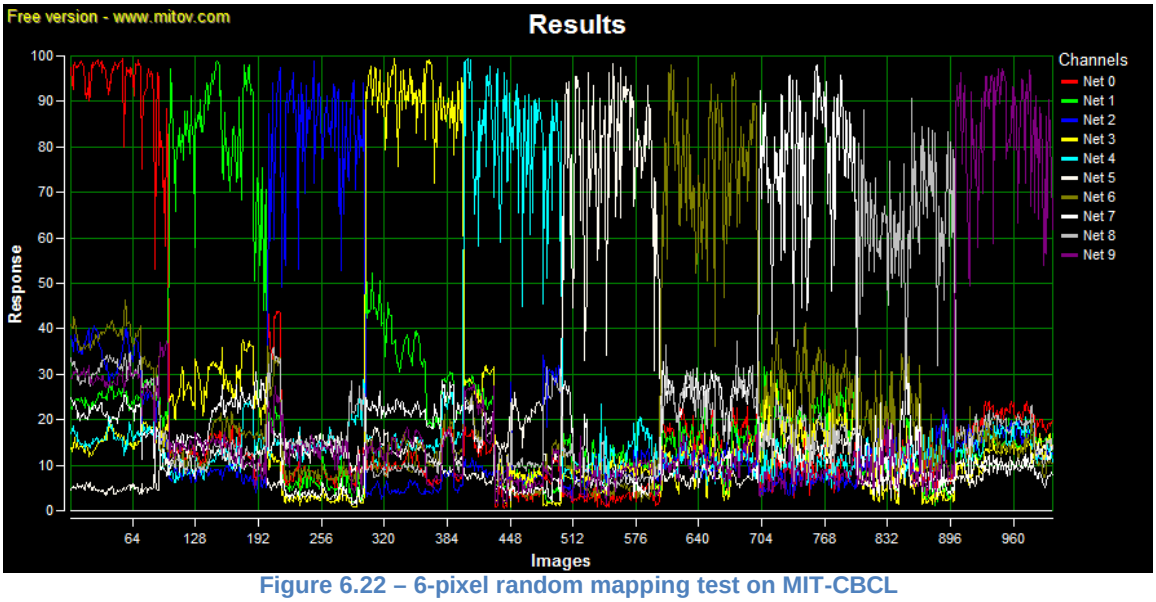


Figure 6.22 – 6-pixel random mapping test on MIT-CBCL

		TESTED ON:									
		Net 0	**Net 1**	**Net 2**	**Net 3**	**Net 4**	**Net 5**	**Net 6**	**Net 7**	**Net 8**	**Net 9**
T	**Net 0**	93.12	24.56	31.43	15.62	15.85	5.47	35.30	19.79	29.34	29.05
E	**Net 1**	13.73	81.81	8.00	28.10	12.12	11.96	15.19	20.29	11.51	13.22
S	**Net 2**	12.22	8.43	82.14	4.03	15.31	11.99	11.56	5.56	18.54	13.88
T	**Net 3**	10.21	32.96	5.79	90.11	15.77	12.90	13.15	22.32	10.56	12.66
E	**Net 4**	7.95	12.74	12.24	13.70	81.59	13.76	10.89	9.62	16.05	11.89
D	**Net 5**	3.22	10.12	8.46	8.06	12.45	76.87	6.22	9.19	8.50	7.28
	Net 6	15.80	14.76	10.66	10.62	10.13	8.51	75.31	25.60	24.59	10.80
O	**Net 7**	9.60	20.74	6.53	17.82	9.67	14.01	27.51	77.35	17.18	7.85
N	**Net 8**	10.18	7.91	13.03	5.70	12.33	6.94	18.05	11.38	63.83	9.64
	Net 9	19.97	14.61	15.98	12.27	14.94	9.73	14.23	9.47	18.21	82.81

Table 6.5 - Average response 6-pixel random mapping on MIT-CBCL

When the average responses are plotted in Figure 6.23, the slight decrease in the average correct response value is offset by the great decrease in the average false response values, which are now mainly between 0% and 30%.

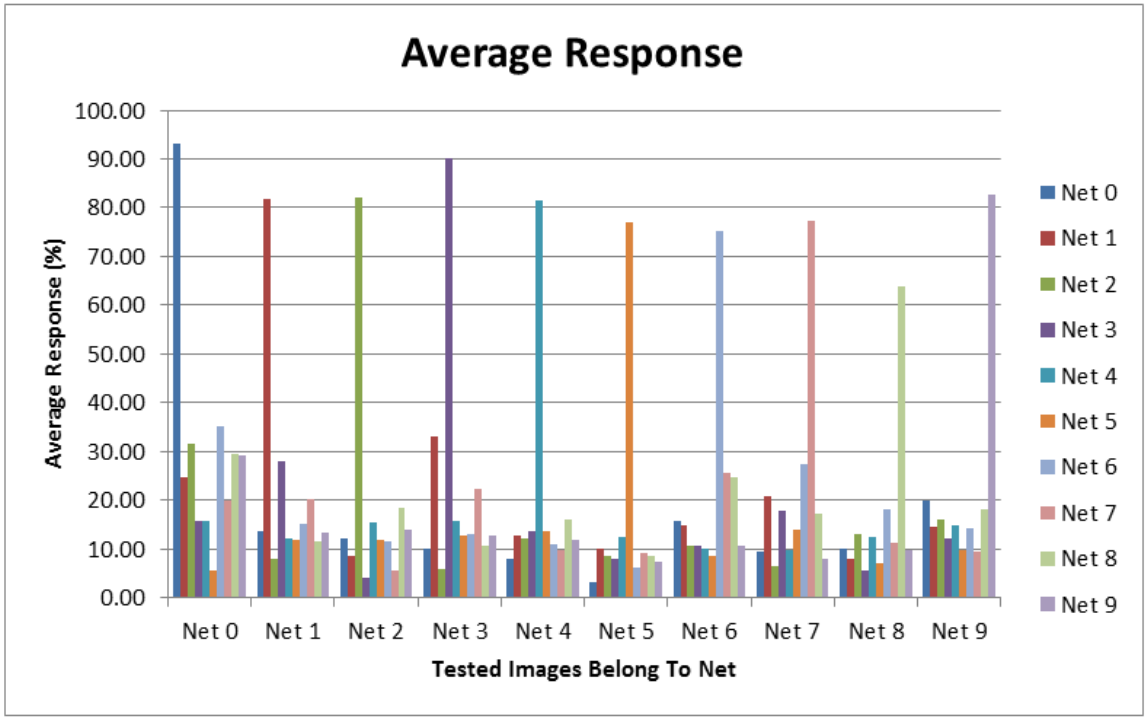


Figure 6.23 - Average response 6-pixel random mapping on MIT-CBCL

This is also shown in the confidence graph displayed in Figure 6.24, where the majority of confidence values lie between 40% and 80%, compared to the 20%-60% achieved by the 5-pixel n-tuple system.

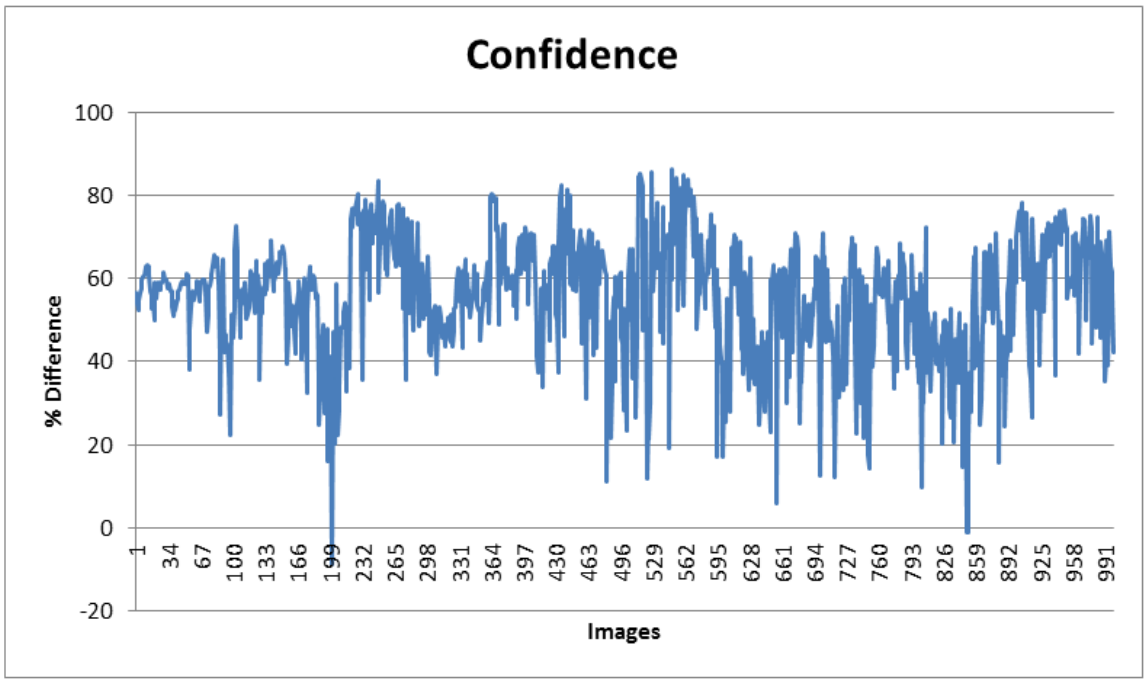


Figure 6.24 - Confidence Graph of 6-pixel random mapping test on MIT-CBCL

6.2.4 Seven-pixel n-tuple

When the MIT-CBCL data is used to test the 7-pixel n-tuple system, even though the results (displayed in Figure 6.25) have an overall accuracy of 99.7%, this graph show the greatest decrease in the value of the correct response when compared to the other n-tuple systems.

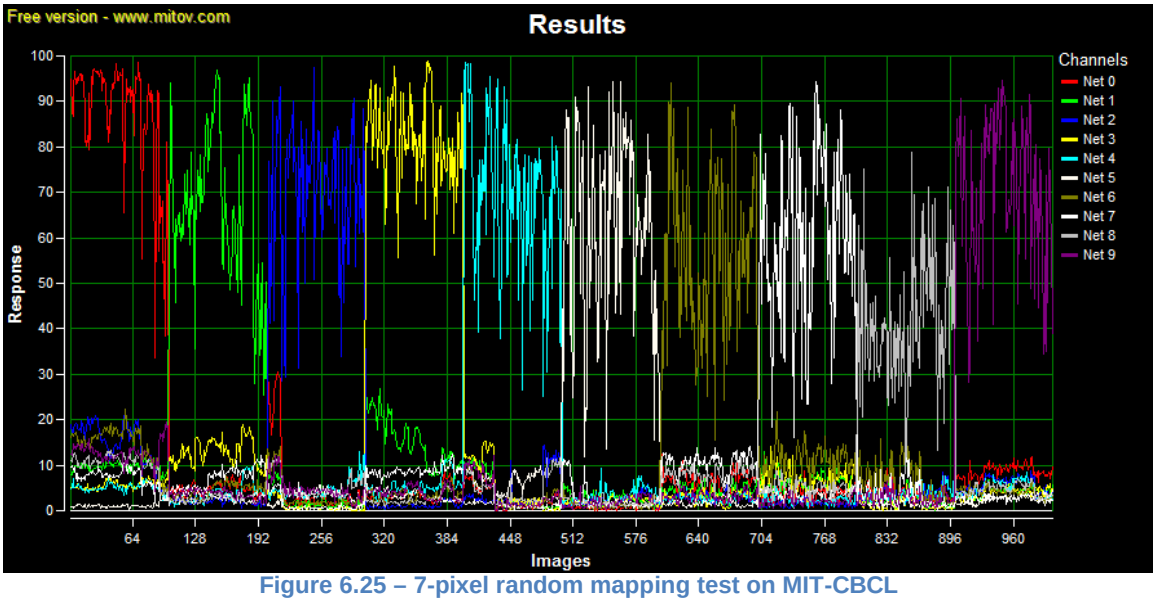


Figure 6.25 – 7-pixel random mapping test on MIT-CBCL

The average correct response varies between 41% and 87%, as shown in Table 6.6, however the average false response falls drastically to between 0% and 10%, as seen in the plot of Table 13 shown in Figure 6.26.

		TESTED ON:									
		Net 0	**Net 1**	**Net 2**	**Net 3**	**Net 4**	**Net 5**	**Net 6**	**Net 7**	**Net 8**	**Net 9**
T	**Net 0**	87.04	9.69	14.77	5.66	5.03	1.22	14.70	7.27	10.35	12.73
E	**Net 1**	5.47	67.08	2.06	12.80	3.77	2.96	4.36	7.12	2.65	3.78
S	**Net 2**	5.37	2.63	69.35	1.10	4.58	3.28	3.21	1.36	5.10	4.69
T	**Net 3**	3.37	15.14	1.84	81.41	5.47	3.37	3.62	8.68	2.47	3.98
E	**Net 4**	2.73	4.05	3.82	5.78	69.00	4.10	3.40	3.02	4.83	3.87
D	**Net 5**	0.83	2.62	2.45	2.18	3.99	61.69	1.47	2.78	1.88	1.96
	Net 6	6.36	4.24	2.90	2.88	2.41	1.97	56.59	10.37	7.70	3.03
O	**Net 7**	3.47	6.80	1.50	6.63	2.38	3.85	10.88	60.41	5.17	2.13
N	**Net 8**	3.60	2.06	3.60	1.54	3.58	1.59	6.21	3.83	41.19	2.76
	Net 9	8.56	4.18	5.67	3.85	4.87	2.57	4.03	2.68	5.17	70.22

Table 6.6 - Average response 7-pixel random mapping on MIT-CBCL

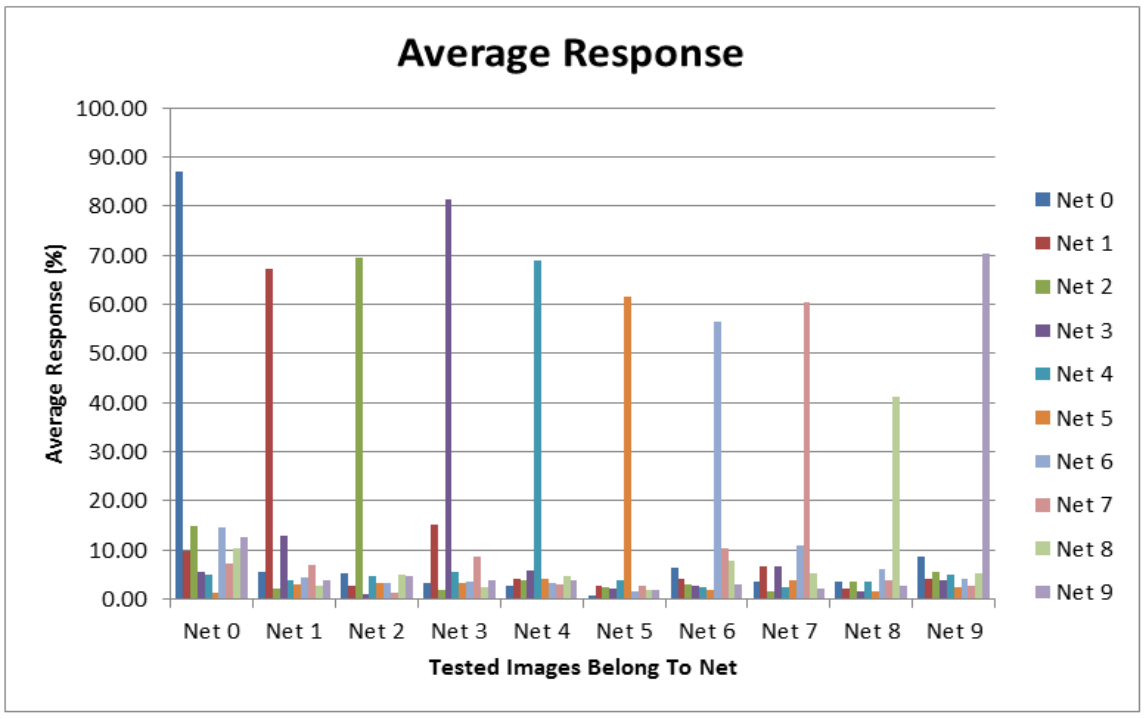


Figure 6.26 - Average response 7-pixel random mapping on MIT-CBCL

Together with the above graph, the confidence graph in Figure 6.27 also displays an increase in the average confidence, and although the values vary between 20% and 80%, the majority of the confidence values are higher than that achieved in any other n-tuple system.

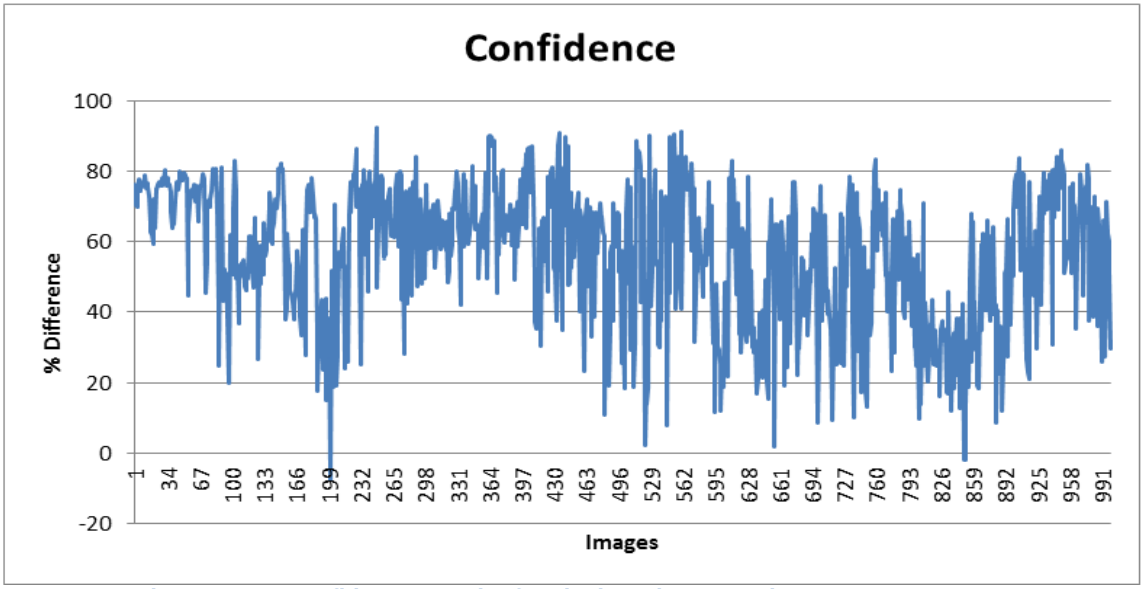


Figure 6.27 - Confidence Graph of 7-pixel random mapping test on MIT-CBCL

6.2.5 Overall comparison of results

Table 6.7 displays the average, maximum and minimum correct responses achieved in the four tests performed above. The difference between the minimum correct response values achieved in the four tests, and the average response values, proves that a higher n-tuple size system discriminates more than a lower n-tuple size system.

Correct Response				
	n-tuple size			
	4	5	6	7
max	99.96	99.8	99.4	98.95
average	97.84	92.0	81.16	66.98
min	81.96	56.35	30.91	11.83

Table 6.7 - Comparison of correct responses of different n-tuple sizes in MIT-CBCL test

Table 6.8 compares the confidence values achieved in the four tests, the rising maximum and average confidence values also proves the discrimination properties discussed above.

Confidence				
	n-tuple size			
	4	5	6	7
# of Errors	1	1	3	3
max	36.4	70.35	86.32	92.3
average	17.66	40.28	55.45	56.25
min	-7.56	-8.15	-8.7	-7.56

Table 6.8 - Comparison of confidence of different n-tuple sizes in MIT-CBCL test

It is important to note here, that when the n-tuple size increases, the amount of memory the system uses increases significantly, as is shown in Table 6.9. From only 61kB required per net when 4-pixel n-tuples are used, to 6.9MB in a 7-pixel n-tuple system. This is the amount of storage space required per net, so if in a real-world scenario 1000 nets are stored by the system, the overall memory space required would be approximately 7GB, which is neither huge, nor costly, with the advanced state of technology today.

n-tuple size	bits required per function	storage required per net
4	25	61kB
5	121	236kB
6	721	1.2MB
7	5041	6.9MB

Table 6.9 - Comparison of storage required per n-tuple size for MIT-CBCL

Regarding the results obtained in the four tests above, it is interesting to note however, that when the 4 and 7-pixel n-tuple systems are compared, displayed in Table 6.10, even though the maximum correct response remains above 98%, the average drops by more than 30%, and the minimum by 70%.

Correct Response		
	n-tuple size	
	4	7
max	99.96	98.95
average	97.84	66.98
min	81.96	11.83

Confidence		
	n-tuple size	
	4	7
# of Errors	1	3
max	36.4	92.3
average	17.66	56.25
min	-7.56	-7.56

Table 6.10 - Comparison between 4 and 7 n-tuple sizes on MIT-CBCL test

The same is found in the confidence table above, where even though the maximum confidence increases from 36% to 92%, the average only rises to 56%, and the minimum stays the same at -7%.

To investigate why the average value is much lower than the maximum value in the confidence and the correct response categories, the type of images being used in each subject has to be investigated. To do so, one image was selected per class, for each of the ten classes, and each image was tested against the net it belongs to. This test was

performed with the same image on each of the four different n-tuple size systems, and the results obtained are displayed in Table 6.11.

n-tuple size	Image From									
	Net 0	Net 1	Net 2	Net 3	Net 4	Net 5	Net 6	Net 7	Net 8	Net 9
4	99.84	99.84	97.12	99.44	98.68	99.6	92.6	98.76	98.96	99.8
5	98.85	99.45	87.5	98.25	94.5	98.55	74.7	79.35	92.8	99.15
6	97.12	98.62	71.91	94.54	86.25	96.4	50.48	58.4	76.29	97.24
7	94.61	94.54	50.91	89.29	74.72	92.02	25.14	34.03	52.52	94.68

Table 6.11 - Comparison of response on different n-tuple sizes

The above results show a great difference between nets, where some images respond above 90% regardless of the n-tuple size, and some have a reduced accuracy as the n-tuple size is increased.

Table 6.12 divides these nets better, in order to differentiate why this happens.

n-tuple size	Image From									
	Net 9	Net 0	Net 1	Net 5	Net 3	Net 4	Net 8	Net 2	Net 7	Net 6
4	99.8	99.8	99.8	99.6	99.4	98.7	99	97.1	98.8	92.6
5	99.2	98.9	99.5	98.6	98.3	94.5	92.8	87.5	79.4	74.7
6	97.2	97.1	98.6	96.4	94.5	86.3	76.3	71.9	58.4	50.5
7	94.7	94.6	94.5	92	89.3	74.7	52.5	50.9	34.0	25.1

Table 6.12 - Grouping of comparison of response on different n-tuple sizes

Table 6.12 sorts these nets into four colour-coded groups, where 4 nets have responses above 90% on their images regardless of the n-tuple size, 2 nets have responses above 70%, two remain above 50%, and two reduce to below 35%. Due to the drastic fall in accuracy in Net 6 and Net 7, both the overall average confidence and average correct response of the 6 and 7-pixel system tests is reduced. To understand why such a percentage difference could occur between nets, some images from Net 0 and Net 7 are displayed in Figure 6.28 and Figure 6.29.

Figure 6.28 - Sample images from Subject 0

Figure 6.29 - Sample images from Subject 7

It is clearly visible that Subject 0 has a constant background, camera angle, and fairly standard lighting throughout the database, and so has a high response throughout, however Subject 7 has images that vary greatly in all three aspects, and so the reduction in average accuracy and confidence can be attributed to this fact.

It is important to note however, that the results in Table 6.8 comparing the number of errors and the confidence between 4-7 n-tuple systems are based on the assumption that no threshold is applied on the overall response, to determine whether or not the image being tested on is part of a net or not. This means that each image is attributed to the net on which it achieves the maximum response, whether the response is above 90%, or below 20%.

It should also be noted, that although the effects of applying a threshold on the results of the system are being investigated, there is no single threshold value that can be used, and that every database and n-tuple size has its own threshold, and only by adjusting this value can system optimization be achieved. Table 6.13 displays the effects different thresholds have on the results of the MIT-CBCL tests.

n-tuple size	Threshold	FAR	FRR	FAR (%)	FRR (%)
4	90	0	21	0	2.1
4	80	1	0	0.1	0
5	65	0	9	0	0.9
5	56	1	0	0.1	0
6	50	0	32	0	3.2
6	37	0	10	0	1
6	36	3	0	0.3	0
7	50	0	229	0	22.9
7	19	0	11	0	1.1
7	10	3	0	0.3	0

Table 6.13 - Comparison of threshold required for different n-tuple sizes on MIT-CBCL

As the n-tuple size increases, the response of the images on their own class drastically reduces, from a 90% threshold on a 4-pixel n-tuple system resulting in a FRR of 21, to a threshold of 50% on a 7 n-tuple system resulting in the FRR being 229, which is an increase of more than 20%!

6.3 Training set size

As stated in Chapter 6.2, it is theorised that at a lower n-tuple size, fewer images are required to train the system as each image causes the net to generalise greatly, and at a higher n-tuple size more images are required to adequately generalise the net. However, just as the optimum n-tuple size and the response threshold depends on the data being used, so does the number of images that are required for training.

To investigate this, the MIT-CBCL database was used, and tests were run on four different n-tuple size systems – 4, 5, 6 and 7 n-tuple systems. Three tests were run on each of the system, with the difference between each test being the size of the training and testing dataset, the sizes being:

1. 50 images for training and 150 for testing
2. 100 images for training and 100 for testing
3. 150 images for training and 50 for testing

It should be noted that in all the tests conducted, each of the images in the testing set were not used during the training process. The overall confidence of each system is displayed in Table 6.14 below:

Train 50, Test 150		
n-tuple size	**Number Of Errors**	**Average Confidence**
4	76	21.66
5	75	37.94
6	72	43.34
7	65	38.75
Train Even, Test Odd		
n-tuple size	**Number Of Errors**	**Average Confidence**
4	1	17.7
5	1	40.2
6	3	55.5
7	3	56.3
Train 150, Test 50		
n-tuple size	**Number Of Errors**	**Average Confidence**
4	3	14.99
5	1	37.86
6	1	54.76
7	1	57.03

Table 6.14 - Comparison of training set size on different n-tuple sizes on MIT-CBCL

When the n-tuple size is 4 and the training set consists of 50 images, 76 images are recognized incorrectly, when the training set and the testing set consist of 100 images, 1 image is recognized incorrectly, and when the training set consists of 150 images, 3 images are recognised incorrectly.

Although the decrease in the number of errors between the 50 image training set and the 100 image training set can be recognized as the system generalizing, the increase in the number of errors between the 100 image training set and the 150 image training set is due to the system over-generalizing, which is known as saturation. This means that although 50 images is too little to train a 4-pixel n-tuple system at 100x100 pixel per image, 150 images is too many, and so the number of images required is around 100 with regards to this database.

When the average confidence of the 4-pixel n-tuple systems is observed, it is noticed that there is a significant decrease in confidence between the 50 image training set and the 100 image training set, which can mean one of two things:

1. The images across the different training nets are very similar
2. 100 images are too many to train on for a 4 pixel, 100x100 pixels per image, n-tuple system, as the net is already approaching saturation.

When the 5-pixel n-tuple system is observed, 50 images is too small a training set, and 100 images seem to be the correct amount, as even though the number of errors does not increase when the training set is increased to 150 images, the average confidence decreases, which is due to the system over-generalizing.

The 6-pixel n-tuple system when trained on 50 or 100 100x100 pixel images under-generalizes, however even though the number of errors decreases to 1 when the training set consists of 150 images, the average confidence decreases by approximately 1%, which suggests that the training set should consist of around 120-140 images.

For the 7-pixel n-tuple system however, both 50 and 100 image training sets are too small, and clearly at least 150 images are required to train the system, however the exact number of images required to train a 100x100 pixel image system can only be known by increasing the training set further and comparing the results.

6.4 Conclusion

In this chapter system optimization was investigated, where the different aspects of the system were changed in order to observe the effect it had on the system. The mapping used in the system was investigated, arriving to the conclusion that random mapping is the only suitable mapping that can be used for pattern recognition.

The n-tuple size and training net size were investigated, where it was concluded that the two aspects were dependant on one another, and an increase or decrease in the overall accuracy cannot be solely attributed to one factor.

The conclusion arrived to, after all these tests, is that there exists no single global optimum solution with regards to this system, and each implementation can be optimized differently, by observing the storage available, the amount of training data that exists, the initial image size, and the total generalization and discrimination required.

Chapter 7 :

Image Manipulation

7. IMAGE MANIPULATION

There are various aspects of the methodology presented in Chapter 3 of this thesis that can be used to optimise the system, and whilst the internal variables have been discussed in the previous chapter, it is necessary to investigate the external variables; which are:

1. Image size.
2. Image lighting.
3. Image viewing distance.

7.1 Image size

It is important to stress that the amount of images required to train a system does not only depend on the n-tuple size, but also on the size of the images being processed by the system. As the image size increases, the number of n-tuples in the system increases, and therefore the number of functions being used to store each net increases. This in turn requires more images for training in order to generalize the system.

This however requires investigation - if the original image size is at for example 480x320 pixels, this requires 3.5MB of memory to store each net. What effect is there on the overall response of the system, if the images are resized to 240x160 pixels, which is half their original size, where each net then requires only 0.88MB?

7.1.1 Resizing methods

Before the effects of resizing are investigated, it is important to investigate the effects different resizing methods have on the algorithm, and whether any specific resizing algorithm results in a better accuracy when the image is tested by the system.

There are four main methods of resizing an image:

1. Interpolation – these methods involve taking specific pixel locations in the image being processed, and then attempt to determine a logical colour value, based on the colour values of the pixels surrounding it.
2. Gaussian Filter – these are transforms that are meant to remove any high frequency noise that may be present in the image being processed.
3. Windowed Filter – windows are applied to the impulse response of an ideal low pass filter of the image.
4. Cubic Filter – applies a pre-filter to the image to assert high fidelity scaling.

7.1.2 Resizing test

The resizing methods being tested are:

1. Nearest Neighbour – Point Interpolation
2. Bilinear Interpolation – Triangle Interpolation
3. Gaussian Filter

4. Lanczos Filter – Windowed Filter
5. Mitchell-Netravali Filter – Cubic Filter

In order to test each of these resizing methods with standardized conditions, Subject 0 from the MIT-CBCL Test Database was used to train and test the 5-pixel n-tuple system. As mention earlier in Chapter 5, the MIT-CBCL Test Database consists of 10 individuals, each with 200 unique images at 100x100 to 115x115 pixel resolution. Therefore, the 200 images of Subject 0 were divided into two, with 100 images being used to train each system, and 100 unique images being used to test it.

In order to test the resizing methods, two tests were run for each method, one with a lower resolution than the original image, and the other with a higher resolution. As it is generally easier to resize to half or double the original size of the image, sizes unrelated to the original images were chosen, and therefore, for each of these tests, the 200 original images were resized to 69x69 pixels and 169x169 pixels.

In accordance with the rule that standardized conditions must be used in each test, the five tests at 69x69 pixel resolution were all run with the same random mapping, and so were the tests at 169x169 pixels.

The average, maximum and minimum response in each of the test is shown below in Table 7.1.

69x69

	Bilinear	Gaussian	Lanczos	Mitchell	Nearest N	Difference
Max	99.5	99.87	99.5	99.75	99.5	0.37
Average	93.14	94.32	93.21	93.8	93.14	1.18
Min	54.73	53.47	53.09	52.84	54.73	1.89

169x169

	Bilinear	Gaussian	Lanczos	Mitchell	Nearest N	Difference
Max	99.35	99.43	99.26	99.41	99.35	0.17
Average	93.06	93.61	92.89	93.36	93.06	0.72
Min	53.11	52	52.54	52.52	53.11	1.11

Table 7.1 - Comparison of resizing filters

As it can be seen in the table above, in both tests the average results are very similar, with the maximum difference between the lowest average and the highest average being between 0% and 2% in each category, and although there are differences in the results achieved, no single resizing method stands out between both of the resolutions.

Due to the results achieved in the resizing tests, a decision was made that the methodology with the fastest processing time, and which required the least amount of computation intensity would be used to resize images within the system, which was the point interpolation method, known as the nearest neighbour.

7.1.3 Testing resized images from 480x320 pixels to 15x10 pixels

Although tests have been run to investigate the difference between various resizing methods, another aspect that requires investigation is whether or not resizing an image has an effect on the overall response of the image.

This is meant to address the storage requirements of the system, for example if tests were run on a 5-pixel n-tuple system, with images at a resolution of 1280x720 pixels, each net would require 21MB of storage. However, if the images were resized to half their original size, at 640x360 pixel resolution, each net would only require 5.3MB, which is around a quarter of the original amount of memory required.

The MIT-CBCL Test Database could not be used to determine this, as the original images were already at a small resolution, being between 115x115 to 100x100 pixels, and therefore the images obtained from the Jenkem video file (described in Chapter 4) were used. The original images were at a resolution of 480x320 pixels, and so 6 tests were run, one at the original resolution, and then at $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \text{and } \frac{1}{32}$ of the original size. In order to understand just how the detail in the image changes as its resolution is lowered; the same image from each of these sizes is displayed below at a standardized size, from Figure 7.1 to Figure 7.6:

Figure 7.1 - Sample image from Jenkem Video at 640x480 pixel resolution

Figure 7.2 - Sample image from Jenkem Video at 240x160 pixel resolution

Figure 7.3 - Sample image from Jenkem Video at 120x80 pixel resolution

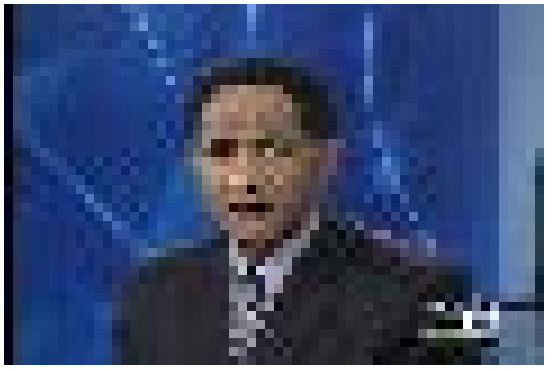


Figure 7.4 - Sample image from Jenkem Video at 60x40 pixel resolution

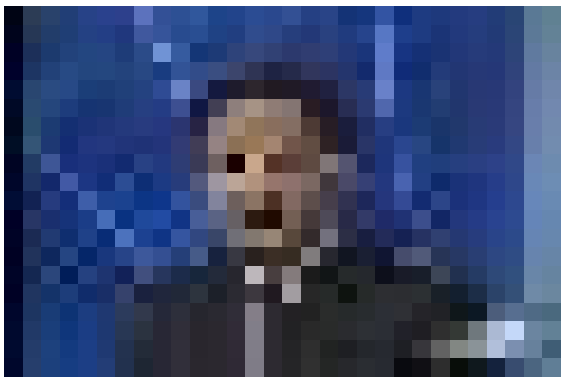


Figure 7.5 - Sample image from Jenkem Video at 30x20 pixel resolution

Figure 7.6 - Sample image from Jenkem Video at 15x10 pixel resolution

It should be noted that although the smallest image size is 15x10 pixels, which is tiny compared to the megapixels we use today, if the image was in binary, the pattern space universe for a binary 15x10 pixel image would be 2^{150}, which is approximately 10^{45} possible patterns!

The resizing was done using the nearest neighbour method, and four tests were run at each resolution, one for each n-tuple size, and although the FAR and FRR were both 0%, resulting in 100% accuracy in each of the tests, a change in the average confidence did occur, as shown below in Table 7.2:

4-pixel n-tuple	
image size	**average confidence**
480x320	43.66
240x160	45.74
120x80	44.94
60x40	45.12
30x20	43.45
15x10	36.31

5-pixel n-tuple	
image size	**average confidence**
480x320	54.7
240x160	57.36
120x80	56.78
60x40	55.65
30x20	55.52
15x10	50.52

6-pixel n-tuple	
image size	**average confidence**
480x320	57
240x160	59.82
120x80	59.84
60x40	59.24
30x20	58.44
15x10	58.98

7-pixel n-tuple	
image size	**average confidence**
480x320	54.76
240x160	54.79
120x80	54.76
60x40	54.77
30x20	54.68
15x10	57.11

Table 7.2 - Image size comparison

Although the results recorded in the table have seemingly no pattern, they point towards an interesting theory – at lower n-tuple sizes, a larger image size results in a higher average confidence, whereas at higher n-tuple sizes, lower image sizes results in a higher average confidence. Although this seems to be the case, the actual difference resizing an image has on the system is minimal, with differences ranging from 0.01% to 2.82%.

The only exception in this minute difference is in the 4-pixel n-tuple system, when the image is halved from 30x20 pixels to 15x10 pixels, and the average confidence decreases by 7.14%. This is due to the small amount of ranks (37 total) in total a 15x10, 4-pixel n-tuple system uses. Figure 7.7 below displays just how minimal the average change is:

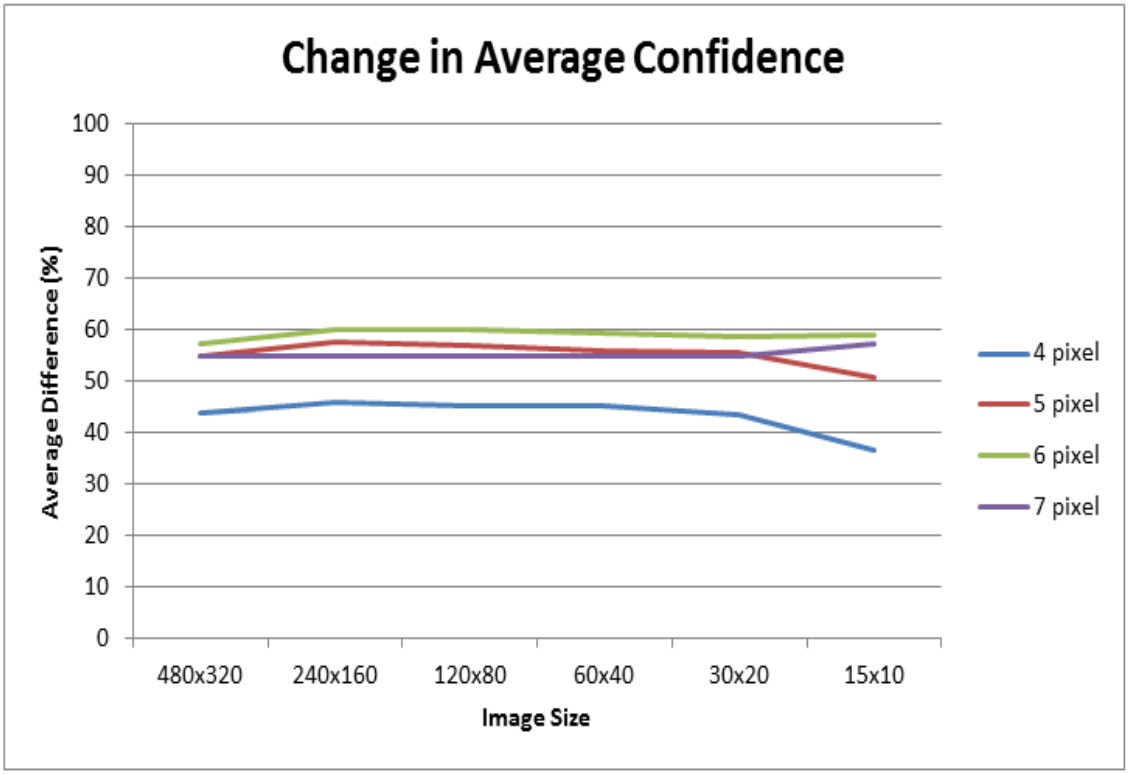


Figure 7.7 - Effect a change in image size has on the average confidence

Resizing an image to half its original size makes a large difference on the amount of storage that is required to train a net, and this is further affected by increasing or decreasing the n-tuple size, as seen in Table 7.7 below:

Image Size	n-tuple size			
	4	5	6	7
480x320	900KB (0.88MB)	3600KB (3.52MB)	18000KB (17.58MB)	108000KB (105.47MB)
240x160	225KB	900KB (0.88MB)	4500KB (4.39MB)	27000KB (26.37MB)
120x80	56.25KB	225KB	1125KB (1.1MB)	6750KB (6.59MB)
60x40	14.06KB	56.25KB	281.25KB	1687.5KB (1.65MB)
30x20	3.52KB	14.06KB	70.31KB	421.88KB
15x10	0.88KB	3.52KB	17.58KB	105.47KB

Table 7.3 - Storage Size Comparison

The average confidence is not the only determining factor when deciding on what image and n-tuple size the system is going to use, average correct response is also a determining factor. Table 7.4 displays the maximum, minimum and average correct responses for all the tests conducted above, in order to observe what difference resizing an image has on the results.

As can be expected, as the image size is reduced, the average correct response tends to increase, regardless of the size of n-tuples being used. Another pattern that is detected in the results below is that a higher n-tuple size always results in a lower maximum, minimum and average response when compared to a lower n-tuple size system.

4-pixel n-tuple			
Image Size	**Max**	**Average**	**Min**
480x320	99.96	89.53	66.2
240x160	99.94	90.15	66.03
120x80	99.96	90.37	67.08
60x40	100	90.56	66.17
30x20	100	90.85	68.67
15x10	100	91.5	72.97

5-pixel n-tuple			
Image Size	**Max**	**Average**	**Min**
480x320	99.87	81.07	54.08
240x160	99.88	82.18	54.04
120x80	100	82.33	53.54
60x40	100	82.59	54.79
30x20	100	83.81	53.33
15x10	100	84.84	53.33

6-pixel n-tuple			
Image Size	**Max**	**Average**	**Min**
480x320	99.77	70	39.16
240x160	99.84	71.7	41.36
120x80	99.81	71.82	42
60x40	100	71.97	40
30x20	100	73.76	38
15x10	100	75.96	40

7-pixel n-tuple			
Image Size	**Max**	**Average**	**Min**
480x320	99.57	57.78	25.96
240x160	99.67	59.8	25.56
120x80	99.71	60.2	26.84
60x40	99.71	60.28	25.15
30x20	98.82	61.55	24.71
15x10	100	65.72	23.81

Table 7.4 - Comparison of average correct response at different image sizes

What can be deduced from the results achieved in these tests is that resizing an image to a smaller image size may or may not affect the performance of the images, and this depends on the size of image, the factor of resizing being done, and the n-tuple size being used. Therefore it is not possible to categorically state whether or not resizing an image is better or worse for a system, every system is different with regards to the type of images being used, the n-tuple size, the image size, and the amount of storage available per net; therefore how much resizing is to be performed can be optimized per user requirements.

7.2 Image lighting

The system presented in this thesis is known to generalise between patterns, hence it is theorised that this algorithm can achieve high accuracy rates even if the images of the subject have been taken under various lighting conditions. To prove that the system can generalise between different lighting conditions, a database of 3 subjects was created, with all the images being full frontal, with the subjects being instructed to stay as still and expressionless as possible.

A white LED array was used to focus light on the subject in 11 different angles, and 30 images were taken of the subject at each illumination angle; the angles were:

1. LEDs positioned facing the background directly above the subject's head.
2. LEDs facing the background to the top-left of the subject's head.
3. LEDs facing the background to the top-right of the subject's head.
4. LEDs directly facing the subject.
5. LEDs facing the left of the subjects face.
6. LEDs facing the background to the left of the subject.
7. LEDs facing the right of the subjects face.
8. LEDs facing the background to the right of the subject.
9. LEDs facing the subject's chest.
10. LEDs facing down to the subjects feet.
11. LEDs facing the ceiling directly above the subject's head.

Figure 7.8 - Training images for subject 1 - angles 3, 4, 5, 7 and 11.

To prove the generalisation theory, each subject's net was trained on a single image from only five angles – angle 3, 4, 5, 7, and 11, displayed in Figure 7.8 and Figure 7.10.

Figure 7.9 – Subject 1 lighting angles that were not trained on - 1, 2, 6, 8, 9 and 10

Figure 7.10 - Training images for subject 2 - angles 3, 4, 5, 7 and 11.

Figure 7.11 - Subject 2 lighting angles that were not trained on - 1, 2, 6, 8, 9 and 10

The rest of the 295 unique images, including images from angles 1, 2, 6, 8, 9, and 10 (displayed in Figure 7.9 and Figure 7.11) were tested on using a 5-pixel n-tuple system with random mapping. The results of this test are shown in Figure 7.12 below.

Figure 7.12 - Results of Lighting Test on 3 subjects, training on 5 images and testing on 325 unique images

It can be observed from the graph above that all of the images have been accurately classified, with a 0% FAR and FRR. The system achieved 100% accuracy, with the response percentages between 49% and 100%, and the majority of correct responses being between 70% and 90%. This is proved by the confidence graph shown in Figure 7.13 below.

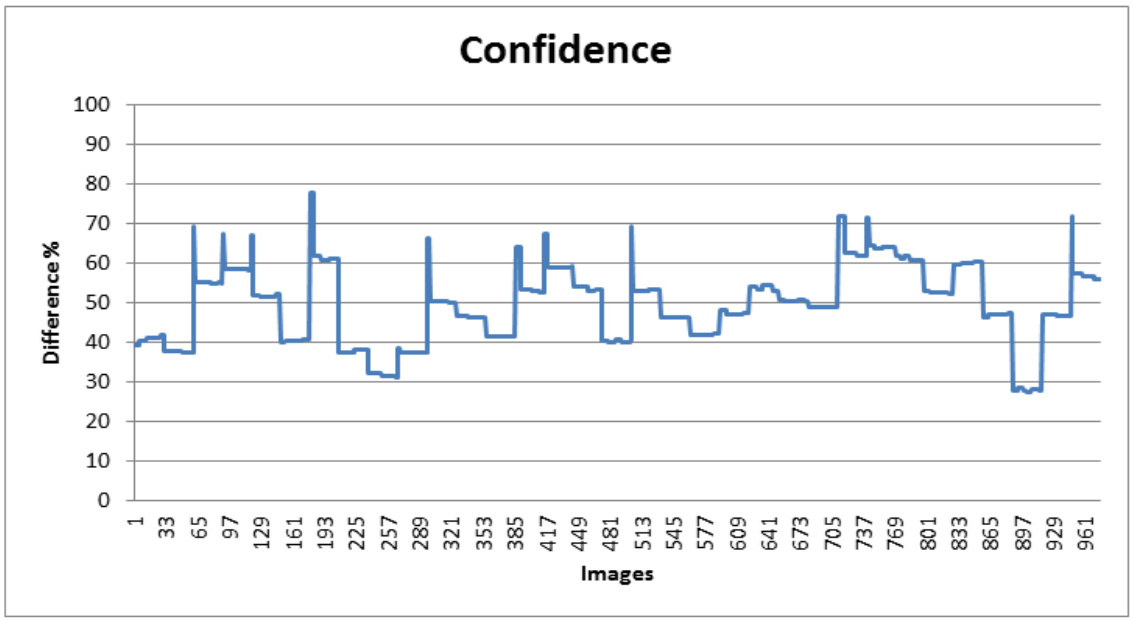


Figure 7.13 - Confidence of 5-pixel n-tuple system on the Lighting Test

Although the percentage response drops below 50% for some images, the system clearly generalises between the various illumination conditions, and even though only 1 image from each of the 5 illumination conditions was trained on, all 975 images from 11 different illumination conditions were accurately classified!

7.3 Image zoom

As discussed in Chapter 1 and 2, alongside variation in illumination, image zoom is another factor that greatly affects system performance. As the methodology presented in this thesis does not perform face detection as a pre-processing step to the recognition, it is imperative that the effect of image zoom is tested, to analyse how the performance is affected by it. It is important to note that the zoom being tested is optical zoom, and not digital zoom. Digital zoom is a computing process performed on the image to enlarge or reduce the size of the image, whilst optical zoom does not perform any mathematical process on the image, the camera (or lens) is moved closer or farther away from the subject.

As no suitable face database was found to investigate this crucial factor, a database was created that consisted of 3 subjects, each having 6 levels of zoom. Each class consisted of 240 images, where each zoom level, 10 images were taken for 4 different facial

expressions – smiling, sad, angry, and expressionless. Sample images from each zoom level are found in Figure 7.14 and Figure 7.15 below:

Figure 7.14 - Zoom levels 1-6 for subject 1

Figure 7.15 - Zoom levels 1-6 for subject 2

One image per zoom level was used to train on each of the subjects, and therefore 6 images per subject were trained using a 5-pixel n-tuple system, and 703 images were tested on. The results are shown in Figure 7.16.

Figure 7.16 - Results of zoom test on 3 subjects, 6 images per subject trained on, and 702 images tested on

The responses on the graph above clearly show that the system achieves 100% accuracy for all 702 images of different zoom levels and different facial expressions; however it is important to notice the effect zoom has on the performance. To see this effect better, the results of only the images belonging to class 1 have been enlarged, and displayed in Figure 7.17. From the results graph, it can be noticed that as the zoom levels increase, the number of standardised background pixels increase, and therefore even though the facial expression has not been trained on, the system achieves a high percentage response.

Due to zoom level 1 containing a small percentage of background pixels, a change in facial expression causes a significant drop in percentage response, whereas a change in facial expression does not affect the higher zoom levels, where the percentage of background pixels is high.

A visual comparison of the two extreme zoom levels (1 and 6) with regards to a change in facial expression can be found in Figure 7.18 and Figure 7.19.

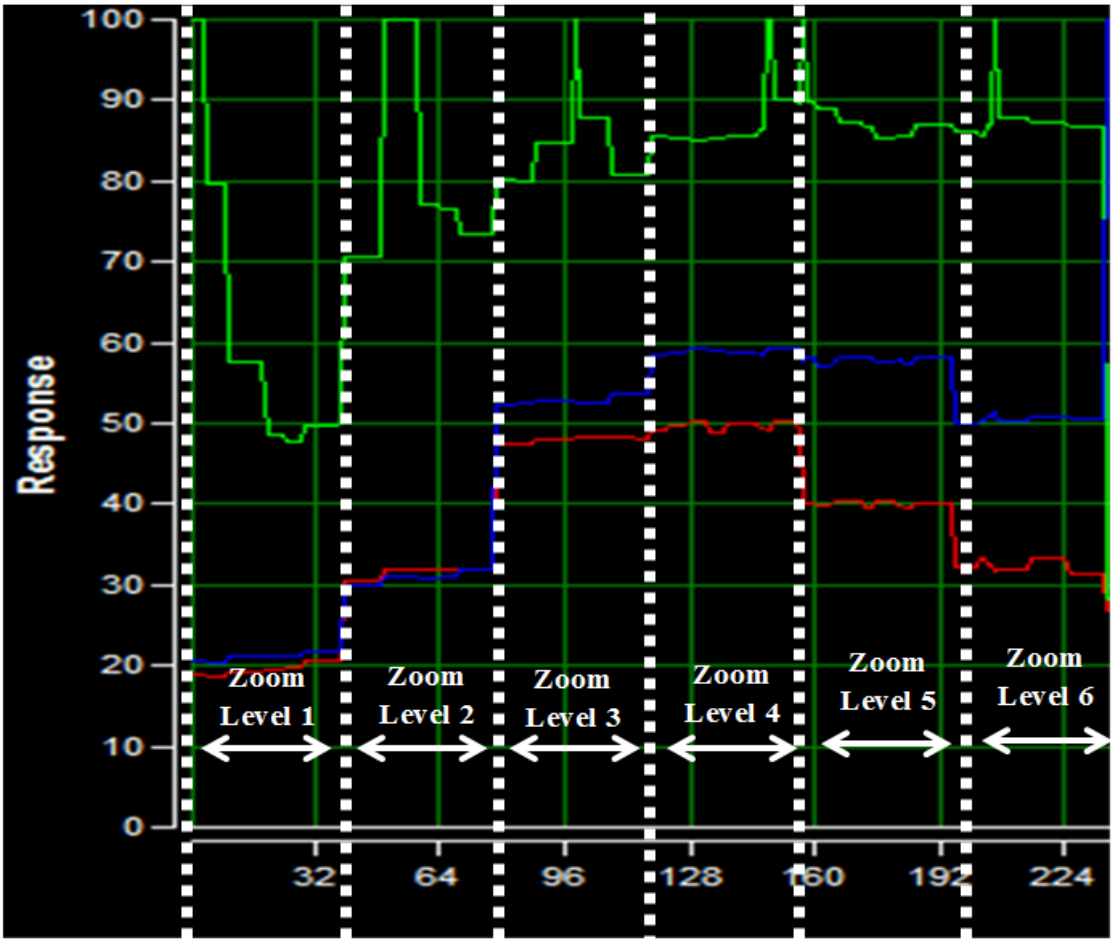


Figure 7.17 - Subject 1 results from zoom test - investigating the effect of zoom on system performance

Figure 7.18 - The 4 different facial expressions displayed in zoom levels 1 and 6 for subject 1

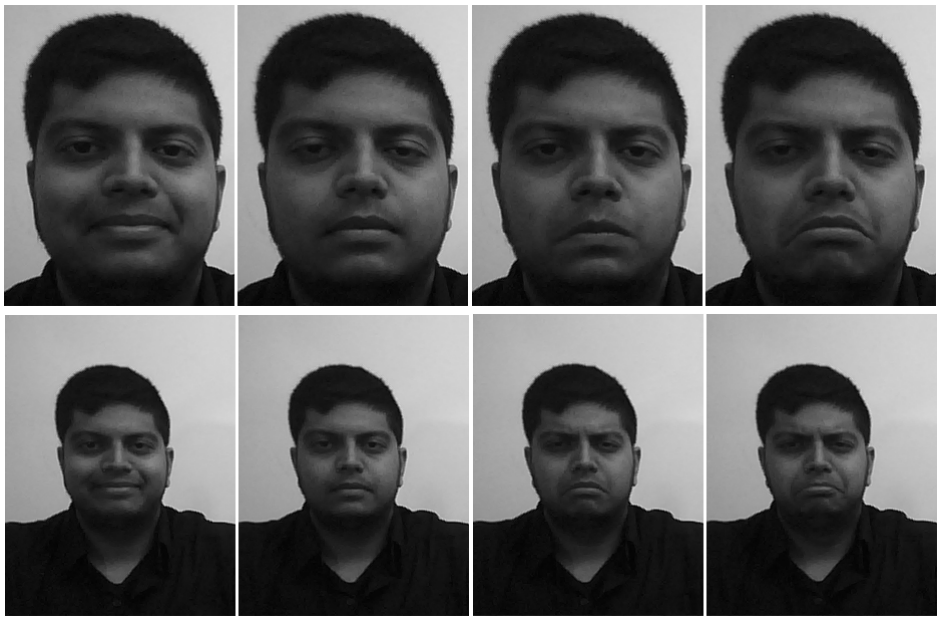


Figure 7.19 - The 4 different facial expressions displayed in zoom levels 1 and 6 for subject 2

It should be noted that although zoom level 6 in all the subjects consists of at least 35% of the same background pixels, with similar illumination conditions, the system still achieves a minimum of 24% confidence, as shown in the confidence graph in Figure 7.20 below.

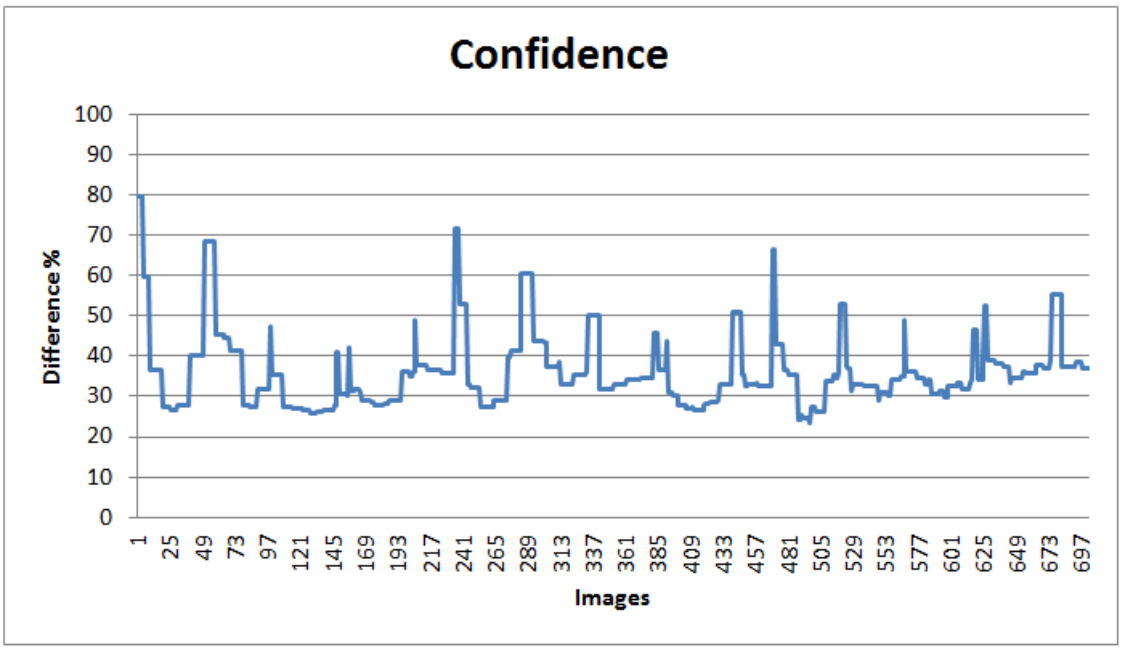


Figure 7.20 - Confidence graph of zoom test on 3 subjects

7.4 Conclusion

This chapter investigated the effect external variables have on the performance of the system, such as image size, illumination and zoom.

To investigate the effect of change in image size, 5 different resizing techniques were used to reduce and increase the size of the same dataset, and tests were run that showed a minute percentage change of between 0.17% and 1.89% in the responses. As this did not affect the overall accuracy of the system, it was concluded that any technique can be used in the system implementation, and nearest neighbour method would be used for tests conducted in this chapter, as it required the least amount of computational intensity.

The 'Jenkem' video file from Chapter 4 was used to test the accuracy of the system at different image sizes, from 480x320 pixels to 15x10 pixels. All six image sizes were tested on using 4, 5, 6 and 7-pixel n-tuple systems, and all systems achieved 100% accuracy rates. However, the main effect image size has on the system is on the function storage required, which increases significantly as n-tuple size or image size increases.

A database was created to test the systems generalisation properties with regards to variation in illumination conditions, and the database consisted of 3 subjects, each having images in 11 illumination conditions. Training on just 5 images, each from different illumination conditions, the system achieved 100% accuracy on all 11 illumination conditions.

The effect of zoom was also investigated by creating a database consisting of 3 subjects, each containing 6 different zoom levels. 10 images were taken of 4 different facial expressions for each of the zoom level, resulting in 240 images per individual. 6 images were trained on per subject, one from each zoom level, and the rest of the 702 images were tested on, and the system resulted in a 100% accuracy, with both FAR and FRR being 0%.

The system has proved robust in conditions that many face recognition systems fail in, and it is proposed that due to this, the methodology can be implemented as a real-time pattern recognition system in a real-world scenario!

Chapter 8 :

Data Security

8. DATA SECURITY

Although the algorithm presented in this thesis is based on pattern recognition, due to its face recognition abilities it is important to ensure that the data stored within the system is secure. If the algorithm is used to design a face recognition system for any purpose, the information that is stored in the nets is a representation of the individual that has been trained on, and thus it needs to be protected.

There are three main aspects of the system that require protection features to be implemented, and they are discussed in the sub-chapters below.

8.1 Random mapping creation

As explained in Chapter 3, the random mapping that is used in a system is the sole determining factor of where each pixel from an image is then located in an n-tuple. If a net is trained on an individual with one random mapping sequence, and then this sequence is changed, the data stored in the net becomes irrelevant, as it is not accurate any more.

Using the same principle, if the random mapping used to create a specific net based on an individual is acquired, together with the stored net of the individual, it is possible to create an image that achieves 100% accuracy on the net, thus rendering the security feature irrelevant. It should be noted here, that even though an image can be recreated from the data stored in the net and the random mapping that was used to train it, this image would not represent the individual at all; this will be explained in detail in Chapter 8.2.

Nevertheless, it is therefore important that the random mapping be secure in the system, and the only way to do that is to change the random mapping constantly. However, as explained before, that would render the trained net useless. In order to overcome this obstacle, it is important to understand how the random mapping has been designed in the current system.

The steps used currently to create a simple unique random mapping sequence for an image size of 100x100 pixels are below:

1. Create a table containing 2 columns and 10000 rows.
2. List a number from 0 to 100 in a sequence in column 2, and continue this until the end of the table is reached.
3. List a number from 0 to 100 in a sequence in column 1, repeating each number 100 times.
4. Initialise the pseudo-random number generator with a seed based on the current time.
5. Shuffle the rows in the table 100 times, with each shuffle being decided by the random function.

Based on the steps above, a table is created and populated with each pixel location in the image sequentially, with the Y-Axis mapping being in the second column, and the X-

Axis in the first. Then, the random sequence to be used is re-initialized based on the current time, which also includes the date, and is therefore always unique), and then random row numbers are created in order to shuffle the rows in the table.

This method ensures that every time the software is executed, a new unique random mapping table is created, this being optimum for testing the algorithm. However, due to the requirement of a standard random mapping sequence per net that is used, this algorithm needs to be changed for real-world implementation.

The only change that is required in order to create the same set of random numbers is the initialization of the random sequence. At the current stage, this is being initialized by the date and time value, and therefore is unique at any given second; however if constant number was used as the seed in the random initialization, the random set of values would be the same every time the software is executed.

Although the possibility of generating the exact same random mapping without knowing the initialization value is low, in order to enhance the security further, it is proposed that a formula be used to create this single initialization value, with a user generated key being used as the values in the formula. Therefore, if an individual was trying to gain access to the data secured by the face recognition system, he would have to first acquire the stored net of that user, then gain the formula used by the system to generate the random mapping, then also find out the key-code generated by the user, and finally reverse engineer an image that can achieve 100% accuracy of the face recognition system.

In order to imagine just how difficult performing all these steps would be, below are the steps an individual would have to take in order to do all of the above:

1. Hack into the server database that contains all the nets, and decipher which net is the correct one.
2. Perform a hardware hack on the microprocessor that performs the random mapping, in order to find out the formula used by the system.
3. Somehow find out the key-code that has been generated by the user.
4. Reverse engineer an image from the data in the net, based on the mapping
5. Perform a hardware hack on the face recognition system to accept the image that has been created, and not the images from the camera that is being used for face recognition.

Step 5 is very important due to the following:

1. The data in the image that has been reverse engineered is pixel specific, and for a high recognition accuracy to be achieved, each pixel has to be in the correct place. Therefore, holding up the reverse engineered image to the camera of the face recognition system would not be accurate enough.
2. As explained in Chapter 7, the system is able to differentiate between the different image mediums, and therefore holding up an image of a face to a camera does not deliver the same accuracy as the actual face itself.

Based on these steps, applying a formula to create the random mapping would assist in creating the highest level of security for the data stored in the face recognition system.

8.2 Safety of trained individuals

As explained in Chapter 3, when a net is created and trained on a subject, the images that are used to train the class are not stored, and once the system has completed training on an image it is deleted. However, the ranks that are extracted from the image are stored in functions, where every single neuron has a function.

Each function is made up of binary bits, where the number of bits is equal to the factorial of the n-tuple size, and the trained net is made up of all these individual functions. Since random mapping is used to determine the location of each pixel with regards to the n-tuples, it can be proposed that the image that has been trained on can be reverse engineered from the available ranks, if the random mapping used to train it is known.

To test this theory, software was designed that creates images based on ranks that have been stored in the functions of the trained net, to see the likeness of the created image to the initial trained image. The database that was used to test this proposal was the MIT-CBCL Test Database, which as explained in Chapter 5 consists of 10 individuals, 200 images per individual, each image at 100x100 to 115x115 pixel resolution. The images were resized to 100x100 pixels using subsample normalization, and as with the test conducted in Chapter 5, the database was divided into two so that 100 unique images were used to train each net. Then, the software was used to extract ranks from each of the trained nets, to create a single image for each net. Figure 8.1 to Figure 8.3 display the original trained image (top) compared to the recreated image (below):

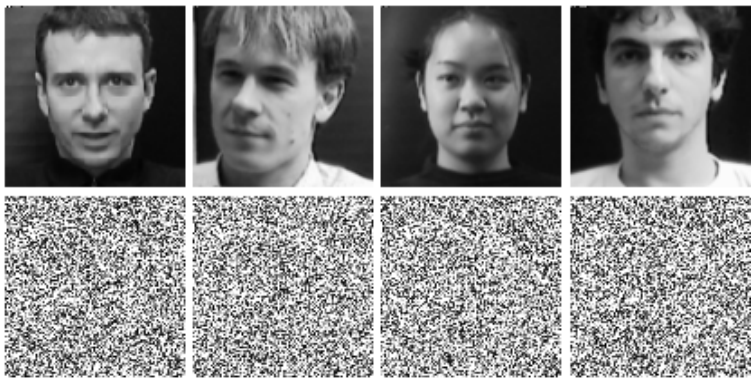


Figure 8.1 - Sample Images from Subject 1 - 4 from MIT-CBCL Test Database Folder

Figure 8.2 - Sample Images from Subject 5 - 8 from MIT-CBCL Test Database Folder

Figure 8.3 - Sample Images from Subject 9 & 10 from MIT-CBCL Test Database Folder

Visually observing these created images result in the theory that these images contain only noise, and that the images contain no specific pattern. However, this is due to the fact that the functions only store the ranks that are attained from the n-tuples, hence it is impossible to obtain or determine the original pixel values from the net. Therefore, what the software does is, for each n-tuple, a stored rank is extracted, and pixel values are created that would arrive at the same rank if passed through the neuron. Then, the random mapping is checked, and each pixel value is allocated to the corresponding pixel in the image.

Theoretically, each created image should result in 100% accuracy when tested against its own net, as all the ranks used to create the image were obtained from the trained class originally. However, due to the images being visually similar to pattern-less noisy images, it was important to check the response obtained when each image is tested against all 10 trained nets; the resulting accuracy of the system is displayed in Figure 8.4 and the response table that was used to plot this graph is in Table 8.1.

As discussed, the response of each created image on its own net is always at 100%, due to all the ranks extracted in the image being present in the trained net. However, even though these images seem to consist of random noise patters, the response of

each image on other nets in the database is always lower than 50%, with the highest false response being 49.21%, and the average being 29.34%.

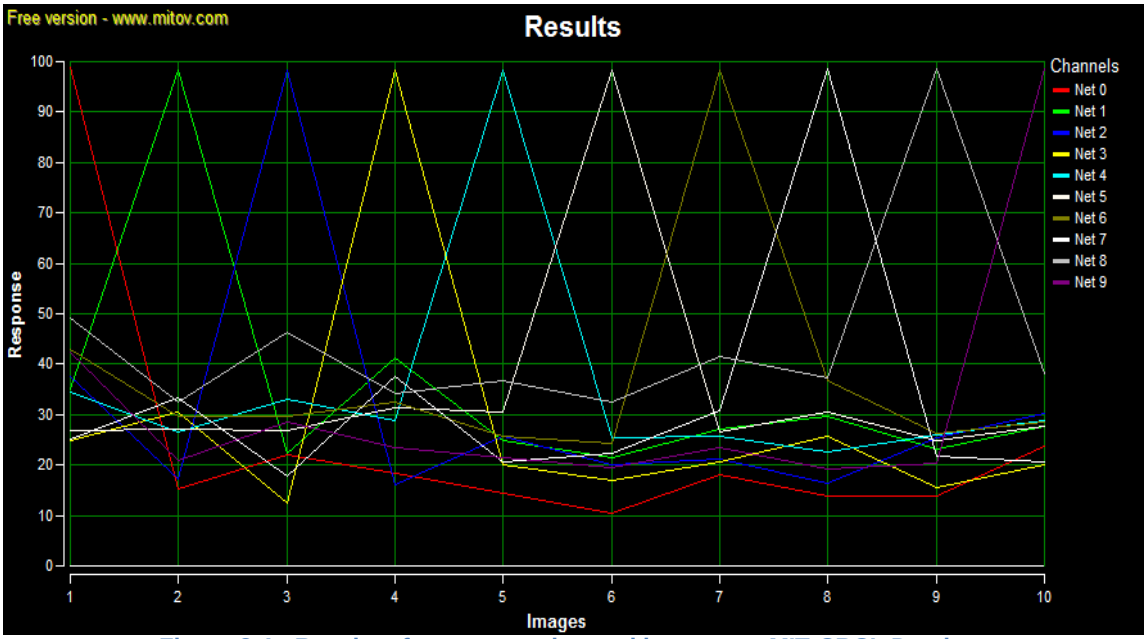


Figure 8.4 - Results of reverse engineered images on MIT-CBCL Database

Net 0	Net 1	Net 2	Net 3	Net 4	Net 5	Net 6	Net 7	Net 8	Net 9	Result
100	35.14	37.57	24.79	34.5	26.93	42.93	25.07	49.21	42.29	Net 0
15.21	100	17.36	30.57	26.43	27	29.79	33.43	32.57	21	Net 1
22.07	22.29	100	12.43	32.93	26.79	29.57	17.79	46.43	28.57	Net 2
18.29	41.36	16.14	100	28.93	31.29	32.5	37.71	34.07	23.43	Net 3
14.5	24.86	25.57	20	100	30.64	25.57	20.64	36.79	21.36	Net 4
10.5	21.57	20.14	17.07	25.29	100	24.36	22.43	32.36	19.43	Net 5
18	27	21.07	20.64	25.64	26.43	100	30.93	41.64	23.36	Net 6
13.71	29.71	16.5	25.57	22.5	30.64	36.71	100	37.29	19.07	Net 7
13.86	23.14	25.5	15.64	26	25	26.21	21.79	100	20.36	Net 8
23.71	27.57	30.14	19.93	28.86	27.79	28.5	20.64	38.07	100	Net 9

Table 8.1 - Response of reverse engineered images on MIT-CBCL Database

Seeing as images can be reverse engineered from stored nets, that result in the same ranks as those that exist in the trained net, it can therefore be proposed that if the system is required to store the images that are used to train the system for later training and testing, and yet security concerns prevent the storage of images that show the faces of the subjects, or the amount of storage required to keep all the images is great, this method can be used to accurately retain the information required by the system.

For example, if a 5-pixel n-tuple system is designed to train on images of 480x320 pixel resolution, and 200 images are used to train a single net, each image being around

450KB; and if a requirement of the system is that each trained image should be stored for later training or testing purposes, the database would require approximately 88MB to store the images of each net.

However, if after training a net on the 200 images, the maximum number of ranks that are set in any of the 43,520 (120 bit) functions is 40, a total of 40 images would need to be created and stored, in place of the 200 original images. Therefore, with just 17.6MB being used to store 40 images, all the data that is extracted from the original images, and that is required by the system can be stored reliably, without any security concerns.

8.3 Net security

As explained in the previous chapter, once images of an individual have been trained on by the system, the images themselves do not exist, however the ranks that have been extracted from these images are stored in the individual's net. In any face recognition system, the net is then stored in an allocated memory location, normally a storage server.

The security of these trained nets solely depends on the security of the server itself, based on the location of the physical server, and the software securities surrounding access to the data stored within it.

If a real-world system was to be designed and implemented based on the algorithm presented in this thesis, the storage server would need to be secured in two ways:

1. Physical access to the server would need to be limited to authorized personnel only, as any individual that has physical access to the server could access the data within the hard drives by simply booting the server from another operating system, thus rendering any password implementations useless.
2. In order to secure the data in the hard drive in the event of it being booted by another operating system, the nets stored in the server should be encrypted, with the decryption algorithm based on the same user-generated pass-key that has been explained in the section above.

Although obtaining an individual's net is not enough to gain access to his secured data, as explained in the previous sections, nevertheless it is important that the net be protected as an added precaution, which will increase the security of the face recognition system.

8.4 Conclusion

In the previous chapters, the methodology, accuracy, and variability of the face recognition algorithm being presented in this thesis has been discussed, however it is important to also investigate the methods of securing the data obtained and used by the system itself.

In this chapter, a different method for the creation of random mapping sequences was proposed, one that would ensure that the random mapping itself would remain unique per individual, and secure. Then, the trained nets were discussed, displaying the fact that images resembling the trained individual cannot be reverse engineered from a trained net, however images that would represent the same accuracy when tested against the system could.

Based on this, the security of the trained nets themselves was proposed, with the implementation of data encryption, and the limitation of access to the physical server, the stored nets would remain safe from malicious intent.

By merging all these security measures, the safety of the data stored in the system, and the algorithm itself is ensured, hence allowing the implementation of a face recognition system to be a secure alternative to other systems available today.

Chapter 9 :

Conclusion

9. CONCLUSION

9.1 Summary of research

Pattern recognition and Face recognition have been researched for decades, and although humans are able to recognise sounds and images on a daily basis, no machine vision system has been designed that can operate accurately regardless of illumination, viewing distance or image noise. This thesis is a presentation of the research performed on Pattern recognition and Face recognition in order to emulate the brain in designing a simple, fast, and efficient recognition system.

In the past various techniques have been used in the design of pattern recognition systems, such as template matching, syntactic approaches, statistical, and neural networks (Jain, Duin and Jianchang Mao, 2000), however due to many algorithms being a combination of two or more techniques, it is difficult to classify them. The most frequent techniques used to build face recognition systems are either statistical or neural network-based approaches (Zhao *et al.*, 2003).

Many statistical approaches were reviewed, including template matching, Eigenfaces, Fisherfaces, etc., and although statistical approaches have been used to design commercial pattern recognition systems in the past, they are plagued by the 'curse of dimensionality', and based on this and the numerous other reasons discussed in Chapter 2, the neural network approach was used to design the algorithm presented in this thesis.

Neural networks are divided into two categories, analogue and digital:

- Analogue neural networks perform function approximation, in an attempt to divide the universe in order to classify patterns correctly. Analogue feed-forward neural networks cannot solve linearly inseparable problems such as the Boolean XOR Gate, and although techniques like back-propagation have been designed to overcome this problem, they have long learning times, it is not known when if at all the system would converge, and there is no set criteria for stopping the process.
- Digital neural networks perform pattern classification, where the classes are depicted as clusters in a 2-dimensional universe, and training any pattern from the dataset once would generalise the class significantly, and that pattern would achieve 100% accuracy on the net if tested against the system at any later stage. The first automatic face recognition system was the WISARD system (Aleksander, Thomas and Bowden, 1984), and although this was a revolutionary development, the system still relied on binary image thresholding as the input to the system.

Based on the speed of the system, the simplicity of design, the low computational cost, and the generalisation features of digital neural networks, together with the fact that an accurate efficient biological implementation is used by humans every day, the research focussed on designing a pattern/face recognition system using weightless neural networks, that could automatically recognise and classify image patterns.

The designed system was an adaptation of the WISARD system (Aleksander, Thomas and Bowden, 1984), which was based on the method proposed by Browning and Bledsoe in 1959 (Bledsoe and Browning, 1959). The methodology was first presented by the author in 2012 (Khaki and Stonham, 2012).

The neurons processed groups of data consisting of grayscale pixel values extracted from the image – these are called n-tuples. The location of the pixels are specified by a 'mapping' table, and when the extracted n-tuple is input into its respective neuron, a 'ranking algorithm' is applied to it. During training, the state achieved by each neuron becomes a term of the logic function implemented by the neuron, and during testing the logic function determines whether the state value results in a '1' or '0'; i.e. whether or not the test pattern n-tuple is a state that has occurred during training.

The system can be divided into three parts, with possibility for optimisation in each section:

1. **Pixel value extraction** – there are three different types of mapping:
   a. **Linear Mapping** – although this mapping did not produce any significant errors in the test results explained in this thesis, linear mapping was found to generalise too easily between patterns belonging to the same class, and also between patterns found in other classes. Since the confidence of the system (amount of discrimination in response values) was very low, this mapping was not used in any tests.
   b. **Specific Mapping** – this mapping is proposed to be used for any face detection system that would be built based on the recognition system presented in this thesis. As it ensures that a set pattern is being 'observed' in an image, it is proposed that this mapping would result in high detection accuracy rates.
   c. **Random Mapping** – this mapping was found to adequately generalise and discriminate between both images in the same class, and two different classes. Based on the results obtained, this was the mapping chosen for all tests run in this thesis.
2. **Neuron computation** relates to the grayscale ranking algorithm that is used by the system. The method of optimisation is to increase or decrease the n-tuple size and test it against a standard data set. This was thoroughly investigated in Chapter 6, and it was found that the lower the n-tuple size, the faster the system generalises on trained images, and the lower the confidence value; vice-versa for the higher n-tuple values. It was observed that every situation determined the use of the n-tuple size based on the training net available, and the image size, not to mention the storage space available for the trained nets. It was decided that based on the accuracy, generalisation and discrimination characteristics, and storage requirements, all the video scene filtering tests, and the face recognition tests would be run using a 5-pixel n-tuple system, in order to achieve standardisation with regards to the presentation of the results.
3. **Output calculation** – the response of the system when an image is tested against the trained nets can either be outputted as a set of percentage values, or as a net classification, and a threshold value can also be applied to the value in determining the validity of the response. In order to analyse the workings of the system being designed, all experiments presented in this thesis have systems that output both the

set of percentage values, and the net classification, and no threshold value has been applied to determine the validity of the response.

It should be noted that although the system was designed in multiple software platforms, a hardware implementation of the system can be designed using FPGAs that are easily available today. The hardware implementation would process images much faster than the software implementation due to the true parallel nature of the system design; however a permanent storage database would also be required for real world implementations.

As the methodology presented in this thesis performs pattern recognition, two scenarios were used to test its accuracy – video scene filtering, and face image recognition. To test the video scene filtering properties of the algorithm initially, a video of a room was used, with the camera rotating from a fixed position from left to right, and back again. The system was trained on a specific part of the room initially, which contained a birdcage, and then the rotating video was tested on.

The trained system accurately classified the images being tested on with 100% accuracy, and the results of the left-to-right scan of the room is a mirror image of the results obtained from the right-to-left scan of the room. This proves that the system generalises during training adequately enough to allow the responses to not be affected by any camera sensor noise.

To test the discrimination properties of the methodology, a news video of a drug named ‘Jenkem’ was obtained from the internet, which consisted of 3465 individual frames that could be categorised into 25 scenes. Half the frames from 10 people-oriented scenes were trained on, and the entire video was tested on, minus the trained frames. Once again, the system generalised adequately enough to recognise frames not trained on, whilst also discriminating between all the scenes in the video sequence, and achieved an average confidence of approximately 55%, with the overall accuracy being 100%.

To test the face recognition properties of the system, 9 widely available face databases were investigated, out of which 4 databases were chosen which contained only full frontal images of the subjects, with standardised backgrounds that were less than 30% of the overall image, standardised zoom and illumination conditions – the AT&T, Essex Faces94, Essex Grimace, and JAFFE databases. A fifth database was also chosen, the MIT-CBCL Test Database, to test the limitations of the algorithm, as some of the classes within the database contained images with varying backgrounds, face angles and zoom, and illumination conditions.

On all 4 databases that consisted of images with standardised conditions, the system achieved 100% accurate responses with regards to image classification, with False Acceptance and False Rejection ratios being 0%. On the MIT-CBCL Test Database, the system achieved 99.9% accuracy, with 1 image being falsely accepted out of 1000 images. It should be noted that this database was used to test the limitations of the system, where it is impossible for the system to accurately classify images that are not similar to the images that have been trained on.

The effect of change in image size on the overall performance of the system was also investigated, with various resizing techniques being used, and it was found that although different resizing techniques achieve different response values, due to the discrimination capabilities of the weightless neural networks these minute changes in percentage did not affect the overall accuracy of the system. The frames from the video file 'Jenkem' were resized in 5 stages, starting from the original 480x320 pixels size down to 15x10 pixels. Tests were performed using 4 different n-tuple sized systems, and although the average confidence of the systems varied, the methodology showed robustness with regards to the size of the images, and achieved an overall accuracy of 100% in all the tests.

Two databases were created consisting of 3 subjects each to test the systems generalisation properties with regards to various illumination and zoom conditions. For the illumination test, 5 images per subject were trained on, each having been taken during a different illumination condition, and 11 illumination conditions were tested upon, with the system not having 'seen' 6 conditions. Although the response achieved varied between the subjects, the system achieved a confidence of between 30%-70%, with few images achieving 28%; the overall accuracy being 100%.

The generalisation and discrimination properties of the methodology with regards to subject viewing distance was tested by training on 6 images per subject, each at a different viewing distance, with a specific facial expression. The entire database of 702 images between 3 subjects was tested; where at each zoom level images consisted of 4 specific facial expressions, one having been trained on. The system achieved an overall accuracy of 100%, however it was found (as expected) that at an increased viewing distance, where a large number of background pixels are present in the images, the system is not able to accurately discriminate between subject facial expression, but still able to discriminate between the subjects.

The security of the algorithm itself was also discussed, where a different method for the creation of random mapping sequences was proposed, one that would ensure that the random mapping itself would remain unique per individual, and secure. Then, the trained nets were discussed, displaying the fact that images resembling the trained individual cannot be reverse engineered from a trained net, however images that would represent the same accuracy when tested against the system could. By merging these security measures and net data encryption, the safety of the data stored in the system and the algorithm itself is ensured, thus allowing the implementation of a face recognition system to be a secure alternative to other systems available today.

9.2 Discussion

The methodology presented in this thesis is a novel weightless neural network algorithm that is based pattern recognition system that can perform in real-time, processing more than 30 frames a second for 160x120 pixel images. The processing speed of the system is only limited by its serial implementation in software, and due to the parallel nature of the single layer neural network, the system can process and significantly higher speeds if implemented in digital logic hardware.

Even though the universe of possible patterns for a small 8-bit grayscale image is immense, the algorithm is able to generalise adequately between images trained in the same class, and discriminate between different classes, resulting in a minimum accuracy of 99.9% in all the tests performed.

The system has proven robustness with regards to image size and resizing, viewing distance and subject angle, variations in illumination conditions and background pixels, and therefore a real-world implementation of the algorithm is possible in all environments, with the exception of those where detection is required as a pre-requisite. Other detection techniques can be used as a pre-processing technique before the data is input into the system however the accuracy of the overall system then depends on the accuracy of the face detection system.

It is imperative to note that although the system recognises patterns with high accuracy, it can only do so if similar patterns have been trained on; the system cannot generalise between patterns it has not encountered during the training stage, and therefore the type of images and the number of images used in training is one of the most important aspects of the system, which will ultimately lead to the success or failure of the system implementation.

9.3 Future work

Based on the accuracy of the face recognition system presented in this thesis with regards to video scene filtering, and the generalisation and discrimination qualities of the network based on the illumination and zoom tests performed in Chapter 7, it is strongly believed that not only can the methodology be implemented in image recognition scenarios but also in image pattern detection situations, such as object detection and face detection.

Conventional methods in both the research and commercial environments focus on using statistical methods such as feature extraction to train systems to detect patterns in images, however it is proposed that since examples of efficient pattern recognisers exist in nature, there exists a simple solution to the pattern detection problem based on weightless neural networks that has not been discovered.

Preliminary tests were conducted throughout stages of this research on the side, in order to observe the responses of various weightless neural network systems for entire face detection, localised face detection (eyes only, nose only, etc.) and the detection of a specific combination of face features (such as eyes and nose, eyes and mouth, etc.). Due to the focus of the research being pattern/face recognition, no results of any face detection-based tests have been published or discussed in this thesis; however the following observations have been made:

- Even faces that are 20x20 pixels in size, with each pixel being 8 bit grayscale, have too many possible patterns, and therefore training a multitude of different faces saturates the net quickly. The system resulted in better accuracy if the faces were blurred to a certain extent, so texture is assessed, and not specific features. The

amount of blurring required has to be determined, and depends on the size of the face; however this can only be determined after significant investigation.

- Specific mapping would be better suited to face detection than random mapping, however numerous tests have to be performed in the search for the optimum method of specific mapping that has to be used.
- The image database to be used to train a face detection system has to have very strict location parameters, as it is imperative that face angle and/or face zoom do not affect the results obtained in any way. The angle and zoom factors can be investigated only after a significant result is obtained by a specific system design.
- Locating the face in an image is the most computationally intensive process, as the size of the faces in the image are not known, and based on the methodology presented, each size requires a specific mapping, or requires resizing.

Researching an implementation of the methodology presented in this thesis is a few years work, however if an effective algorithm is achieved, then it would be faster, and more cost effective than most if not all face detection systems available. The parallel nature of the system would mean a simple implementation in hardware could possibly process all the faces in an image frame within a few milliseconds with the technology available today, and this would lead to an invaluable asset to law enforcement and security agencies.

BIBLIOGRAPHY

1. Aleksander, I., Thomas, W.V. and Bowden, P.A. (1984) "WISARD·a radical step forward in image recognition", *Sensor Review,* vol. 4, no. 3, pp. 120-124.

2. Aleksander, I. (1966) "Self-adaptive universal logic circuits", *Electronics Letters,* vol. 2, no. 8, pp. 321-322.

3. Aleksander, I. and Morton, H. (1995) *An Introduction to Neural Computing,* Internat. Thomson Computer Press.

4. Aleksander, I. and Stonham, T.J. (1979) "Guide to pattern recognition using random-access memories", *Computers and Digital Techniques, IEE Journal on,* vol. 2, no. 1, pp. 29-40.

5. Aleksander, I., Stonham, T.J. and Wilson, M.J.D. (1974) "Adaptive logic for artificially intelligent systems", *Radio and Electronic Engineer,* vol. 44, no. 1, pp. 39-44.

6. Asselin de Beauville, D.B.J.P. and Mraghni, M.-. (1996) "Unsupervised texture segmentation using grey levels rank-vectors", *Signal Processing, 1996., 3rd International Conference on*, pp. 1273.

7. Austin, J. (1988) "Grey scale N tuple processing" in , ed. J. Kittler, Springer Berlin Heidelberg, , pp. 110-119.

8. Beale, R. and Jackson, T. (2010) *Neural Computing - An Introduction,* Taylor \& Francis.

9. Belhumeur, P.N., Hespanha, J.P. and Kriegman, D. (1997) "Eigenfaces vs. Fisherfaces: recognition using class specific linear projection", *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 19, no. 7, pp. 711-720.

10. Belhumeur, P.N. and Kriegman, D. (1996) "What is the set of images of an object under all possible lighting conditions?", *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, pp. 270.

11. Bellman, R. (1961) *Adaptive Control Processes: Aguided Tour,* Princeton University Press.

12. Biswas, S. and Biswas, A. (2011) "Efficient face recognition algorithms based on transformed shape features", *Imaging for Crime Detection and Prevention 2011 (ICDP 2011), 4th International Conference on*, pp. 1.

13. Bledsoe, W.W. and Browning, I. (1959) "Pattern recognition and reading by machine", *Papers presented at the December 1-3, 1959, eastern joint IRE-AIEE-ACM computer conference*ACM, New York, NY, USA, pp. 225.

14. Boughrara, H., Liming Chen, Ben Amar, C. and Chtourou, M. (2012) "Face recognition under varying facial expression based on Perceived Facial Images and

local feature matching", *Information Technology and e-Services (ICITeS), 2012 International Conference on*, pp. 1.

15. Bremermann, H.J. (1971) "What mathematics can and cannot do for pattern recognition", *Pattern Recognition in Biological and Technical Systems,* , pp. 31.

16. Chan, D., Hockaday, S., Tillett, R.D. and Ross, L.G. (1999) "Factors Affecting the Training of a WISARD Classifier for Monitoring Fish Underwater", *Proc. BMVC*, pp. 34.1.

17. Chellappa, R., Wilson, C.L. and Sirohey, S. (1995) "Human and machine recognition of faces: a survey", *Proceedings of the IEEE,* vol. 83, no. 5, pp. 705-741.

18. Chengjun Liu and Wechsler, H. (2001) "A shape- and texture-based enhanced Fisher classifier for face recognition", *Image Processing, IEEE Transactions on,* vol. 10, no. 4, pp. 598-608.

19. Cognitec (2013) *Cognitec*. Available at: www.cognitec-systems.de (Accessed: 06/24 2013).

20. Digia (2013) *Qt 5.1*. Available at: http://qt.digia.com/ (Accessed: 07/16 2013).

21. Ding, L. and Feng, H. (2009) "Quaternion K-L transform and Biomimetic pattern recognition approaches for color-face recognition", *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on*, pp. 165.

22. Draper, B.A., Baek, K., Bartlett, M.S. and Beveridge, J.R. (2003) "Recognizing faces with PCA and ICA", *Computer Vision and Image Understanding,* vol. 91, no. 1–2, pp. 115-137.

23. Embarcadero (2013) *C++ Builder XE3*. Available at: www.embarcadero.com (Accessed: 07/16 2013).

24. FaceKey (2013) *FaceKey - Superior Biometrics*. Available at: www.facekey.com (Accessed: 07/16 2013).

25. Fairhurst, M.C. and Aleksander, I. (1971) "Natural pattern clustering in digital learning nets", *Electronics Letters,* vol. 7, no. 24, pp. 724-726.

26. Fröhlich, J. (2004) *Backpropagation Net*. Available at: http://www.nnwj.de/backpropagation-net.html (Accessed: 07/06 2013).

27. Fulcher, E.P. (1992) "WIS-ART: UNSUPERVISED CLUSTERING WITH RAM DISCRIMINATORS", *International journal of neural systems,* vol. 03, no. 01, pp. 57-63.

28. Georghiades, A.S., Belhumeur, P.N. and Kriegman, D.J. (2001) "From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose", *IEEE Trans.Pattern Anal.Mach.Intelligence,* vol. 23, no. 6, pp. 643-660.

29. Giles, C.L. and Maxwell, T. (1987) "Learning, invariance, and generalization in high-order neural networks", *Appl.Opt.,* vol. 26, no. 23, pp. 4972-4978.

30. Haykin, S. (1998) *Neural Networks: A Comprehensive Foundation (2nd Edition),* Prentice Hall.

31. Hebb, D.O. (1949) *The organization of behavior: a neuropsychological theory,* Wiley.

32. Hepplewhite, L. and Stonham, T.J. (1996) "Texture classification using n-tuple pattern recognition", *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, pp. 159.

33. Hepplewhite, L. and Stonham, T.J. (1994) "Surface inspection using texture recognition", *Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, pp. 589.

34. Huang, G.B., Ramesh, M., Berg, T. and Learned-Miller, E. (2007) *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*, University of Massachusetts, Amherst.

35. Human Recognition Systems (2013) *Human Recognition Systems*. Available at: http://www.hrsid.com/ (Accessed: 06/24 2013).

36. Jain, A.K., Duin, R.P.W. and Jianchang Mao (2000) "Statistical pattern recognition: a review", *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 22, no. 1, pp. 4-37.

37. James, A. (1986) *The Design and Application of Associative Memories for Scene Analysis*, Brunel University School of Engineering.

38. Karim, T.F., Lipu, M.S.H., Rahman, M.L. and Sultana, F. (2010) "Face recognition using PCA-based method", *Advanced Management Science (ICAMS), 2010 IEEE International Conference on*, pp. 158.

39. Kawaguchi, K. (2000) *A Multithreaded Software Model for Backpropagation Neural Network Applications,* University of Texas at El Paso.

40. Kerin, M.A. and Stonham, T.J. (1990) "Face recognition using a digital neural network with self-organising capabilities", *Pattern Recognition, 1990. Proceedings., 10th International Conference on*, pp. 738.

41. Khaki, K. and Stonham, T.J. (2012) "Face Recognition with Weightless Neural Networks Using the MIT Database" in , eds. M. Kamel, F. Karray and H. Hagras, Springer Berlin Heidelberg, , pp. 228-233.

42. Kirby, R.L. and Rosenfeld, A. (1979) "A Note on the Use of (Gray Level, Local Average Gray Level) Space as an Aid in Threshold Selection", *Systems, Man and Cybernetics, IEEE Transactions on,* vol. 9, no. 12, pp. 860-864.

43. Li, S.Z. and Lu, J. (1999) "Face recognition using the nearest feature line method", *Neural Networks, IEEE Transactions on,* vol. 10, no. 2, pp. 439-443.

44. Ludermir, T.B., Carvalho, A., Braga, A.P. and Souto, M.C.P. (1999) "Weightless Neural Models: A Review of Current and Past Works", *Neural Computing Surveys,* vol. 2, pp. 41.

45. Lyons, M., Akamatsu, S., Kamachi, M. and Gyoba, J. (1998) "Coding facial expressions with Gabor wavelets", *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pp. 200.

46. Lyons, M.J., Budynek, J. and Akamatsu, S. (1999) "Automatic classification of single facial images", *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 21, no. 12, pp. 1357-1362.

47. Marsalli, M. (2006) *McCulloch-Pitts Neurons*. Available at: http://www.mind.ilstu.edu/curriculum/mcp_neurons/mcp_neuron_5.php?modGUI=212&compGUI=1749&itemGUI=3022 (Accessed: 07/06 2013).

48. McCulloch, W. and Pitts, W. (1943) "A logical calculus of the ideas immanent in nervous activity", *The Bulletin of mathematical biophysics,* vol. 5, no. 4, pp. 115-133.

49. Ming-Hsuan Yang, Kriegman, D. and Ahuja, N. (2002) "Detecting faces in images: a survey", *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 24, no. 1, pp. 34-58.

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

51. Mitov Software (2013) *VideoLab*. Available at: www.mitov.com (Accessed: 07/16 2013).

52. Nakagawa, Y. and Rosenfeld, A. (1979) "Some experiments on variable thresholding", *Pattern Recognition,* vol. 11, no. 3, pp. 191.

53. Nefian, A.V. (2005) *Georgia Tech Face Database*. Available at: http://www.anefian.com/research/face_reco.htm (Accessed: 07/24 2013).

54. Niblack, W. (1985) *An introduction to digital image processing,* Strandberg Publishing Company, Birkeroed, Denmark, Denmark.

55. OpenCV (2013) *Open Source Computer Vision*. Available at: opencv.org (Accessed: 07/16 2013).

56. Otsu, N. (1979) "A threshold selection method from gray-level histograms", *IEEE Transactions on Systems, Man and Cybernetics,* vol. 9, no. 1, pp. 62-66.

57. Patel, D. and Stonham, T.J. (1992) "Texture image classification and segmentation using RANK-order clustering", *Pattern Recognition, 1992. Vol.III. Conference C: Image, Speech and Signal Analysis, Proceedings., 11th IAPR International Conference on*, pp. 92.

58. Patel, D. and Stonham, T.J. (1991) "A single layer neural network for texture discrimination", *Circuits and Systems, 1991., IEEE International Sympoisum on*, pp. 2656.

59. Pattichis, C.S., Schizas, C.N., Sergiou, A. and Schnorrenberg, F. (1994) "A hybrid neural network electromyographic system: incorporating the WISARD net", *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on*, pp. 3478.

60. Phillips, P.J., Hyeonjoon Moon, Rizvi, S.A. and Rauss, P.J. (2000) "The FERET evaluation methodology for face-recognition algorithms", *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 22, no. 10, pp. 1090-1104.

61. Phillips, P.J., Wechsler, H., Huang, J. and Rauss, P.J. (1998) "The FERET database and evaluation procedure for face-recognition algorithms", *Image and Vision Computing,* vol. 16, no. 5, pp. 295-306.

62. Pollack, J.B. (1989) *No Harm Intended: A Review of the Perceptrons expanded edition*.

63. Priddy, K.L. and Keller, P.E. (2005) *Artificial Neural Networks: An Introduction,* Society of Photo Optical.

64. Ridler, T.W. and Calvard, S. (1978) "Picture Thresholding Using an Iterative Selection Method", *Systems, Man and Cybernetics, IEEE Transactions on,* vol. 8, no. 8, pp. 630-632.

65. Rohwer, R. and Morciniec, M. (1998) "The Theoretical and Experimental Status of the n-tuple Classifier", *Neural Networks,* vol. 11, no. 1, pp. 1-14.

66. Rosenblatt, F. (1962) *Principles of neurodynamics: perceptrons and the theory of brain mechanisms,* Spartan Books.

67. Rosenfeld, A. and de la Torre, P. (1983) "Histogram concavity analysis as an aid in threshold selection", *Systems, Man and Cybernetics, IEEE Transactions on,* vol. SMC-13, no. 2, pp. 231-235.

68. Rumelhart, D.E. and McClelland, J.L. (1986) *Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations,* MIT Press, Cambridge, MA, USA.

69. Sakai, T., Nagao, M. and Fujibayashi, S. (1969) "Line extraction and pattern detection in a photograph", *Pattern Recognition,* vol. 1, no. 3, pp. 233-248.

70. Samaria, F.S. and Harter, A.C. (1994) "Parameterisation of a stochastic model for human face identification", *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*, pp. 138.

71. SeriousBizness123 and Fox News (2007) *Fox News Reports on Jenkem*. Available at: http://www.youtube.com/watch?v=2UsNbsjpuLc (Accessed: 17 February 2012).

72. Sezan, M.I. (1990) "A peak detection algorithm and its application to histogram-based image data reduction", *Computer Vision, Graphics, and Image Processing,* vol. 49, no. 1, pp. 36.

73. Sezgin, M. and Sankur, B. (2004) "Survey over image thresholding techniques and quantitative performance evaluation", *Journal of Electronic Imaging,* vol. 13, no. 1, pp. 146-168.

74. Sinha, P., Balas, B., Ostrovsky, Y. and Russell, R. (2006) "Face Recognition by Humans: Nineteen Results All Computer Vision Researchers Should Know About", *Proceedings of the IEEE,* vol. 94, no. 11, pp. 1948-1962.

75. Spacek, L. (2007a) *Collection of Facial Images: Faces94*. Available at: http://cswww.essex.ac.uk/mv/allfaces/faces94.html (Accessed: 07/23 2013).

76. Spacek, L. (2007b) *Collection of Facial Images: Faces95*. Available at: http://cswww.essex.ac.uk/mv/allfaces/faces95.html (Accessed: 07/23 2013).

77. Spacek, L. (2007c) *Collection of Facial Images: Faces96*. Available at: http://cswww.essex.ac.uk/mv/allfaces/faces96.html (Accessed: 07/23 2013).

78. Spacek, L. (2007d) *Collection of Facial Images: Grimace*. Available at: http://cswww.essex.ac.uk/mv/allfaces/grimace.html (Accessed: 07/23 2013).

79. Stewart Bartlett, M., Lades, M.H. and Sejnowski, T.J. (1998) "Independent component representations for face recognition", , pp. 528-539.

80. Stonham, T.J., Aleksander, I., Camp, M., Shaw, M.A. and Pike, W.T. (1973) "Automatic classification of mass spectra by means of digital learning nets", *Electronics Letters,* vol. 9, no. 17, pp. 391-394.

81. Stonham, T.J. and Shaw, M.A. (1975) "Automatic classification of mass spectra by means of digital learning nets-existence of characteristic features of chemical class in mass spectra", *Pattern Recognition,* vol. 7, no. 4, pp. 235-241.

82. Tambouratzis, G. (1996) "Applying logic neural networks to hand-written character recognition tasks", *Tools with Artificial Intelligence, 1996., Proceedings Eighth IEEE International Conference on*, pp. 268.

83. Tan, H. and Zhang, Y. (2008) "Expression-independent face recognition based on higher-order singular value decomposition", *Machine Learning and Cybernetics, 2008 International Conference on*, pp. 2846.

84. Thamizharasi, A.M.E. (2010) "Performance analysis of face recognition by combining multiscale techniques and homomorphic filter using fuzzy K nearest neighbour classifier", *Communication Control and Computing Technologies (ICCCCT), 2010 IEEE International Conference on*, pp. 643.

85. Turk, M. and Pentland, A. (1991) "Eigenfaces for Recognition", *Journal of cognitive neuroscience,* vol. 3, no. 1, pp. 71-86.

86. Ullmann, J.R. (1969) "Experiments with the n-tuple Method of Pattern Recognition", *Computers, IEEE Transactions on,* vol. C-18, no. 12, pp. 1135-1137.

87. Vatsa, M., Singh, R. and Gupta, P. (2004) "Face recognition using multiple recognizers", *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, pp. 2186.

88. Wang, C., Lan, L., Zhang, Y. and Gu, M. (2011) "Face Recognition Based on Principle Component Analysis and Support Vector Machine", *Intelligent Systems and Applications (ISA), 2011 3rd International Workshop on*, pp. 1.

89. Weber, M. and Caltech (2005) *Faces 1999*. Available at: http://www.vision.caltech.edu/html-files/archive.html (Accessed: 07/24 2013).

90. Weszka, J. and Rosenfeld, A. (1979) "Histogram Modification for Threshold Selection", *Systems, Man and Cybernetics, IEEE Transactions on,* vol. 9, no. 1, pp. 38-52.

91. Weyrauch, B., Heisele, B., Huang, J. and Blanz, V. (2004) "Component-Based Face Recognition with 3D Morphable Models", *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, pp. 85.

92. Wickert, I. and Franca, F. (2002) "Validating an unsupervised weightless perceptron", *Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th International Conference on*, pp. 537.

93. Wickert, I. and Franca, F. (2001) "AUTOWISARD: Unsupervised Modes for the WISARD" in , eds. J. Mira and A. Prieto, Springer Berlin Heidelberg, , pp. 435-441.

94. Wilson, M.J.D. (1986) "Adaptive windows: Edges, stereopsis and stripes", *Pattern Recognition Letters,* vol. 4, no. 5, pp. 351.

95. Yuille, A.L., Cohen, D.S. and Hallinan, P.W. (1989) "Feature extraction from faces using deformable templates", *Computer Vision and Pattern Recognition, 1989. Proceedings CVPR '89., IEEE Computer Society Conference on*, pp. 104.

96. Zhao, W., Chellappa, R., Phillips, P.J. and Rosenfeld, A. (2003) "Face recognition: A literature survey", *ACM Comput.Surv.,* vol. 35, no. 4, pp. 399-458.