Faculty of Science & Engineering
Department of Electrical & Computer Engineering

A Study of Hierarchical Concatenation Networks in the Area of Pattern Recognition

Irwan Ramli

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University

December 2018

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgment has been made.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Signature:

Date: 07/12/2018

Acknowledgement

Firstly, I would like to thank my supervisor A/Prof. Cesar Ortega-Sanchez. I made a communication regarding my research topic with Cesar before I came to Australia. We discussed the possibilities of my research topic and ended with serious consideration of the topic that I brought into the candidacy. Over the next three years with sweat and tears conditions, a PhD thesis was written. Throughout this period Cesar was always there to do weekly discussion on the research idea, proofread my paper, and of course gave comments and edited my thesis. He spent much time with me as it was far beyond what I was expected. I would also like to thank him for his contribution in supporting my travel which allowed me to attend several international conferences. I would also like to thank my associate supervisor A/Prof. Iain Murray for all his helps and guidance during my PhD research. I would like to thank staffs and colleagues in the department of electrical and computer engineering who involved in my survey. I would also especially to thank Azadeh Nazemi who spent time for the discussions in the field. I would like to thank my sponsor (The Ministry of Research, Technology and Higher Education, Directorate General of Science, Technology and Higher Education of Indonesia) for the financial support of my PhD. I would also like to thank my institution (Politeknik Manufaktur Negeri Bangka Belitung) which gave me a chance to continue my study.

Most importantly, I would like to thank my parents. As a son of yours, you always give me the mental support all the time and remind me to not forget to pray. Finally, I would like to thank my wife Dina, my sons Afiq and Mirza, for all their love and support they have given to me all the time.

Irwan Ramli

Abstract

Feature extraction in pattern recognition has been researched for many years. Some works have been inspired by biological systems, particularly neural networks. As the biological system is not fully understood, this thesis proposed an algorithm inspired on the way humans see objects. Humans can tell two objects look similar when both objects have a high percentage of similarity. The similarity of individual features in both objects is measured before deciding how similar they are. Small features construct bigger features of the object, and at the end the combination of some bigger features make up the whole object. This process of recognising patterns following a hierarchical approach inspired the research presented in this thesis; hence the proposed algorithm is called the Hierarchical Concatenation Network (HCN). When two objects are compared, one object is used as the reference object in the comparison. Hence the HCN analyses the features of patterns using other patterns' features as reference.

As features construct a whole object, the HCN proposed in this thesis has been designed to have several levels. The research investigated how many levels the HCN should have to produce better results. This led to investigate different approaches to concatenate low level features to represent the higher level features. Experimental results on measuring the similarity between specimens led the research to explore alternative ways to represent the high level features, hence this thesis reports on two feature activation methods called HCN-I and HCN-II.

Preliminary results showed that the higher the level of the network, the more differentiated the two objects were. Exchanging the position of the tested and reference patterns produced different result. Hence, different methods of feature extraction were investigated to understand the effect of the relative position between the tested and reference patterns. This resulted in the proposed HCN having two methods called 'one to many' and 'many to one'.

To validate HCN networks, a very simple dataset was used in similarity measure tests. The dataset consisted of ten groups of numbers (0 to 9) with ten different style. This dataset was chosen because it contains patterns that are easily identified by humans, which simplified the task of determining if the HCN could achieve recognition or not. More complex datasets that included numbers with various noise levels (extremely rotated or extremely noisy) were not included in the experiment. Results showed that the HCN's performance degrades as the tested patterns' complexity increases. In these tests the HCN was expected to be able to classify test numbers in the correct group, even though their style was different according to their similarity rate. To improve the HCN's performance, some possibilities were investigated, such as overlapping the middle area of the images, and circular and non-circular-shifting (sequential and non-sequential) of the image during the feature extraction process.

The ability of HCNs to classify as well as recognise patterns was also investigated in this research. For all HCN feature extraction methods, three different ways of classification: classification by union, classification by average, and classification by distance were investigated. Referring back to the way humans recognise objects by combining features, the thesis investigated one method of classification called 'classification by union'. In this method, the features of images within the same class are grouped together. This means that a node in a layer of a class contains patterns with similar unique features. Classification is achieved by selecting the highest similarity rate between the features of the tested pattern and the features of a class, which were previously determined.

'Classification by union' combines the features of pattern in the same class, while 'classification by average' takes an average of features of patterns from the same class. This means that a node in a layer of a class has an average value of the features of all patterns in the same class. Similar to the 'classification by union', the classification result is determined by the highest similarity rate between the features of the tested pattern and the features of a class.

The last classification method investigated in this research was different from the two previous methods. This classification method uses standard Euclidean Distance to measure similarity. As the distance of a tested pattern to a class of patterns needs to be determined , this thesis investigated three different distance measurement methods:

distance to mean, mean of distance, and minimum distance. The ‘distance to mean’ method calculates the distance between the features of the tested patterns to the mean of features of the reference patterns. The ‘mean of distance’ method calculates the distance of tested patterns’ features to all features of the reference patterns in the same class. The distance between a tested pattern and a class is the average of their distance. The last distance measure to be considered was the ‘minimum distance’. Similar to ‘mean of distance’, the distance of tested patterns’ features to all features of the reference patterns in the same class is measured, but the distance between a tested pattern and a class is the minimum distance between the tested pattern and one of the patterns in that class.

In terms of the topology of the network, experimental results show that only two levels are enough for HCNs to perform recognition. More levels lead to over distinction among patterns and give incorrect results.

At the end of the investigation, two publicly available datasets (USPS and MNIST) were used to test the HCN network. These datasets were used because they were written by different people with different styles. To feed these datasets to the HCN network, they were first normalized (size and binarization). The experimental results show that HCN is able to increase its performance by increasing the size of the datasets, although the benchmark method LDA still outperformed HCN’s in recognising a big number of datasets. However, the HCN was able to show better performance with small datasets compared to LDA and PCA.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

All processes that make vertebrates act might be controlled by their brain. Even finger tipping is under the brain's control without realizing it. This control process is a unique and amazing operation that all creatures possess. However, the way the brain works is still not fully understood, and to this day, there is still no evidence to prove that current technologies are able to replicate the whole brain's capabilities.

For decades, scientists have been conducting many researches related to the brain in different areas including psychology, computer science, engineering, and neurology (Balaban & Gulyaeva, 2016; Mayer, 1998; Pompe, 2013). Some applications have been developed from this inter-disciplinary knowledge. For example, iris-, fingerprint-, and face-recognition systems make it easy to record personal data that can be used for many purposes such as an absence system, and/or to solve criminal problems and identification processes.

However, although the above are examples of research works related to imitate some humans' abilities into a machine, they are still far removed from the real process of the human brain as it is not fully understood. Replicating the brain process is difficult to achieve, but researchers have developed algorithms that produce results close to the ability of the human brain. The implementation of developed algorithms into applications that are run by machines makes our lives easier. For example, machines can do repeated tasks over a long period of time with minimum error rates compared to a human.

Research in this area belongs to certain disciplines with the Artificial Intelligence discipline. Many methods and algorithms of artificial intelligence have been studied and tested to solve specific problems such as decision-making (Dragulescu & Albu, 2007; Vari & Vecsenyi, 1988) and prediction (Daynac, Cortes-Cabrera, & Prieto, 2015; Khamehchi, Rahimzadeh Kivi, & Akbari, 2014). Each method or algorithm performs specific tasks by replicating one of the brain's abilities. For example, image or pattern recognition and classification algorithms propose computers to be able to perform the tasks related to visual information processing that supposed to be close to the humans' ability. Voice recognition seeks to imitate the brain's ability to process voices.

Neural network has been a part of research area in the artificial intelligence since the first time it emerged. Researchers continue to develop neural networks such as convolutional neural networks, residual networks, and densely connected networks. However, this research area is being continued to find the best solution if the network becomes deeper and wider (K. H. and, and, and, & Sun, 2016; K. S. and & Zisserman, 2014; He, Zhang, Ren, & Sun, 2016; Huang, Liu, & Weinberger, 2016; Wen, Fu, Sun, Sun, & Wang, 2018; Zagoruyko & Komodakis, 2016). Yet, the knowledge of how exactly the brain performs its tasks is not known. For example, how does the information flow and how is it used in each level of the brain? This lack of total understanding causes difficulties for computer scientists when attempting to develop an algorithm to replicate the biological process of the human brain. However, it is not the objective of the researches to replace the human brain with a machine's algorithms. Instead, it is just to replicate the behaviour of the human activity with computer algorithms. If a machine can help make decisions or do specific tasks, the machine algorithm will be useful for human life. Figure 1.1 illustrates three objects that are easy for human to recognize them.

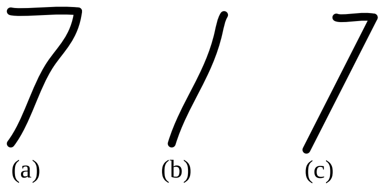


Figure 1.1: Examples of objects

As a human, we can instantly conclude that Figure 1.1(a) and (c) looks similar compared to (b). Without looking at how the brain works in this case, we see each object first and then compare each features they have. Both objects (a) and (c) have two features (two lines) while (b) only has one feature (one line). At this point, it is clear that object (a) does not look similar to object (b). But can we say that object (a) is similar to object (c)? The answer is “yes we can” even though both features belong to object (a) and (c) are not 100% similar. Figure 1.2 gives more detail of possible process comparing the three objects by their features.

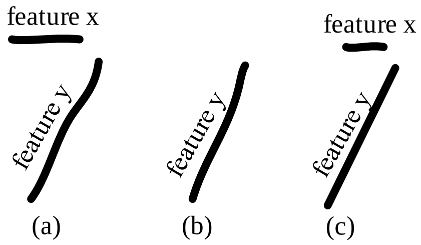


Figure 1.2: Features of patterns

As an illustration, we are comparing three objects in Figure 1.2 feature by feature. At this point, we use image (a) as the reference object to be compared with the two others. The first comparison is between image (b) and (a). When we see feature y of image (b), we can say that this feature looks similar to feature y in image (a). To indicate there is a similarity between them, we can imagine that feature y of image (a) as an active feature. With one feature of image (a) is active, we can say image (b) could have several percentage of similarity to image (a). For the comparison between image (c) and (a), both features x and y would be active as they look similar. We can conclude that image (c) and (a) has higher percentage of similarity rather than image (b) and (a).

The way humans see, separate, and compare the features of the objects as illustrated above is an interesting process, but the way how the brain process the information come to it is very complex tasks. A decade ago, a book called “On Intelligence” was written by Jeff Hawkins (Hawkins & Blakeslee, 2005). This book presents the conceptual theory of the neocortex as the part of the brain that processes external information through sensory systems such as the eyes, the tongue, ear, nose, and skin, receiving visual information, taste, sound, smell, and touch, respectively. A few years after the book was published, an investigation into how the brain might work was conducted (George, 2008b). This work was inspired by Hawkins's book and since that time, Hawkins's theory has become more popularly known as hierarchical temporal memory, or HTM. It is a biological-inspired computer algorithm discussed in (George, 2008a; George & Hawkins, 2005b; George & Hawkins, 2009). It is further investigated in several research works (Chen, Wang, & Li, 2012; George & Hawkins, 2009; Kalmar & Vida, 2013; Kostavelis, Nalpantidis, & Gasteratos, 2012; Melis, Chizuwa, & Kameyama, 2009; Sherwin & Mavris, 2009). The works related to Hawkin's concept tried to implement the intelligence system from biological point of view while the truth of the biological systems work is still not fully understood. Even though Jeff Hawkins stated in his book that studying the detailed biology of brain could be the best way to understand the intelligence, filling the gap between the truth of the brain-based intelligence and the computer ability could be another way of understanding the intelligence concept. This thesis is supposed to study the basic way humans see the object and implement that way within the computer term. The concept of spatial and temporal pooler in the HTM network has inspired this thesis to investigate the process of humans see the object as illustrated in Figure 1.2. It is started with hierarchical feature extraction procedure and ended with the ability of recognition or classification as the end product of the algorithm.

Figure 1.3 shows the possible way of hierarchical feature separation from the whole object within m square area defined as coincidence array in this thesis.

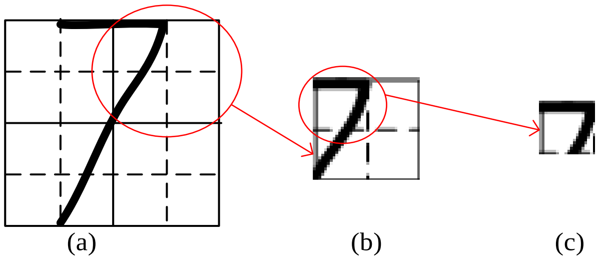


Figure 1.3: Example of extracting feature

Figure 1.3(a) shows the view of the whole image called the top-level image. Four concatenated features from the level below construct this image. One red circle in image (a) shows the concatenation of four square indicated by the red circle. This figure illustrates that image (b) is one feature that construct the image (a), while image (c) is one of the feature that construct image (b).

The example above rises several research questions to investigate the way humans see the object. How to extract features of an object using another object's features hierarchically? How to represent the new features on upper level? As the proposed algorithm extracts the features of an object using other objects' features, is there any difference in terms of feature similarity by exchanging the position of tested and reference patterns? Is there any difference in terms of similarity measure between the features on the lowest level and the features on upper level? If yes, how many levels the network should have? What method is suitable to perform classification? These are major questions that will be addressed in this thesis. Several experiments need to be conducted to answer those questions. Hence, digits datasets that represent known patterns were applied in the experiment.

The literature reveals that the development in the area of artificial intelligence has grown very fast. (Alom et al., 2018; Sze, Chen, Yang, & Emer, 2017) presented the big picture of the development area in the context of artificial intelligence. They did not put HTM in the picture, even though HTM algorithm is also inspired by brain. HTM has inspired this thesis by simplifying its process. Figure 1.4 could be an illustration of the position of this thesis within the picture of (Alom et al., 2018; Sze et al., 2017).

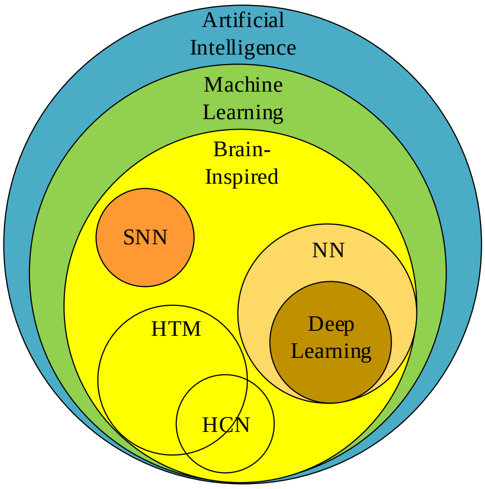


Figure 1.4. HCN position in the context of Artificial Intelligence

The remaining of this thesis is structured as follows. Chapter 2 organizes the major themes of the background knowledge, the theoretical and methodological findings in the field of pattern recognition. It starts with an overview of early works, showing how objects or images are represented in the area of pattern. It then reviews related works in pre-processing and feature extraction to support this research. One particular hierarchical network called Hierarchical Temporal Memory (HTM) is reviewed to analyse specific processes such as input extraction, grouping, and activation of patterns in the upper layers. At the end of this chapter, the general process of image or pattern recognition is explored.

In Chapter 3, the methodology used in this thesis is presented. As this research aims to verify the hypothesis written in subsection 1.2, a quantitative methodology is used throughout the research. The methodology consists of three stages. The first is the review of hierarchical concatenation networks to assess the possibility of simplifying the concatenation and pattern activation processes within the network. The second stage is the exploration of feature extraction and activation methods to investigate how the hierarchical concept is achieved. The last stage is the implementation of a hierarchical concatenation network for classification. The performance of the network at classifying patterns is also investigated.

Chapter 4 presents a step-by-step description of the hierarchical concatenation method to recognise patterns.

Chapter 5 describes the implementation of the hierarchical concatenation algorithm to conduct feature extraction and activation.

Chapter 6 describes the ability of the network developed in this research to classify patterns.

Chapter 7 concludes this thesis with a summary of the key contributions, and presents possibilities of future works.

Chapter 2
Background Knowledge

2.1. Introduction

Human ability to recognise objects or patterns is amazing. It is easy for humans to distinguish what objects they see, what aromas they smell, what sounds they hear, what foods they taste, and what textures they touch. Recognising or classifying objects or patterns through sensory systems is also a natural capability for humans; a very complex process happens almost unconsciously. For example, within a crowded market, humans can easily identify their friends among the people there. They are able to recognise a mango located on the other side of the market from the place where they stand. They can also easily find the exit sign above the exit gate, and are able to evacuate themselves if the fire alarm is triggered. This process of pattern recognition is one example of the ability of the human brain. Neuroscientists have been unable to fully explain the complexities involved in pattern recognition, because most of the complex mechanisms involved happen at an unconscious level. Nevertheless, computer scientists have made significant progress replicating some of these amazing abilities using algorithms implemented by computers.

This chapter presents a brief overview of state-of-the-art pattern recognition. Theoretical and methodological findings in pattern recognition are presented, alongside other related topics like similarity measurement and classification. The following section begins with an overview of early works, showing how objects or images are represented in the area of pattern recognition. It continues with an exploration into replicated brain-inspired concepts and their transference to computer algorithms. It ends with a look at the general process of image recognition, and includes a review of features' extraction, similarity measurement, and classification.

2.2. An Overview of Pattern Representation in Early Pattern Recognition

In terms of recognition, shapes or features among objects are the most important properties to be matched. According to (Stefanucci, 2011), the matching process determines a measurement of similarity between the representation of an object or pattern and a model of the object's subset stored in its memory (see Subsection 2.4.2). (Marr & Nishihara, 1978) state that there are three criteria for shape representation in object recognition. These are:

1. Accessibility, which simplifies the representation of objects and can be useful for recognition.
2. Scope and uniqueness, where the shape representation of objects within the same class should have the same description.
3. Stability and sensitivity, where the degree of similarity between two shapes should be reflected by the description of their properties.

The earliest works in the field of object or pattern recognition used single images of static two dimensional (2-D) scenes. The use of 2-D images in this type of application has been prevalent for more than a half century, or at least since Alan Turing raised the idea with a question: 'Can a machine think?' (Turing, 1950; Williams, 1991).

Some practical experiments, conducted in the early 1960s, are reported in (Gold, 1959; Grimsdale, Sumner, Tunis, & Kilburn, 1959a, 1959b; Taylor, 1959; Unger, 1959). The implementation of simple pattern recognition processes to encode the signal of hand-sent Morse code is conducted in (Gold, 1959). Gold introduces the use of thresholds and error rates in the experiment to give a range of value during signal activation. The patterns that need to be encoded are letters, numbers, and punctuation signs. These signs are represented by simple mark spaces. A threshold is also used to classify the marks and spaces. However, since Morse code relies heavily on the ability of an operator, the error rate is variable and relatively high.

In the same year as Gold’s Morse code experiment, (Grimsdale et al., 1959b; Unger, 1959) presents an automatic system for pattern recognition. With the limited capabilities of computers at that time, the patterns used as inputs are printed on a piece of paper and then scanned as image files. These image files are then stored in a computer as the patterns’ library. Each pattern is analysed, based on groups that represent vertical and horizontal lines, and lines with slopes and curvatures. The first step in the algorithm is scanning the input patterns, pixel by pixel, and identifying the continuity of the lines or curvatures. The scanning process analyses the pattern, based on the groups presented in Figure 2.1.

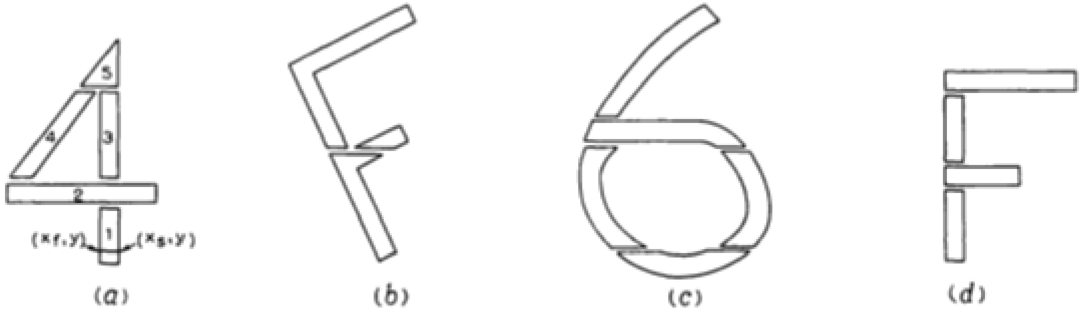
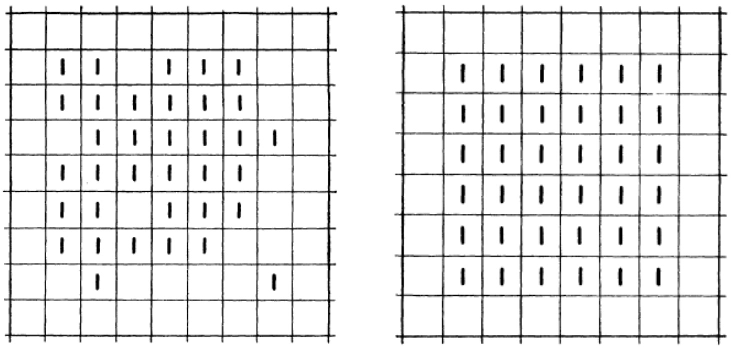


Figure 2.1: Examples of groups of patterns (Grimsdale et al., 1959b)

For example, during scanning, a horizontal line is identified if the line continues with no space at the end (see Figure 2.1 (a), which is a Group 2). The scanned patterns are reproduced by printing the patterns with the highest similarity between the input and stored patterns.

In another work, Unger (1959) implements a noise reduction process that still plays an important role in pattern recognition. The noise reduction process involves certain steps to smooth images, i.e. filling in isolated holes in black areas, filling in small notches in straight-edge segments, eliminating isolated ones, eliminating small bumps along straight-edge segments, and replacing missing corner points. An example of an original pattern along with its smoothed pattern is shown in Figure 2.2.



Original After smoothing

Figure 2.2: Example of noise reduction by smoothing (Unger, 1959)

The use of arrays as the patterns' field, which has been popular in pattern recognition implementation, is introduced by Stearns (1960). Stearns presents patterns as black and white cells within the array. This is similar to pixels, which represent a pattern's binary values. The general approach of Stearn's pattern representation is that the pattern is always to occupy a fixed or finite rectangular area, which is then divided into *n* elements where each element is represented as black or white. Stearn's experiment attempts to perform classification where patterns from the same group are combined together. The combination of patterns in a group is then used as the classifier.

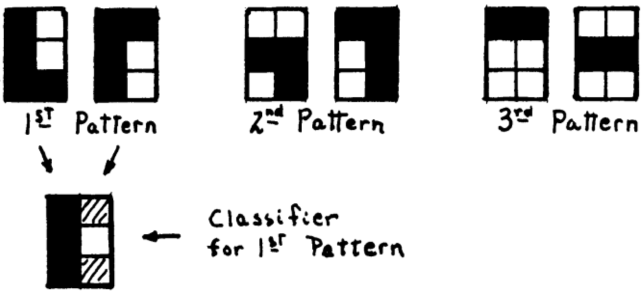


Figure 2.3: Example of forming a classifier using array (Stearns, 1960)

Since that time, pixels representing features of images have been captured in arrays or matrices (Eden, 1962; Horwitz & Shelton, 1961; King-Sun & Rosenfeld, 1976; Levialdi, 1970).

The first concept of a general pattern recognition system is presented by King-Sun and Rosenfeld in 1976. They propose a two-part algorithm as shown in Figure 2.4. The first part is recognition and the second part is analysis or inference. The recognition part consists of pre-processing, extraction, and structural analysis. The analysis part involves selection and structural inference.

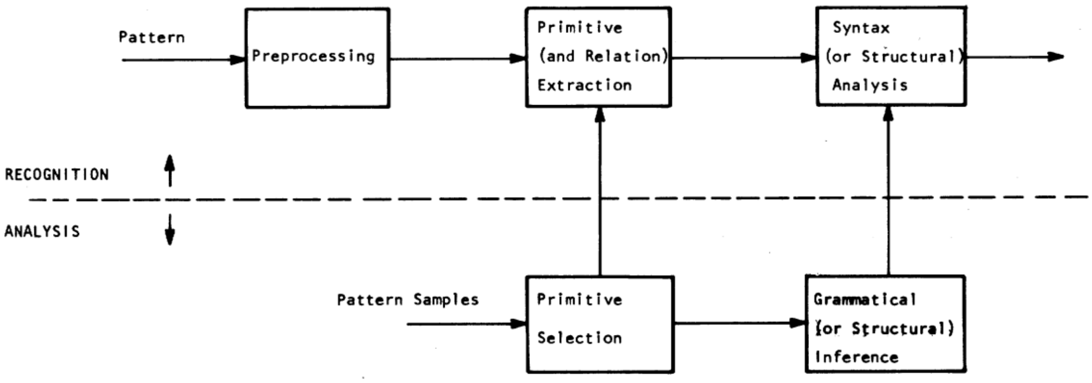

Pattern
Preprocessing
Primitive (and Relation) Extraction
Syntax (or Structural) Analysis
RECOGNITION
ANALYSIS
Pattern Samples
Primitive Selection
Grammatical (or Structural) Inference

Figure 2.4: Block diagram of pattern recognition system as proposed by (King-Sun & Rosenfeld, 1976)

Nowadays, pattern recognition and classification have developed in significant ways. However, the use of arrays to represent an object's or pattern's features, as well as the general concept of pattern recognition and classification, remains the same as in the earliest works.

2.3. Brain-Inspired Intelligence Algorithms

Since Turing's question regarding whether machines can think, research on machine intelligence has been directed at replicating the abilities of the brain, including learning. (Sze et al., 2017) has drawn the big picture of the development of the artificial intelligence as shown in Figure 2.5. Neural networks are considered as the brain-inspired algorithms. With their fast development since the time they were introduced, deep convolutional neural networks aim to achieve the best performance compared with their successor neural networks. On the left-hand side of the brain-inspired algorithm in Figure 2.5, the spiking neural networks (SNNs) are placed, and claimed as more biologically realistic compared to the neural networks (Tavanaei, Ghodrati, Kheradpisheh, Masquelier, & Maida, 2018).

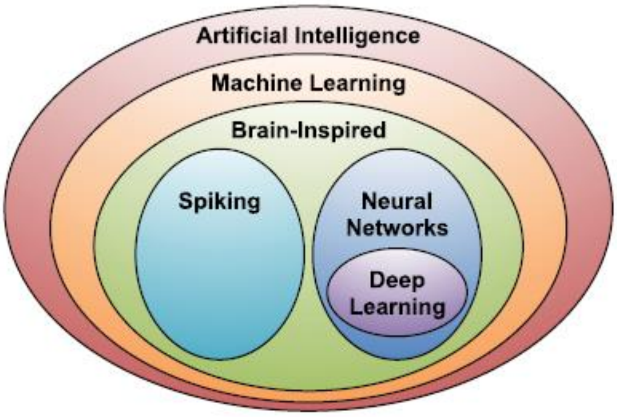


Figure 2.5. The position of several development areas in the context of artificial intelligence

Experience has demonstrated, however, that the mechanisms involved in machine learning are significantly different to the brain's (Baev, 1998). As part of an intelligent machine movement, an article about research on neural networks was published in the early 1940s (Yadav, Yadav, & Kumar, 2015). Other works conducted in the early stages of the intelligent machine movement can be found in (Grimsdale et al., 1959b; Horwitz & Shelton, 1961; Rosenblatt, 1960; Sebestyen, 1961; Taylor, 1959). One of the earliest brain-inspired models implemented in computers is called the perceptron network (Rosenblatt, 1960). In fact, the inspiration for this brain-inspired algorithm came from a behavioural study on how humans learn, and this is presented in the 1949 book "The Organisation of Behaviour" by (Hebb, 2005). The way other creatures (i.e. bees) learn and recognise patterns is also studied in (Ronacher, 1998). In 2005, (Hawkins & Blakeslee, 2005) wrote the book "On Intelligence". This book presents a very interesting theory on how the neocortex of the brain may work. Hawkins' ideas were implemented into the algorithm called hierarchical temporal memory (HTM) in (George, 2008b; George & Hawkins, 2009).

Although HTM is not in the picture drawn by (Sze et al., 2017), the following sections present the artificial neural network, spiking neural network, and hierarchical temporal memory.

2.3.1. Artificial Neural Network

The modern study of artificial neural networks (ANNs) started with a paper that employs Turing's mathematical notion of computation written by McCulloch and Pitts in 1943 (McCulloch & Pitts, 1943). According to (Hebb, 2005), the concept of how humans learn is presented in 1949. A mathematical computing algorithm that replicates the process of a biological neural system is a definition of an ANN (Hen Hu & Hwang, 2001). (Macukow, 2016) states that an ANN is a network of neurons organised in layers. Figure 2.6 shows an illustration of a biological neuron (Yegnanarayana, 1994), while the structure of an ANN with two hidden layers (a multilayer perceptron) is presented in Figure 2.7 (Katic & Vukobratovic, 2003).

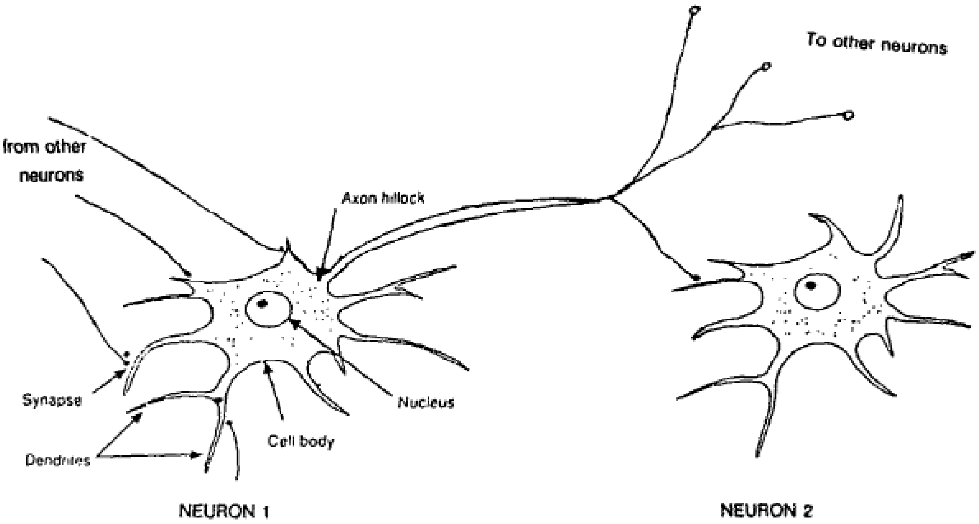


Figure 2.6: Biological neurons (Yegnanarayana, 1994)

The circles in Figure 2.6 represent neurons that are connected to their neighbours, using solid lines, and fulfil the function of nerve fibres (Katic & Vukobratovic, 2003).

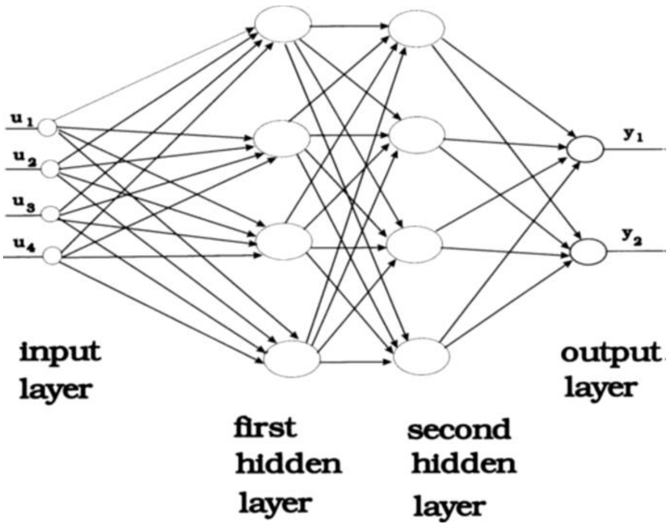


Figure 2.7: Multilayer perceptron network (Katic & Vukobratovic, 2003)

According to (Yagawa & Okuda, 1996), the inputs denoted by u_i flow through the network to produce outputs y_i, applying the state transition t_i into the activation function $f(t_i)$ as:

$$t_i = \sum_{\substack{j \\ i \neq j}} w_{ij} u_j + \theta_i \qquad (1)$$

$$y_i = f(t_i) \qquad (2)$$

Inputs are multiplied by weights w_{ij}, summed up, and then sent to the activation function. The output y_i of unit i is the result of the activation function as presented in equations (1) and (2). θ represents the offset. The activation function $f(t_i)$ is usually the sigmoid function, which has values between 0 and 1. Then y_i will be:

$$f(t_i) = \frac{1}{1+exp^{-t_i}} \qquad (3)$$

An ANN processes inputs or information (images, patterns, objects, etc) in a similar way to the human brain. The simplest ANN has signals that flow from inputs, from hidden nodes that receive the signals from the inputs, and from output nodes that produce the outputs of the network (Macukow, 2016).

The input layer of an ANN is directly connected to the input patterns, and the number of nodes in this layer depends on the number of features (Ukil, 2007). In terms of image recognition, the inputs that are fed to the input layer are the values that represent the colours of pixels as the features of image. In the case of an image of size 32x32 pixels, where each pixel represents one feature, the input layer has 1024 neurons as the ANN's input nodes.

The multilayer neural network is also known as the hierarchical neural network. (Fukushima, Miyake, & Ito, 1983) introduce the concept of a hierarchical network, containing cascaded connections between its layers. (Yeap, Zaky, Tsotsos, & Kwan, 1990) explain that the reason for using a hierarchical network is to reduce the complexity of the network. Because neural networks are inspired by biological neurons, these early researchers implemented components such as cells, synapses, and layers. A few years later, (Pau-Choo, Chen, & Ching-Tsorng, 1994) presented their work on pattern recognition, based on hierarchical neural networks with two layers. They train the two layers to perform specific tasks. The first layer extracts the features of patterns, while the second layer recognises the objects. Following some early success, modifications to the network are conducted to achieve better results. For example, (Ersoy & Shi-Wee, 1995) propose an algorithm to parallelise the hierarchical neural network.

ANNs have been widely applied in many areas, and different topologies have been suggested to achieve better results and performance. For example, the recurrent neural network (RNN), which is a class of neural network with a self-connected hidden layer, produces better results than the hierarchical Markov model (HMM) in handwriting recognition (Graves et al., 2009). The inputs and outputs of early neural networks are independent of one another, whereas the RNN has connections in its hidden layer for the purpose of apprehending information in sequential data (Choi, Kim, & Lee, 2016). With this ability, the RNN can predict what the output will be based on the previous data. RNNs are able to replicate the ability of the brain to conduct prediction. However, RNNs are not appropriate for solving problems involving voice, natural language, and time-series data. RNNs are also part of deep learning models like the enhanced neural network called the convolutional neural network (CNN) (Du, Cai, Wang, & Zhang, 2016).

The CNN is neo-cognition based on the receptive field (the area of input). It is different to the original neural network. CNNs do not propagate each pixel from an image to the hidden layer's nodes through the input layer's nodes as the original neural networks did. CNNs convolve or shift the receptive field across the image area and across the input. Shifting the receptive field means that CNNs look at features of the image within the size of the receptive field and then propagate the output of the receptive field to the hidden layer's nodes. Convolutional feature extraction is conducted hierarchically. For example, an experiment by (Afzal et al., 2015) implemented four convolutional layers. The first layer used a 55x55 receptive field to extract features of an input image of size 227x227 pixels. The convolutional output of the first layer is then pooled to a 27x27 receptive, and filed in the second layer. From Layers 2 to 4, outputs of previous convolutional layers are pooled into 13x13, 6x6, and 1x1 receptive fields in Layers 2, 3, and 4, respectively. Outputs of the last convolutional layer are fed into the input layer's neural network nodes, while the original neural networks are fed directly from the input.

The use of receptive fields to extract features in CNN algorithms looks promising compared to ANN, because the receptive field extracts the shape of images (Afzal et al., 2015; Du et al., 2016; Li, Peng, Zhizhen, & Jingjing, 2016; Pal & Sudeep, 2016). The CNN has been developed further to be Deep CNN (Krizhevsky, Sutskever, & Hinton, 2017). In their work, the proposed network is shown in Figure 2.8.

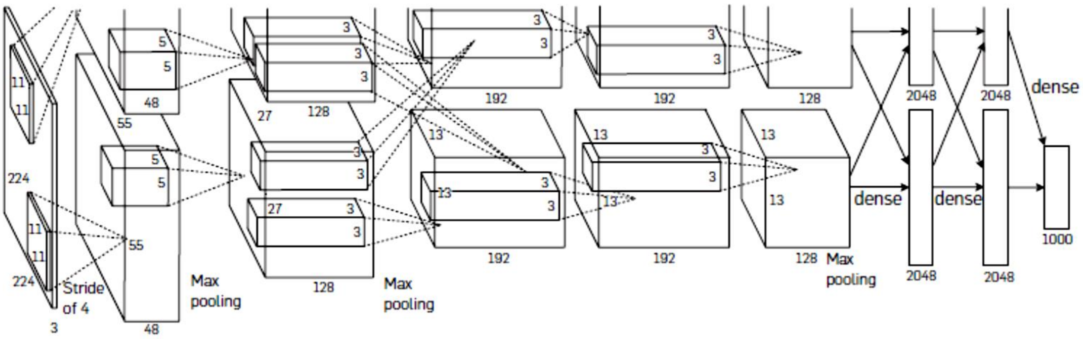


Figure 2.8. The architecture of Deep CNN

This architecture contains eight learning, five convolutional, and three full connected layers. This architecture was designed with unusual features. The first is the Rectified Linear Unit (ReLU) which refers to the neuron with non-linearity. The use of CNN with ReLU increases the training time faster than the standard CNN. The second is the use of multiple GPUs that consumes less time compared with only one GPU. The third is Local Response Normalization which is applied after ReLu. They claimed that with the error rate for dataset CIFAR-10 was 13% without normalization and 11% with normalization. The last feature is Overlapping Pooling to reduce the overfitting.

To simplify the training process on deeper neural networks, (He et al., 2016) proposed a new architecture for deep CNN using deep residual learning framework which is then call as ResNets. This new algorithm has been developed due to the degradation of accuracy when the network's depth increase. To solve this problem, they used the residual formulation that is added by shortcut connection as shown in Figure 2.9 .

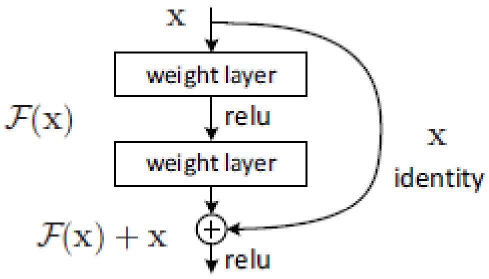


Figure 2.9. A building block of residual learning

ResNets gave good performance compared to the original deep CNN at the ILSVRC 2015 in classification, localization, and detection, and COCO detection and segmentation. Even though residual networks can be used up to thousands of layers or very depth network, it has a problem of diminishing feature reuse that makes the time to train the network is very slow. (Zagoruyko & Komodakis, 2016) proposed a method to solve this problem by making the residual network wider instead of deeper. They claimed that their simple 16 layers wide residual network produce better accuracy and efficiency compared with all previous deep residual networks with thousands of layers. Recent investigation by (K. Zhang et al., 2018) found that the wider residual network in (Zagoruyko & Komodakis, 2016) still having overfitting problems when the network is becoming wider. This is due to the ResNets perform residual mapping fitted by stacked nonlinear layers. They then gave the hypothesis that the residual mapping of residual mapping is easier to optimize than the original residual mapping. They then proposed their hypothesis which is then called residual networks of residual networks (RoR). The idea of RoR was based on the hypothesis to optimize the residual networks.

The exploration of the neural network architectures has been an interesting research area since their performance become well. Residual networks is a part of research area of neural network architectures since the problem emerged when the CNN become deeper. (Huang et al., 2016) argued that the CNNs can be more accurate and efficient to train compared to the original CNN if the connection between layers becomes shorter. If the original CNN with L layers have L connections, their network called Dense Convolutional Network (DensNet) has $\frac{L(L+1)}{2}$ direct connections. Figure 2.10 shows the DensNet in (Huang et al., 2016) whereas the network has five layers and fifteen connections between layers.

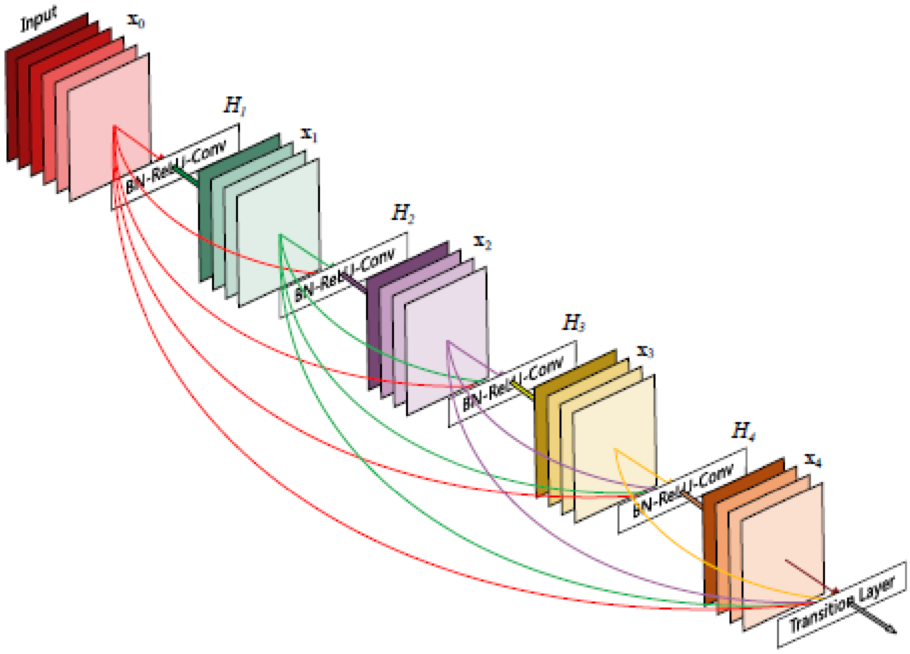


Figure 2.10. DensNet with 15 direct connections

The connections between layers is supposed to ensure the maximum information flow between layers in the network. If the ResNets sum up the features to be passed into the next layer, the DensNet combines features by concatenating them. With this different feature combination method lead to different behaviour between these two network architectures. Regarding the network parameters, by using the same dataset, a DensNet requires only 1/3 of the parameters of ResNet and able to achieve the comparable accuracy (Huang et al., 2016).

The implementation of DenseNet has been used in several experiments. (Rubin, Parvaneh, Rahman, Conroy, & Babaeizadeh, 2017) implemented the DenseNet to analysed the signal quality atrial fibrillation using short single-lead ECG recording, (Sreela & Idicula, 2017) modified the DenseNet for content generation in automatic image description. (Kuang, Ma, Chen, & Li, 2018) utilized the design of DenseNet to be able to learn mapping between low and high resolution images. Even the combination between ResNet and DenseNet called Densely Connected Residual Networks (DRNet) has been investigated very deep and wide network with only few parameters (Wen et al., 2018).

2.3.2. Spiking Neural Networks

According to (Tavanaei et al., 2018), the neural networks with activation values and a set of inputs' weight are inspired by biological component called neuron. These kind of neural networks are categorized as a non-spiking neural network. The more realistic biological inspiration of neuron uses discrete spike to compute and transmit the information. If the non-spiking neural networks use differentiable activation function, the spiking neural networks (SNNs) use non-differentiable activation function. (Tavanaei et al., 2018) claimed that SNNs are still behind ANNs in terms of accuracy, but the gap between them is decreasing.

The SNNs are the third generation of neural networks (Xie, Qu, Liu, Zhang, & Kurths, 2016) which are well known in the cognitive tasks. The first generation of neural networks was indicated by the use of neurons and introduced by McCulloch-Pitts model (McCulloch & Pitts, 1990). The second generations which use perceptron were pioneered in (Rosenblatt, 1958). The spiking concepts has been discussed in (VanRullen, Guyonneau, & Thorpe, 2005) with the main purposes to replicate the fast computation like the brain does. The simulations of firing process in SNNs are shown in (Mehta, Lee, & Wilson, 2002) and (Benchenane et al., 2010). The main goal of research in SNNs area is the learning process of how to recognize the information of spike trains (H. H. Amin & Fujii, 2005). It is based on the spiking neural network model in (H. Amin & Fujii, 2004) which called as spiking response model (SRM). Figure 2.11 shows the basic idea of their network with single neuron (H. Amin & Fujii, 2004). It looks like earlier neural network.

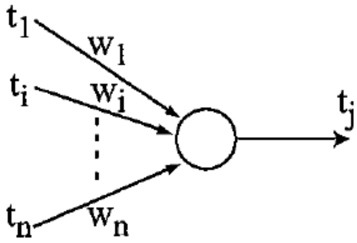


Figure 2.11. A neuron with n inputs and j outpus

The input spikes come to the input synapses of the neuron. It then generates a spike when the internal neuron membranes crosses the threshold that is assumed to be constant. It is shown in Figure 2.12.

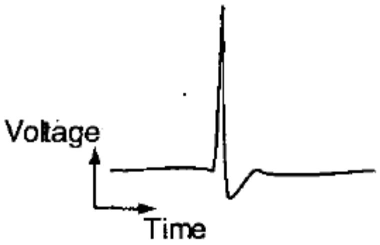


Figure 2.12. A potential spike

According to (H. H. Amin & Fujii, 2005), the learning process in SNNs has two stages. The first is a mapping stage that is composed of neural mapping units (MUs). This stage is used for mapping the input spike trains into unique spatio-temporal patterns. Their mapping stage is illustrated in Figure 2.13.

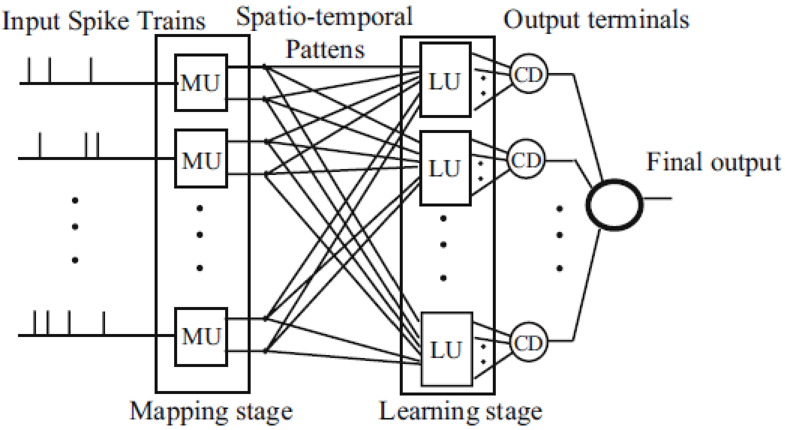


Figure 2.13. Mapping input spike trains

The second stage is the learning stage. This stage consists of several learning units (LUs) as shown in Figure 2.14 (H. H. Amin & Fujii, 2005).

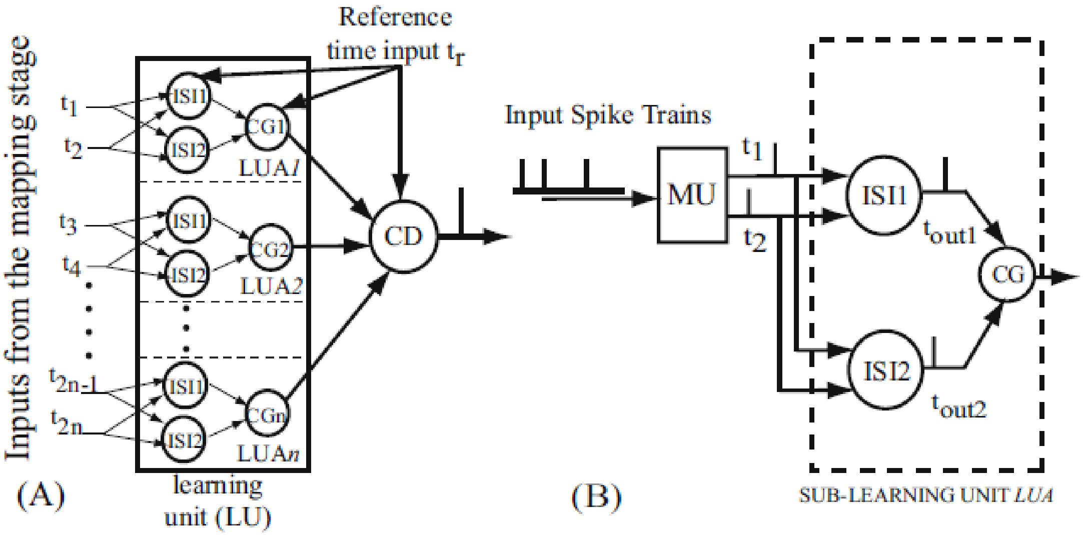


Figure 2.14. (A) Learning units, (B) MU and sub learning unit

The input of learning stage is the output of mapping stage that is the spatio-temporal output pattern. Each LU is formed by sub-LUs as shown in Figure 2.14 (A). Each sub-LU receives input from one mapping unit as shown in Figure 2.14 (B). The inter-spike intervals (ISIs) unit that contains a sequence of spike in LU and sub-LUs have been assigned with synaptic weights. In each Lu there are *2*n* of ISIs, where *n* is the number of spike trains. The output of ISI1 and ISI2 in each LU (see Figure 2.14(A)) fires at a certain times defends on the assigned synaptic weights. Patterns that are close to the train pattern will join the same LU and can cause the output fire. For that reason, the neuron threshold should be adjusted to allow some fuzziness in the input spike times.

SNNs plays important part to realize the model of the brain by modelling the memory mechanism (Lisitsa & Zhilenkov, 2017). They also stated that the use of SNNs are still not wide due to the lack of understanding of the important algorithm of the brain. However, the research in this area is growing and has been implemented in the hardware platform.

2.3.3. Hierarchical Temporal Memory

Hierarchical temporal memory (HTM) is a biological-inspired algorithm discussed in (George, 2008b; George & Hawkins, 2005a; George & Hawkins, 2009). It is originally proposed in the book "On Intelligence" written by Jeff Hawkins (Hawkins & Blakeslee, 2005). To implement the ideas put forward in On Intelligence, Hawkins created a private company called Numenta. Numenta researchers conducted a study to develop and implement the proposed concept. Their works are available on www.numenta.org.

Hawkins mentioned that in order to create intelligent machines it is necessary to understand intelligence and what may be happening in the brain to create it. Like other researchers before him, Hawkins studied how the brain cortex works to try to replicate the emergence of intelligence. Figure 2.7 shows a simplified model of four visual cortex regions labelled as V1, V2, V4, and IT. The V1 region receives the input from the retina. In the HTM, the input that comes to region V1 is processed in a similar way to how neurons are processed in the eyes' receptive field. In this region, a spatial-pooling function is performed on the input; in other words, the input is scanned and extracted to get its features. This is similar to what happens when the eyes look at an object and see every single detail of every feature within the object. Then, when the eyes move to look at another detail, it makes groups of similar patterns that have been seen before. The movement of the eyes is the biological equivalent to moving the receptive field. The groups of similar patterns are useful in firing the neurons in region V2. The process from the bottom region (V1) to the top region (IT) is called feed-forward. This process is indicated by the arrow pointing up in Figure 2.15, while the arrows pointing down represent the feedback process from region IT to V1.

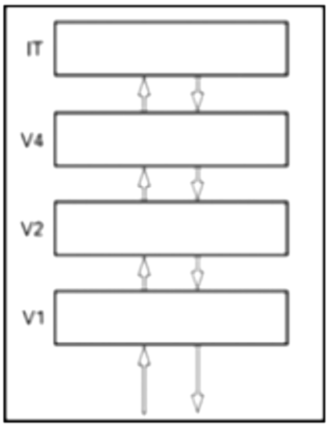


Figure 2.15: Visual regions of cortex for pattern recognition (Hawkins and Blakeslee 2005)

In his book, Hawkins presents his theory of how intelligence is produced in a part of the brain called neocortex. To understand how the brain works, the brain process can be replicated by a computer algorithm (George, 2008a); however, this is not to replicate the real biological brain as a machine algorithm. The current computational capability of machines must also be considered.

The machine has some requirements for replicating the biological process into a computer algorithm (Hawkins, George, & Niemasik, 2009). The first requirement is called probabilistic prediction and is designed to predict future events from noisy input data. The second requirement is for simultaneous learning and recall, meaning that the machine should have the ability to learn and predict simultaneously. Intelligent machines are also supposed to be able to recognise, even if it gets only a partial part of an object. This is called auto-associative recall. Variable-order memory is the next requirement, where a computer algorithm should be able to predict the next input. The last requirement for the machine is that the intelligent system should be mapped to the neocortical anatomy.

The HTM algorithm is inspired by the brain's neo-cortex, hence it is called a cortical algorithm. Figure 2.15 shows a simplified interpretation of the hierarchical structure of the neo-cortex.

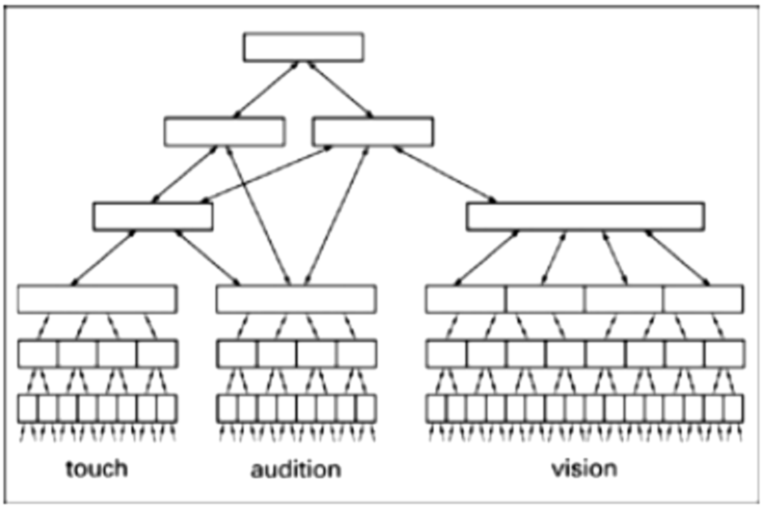


Figure 2.16: The hierarchical structure of the neo-cortex (Hawkins and Blakeslee 2005)

Figure 2.16 shows three kinds of sensorial systems that perceive information from outside of the human body. The skin constitutes the touch sensory system, which receives tactaile information by touching. Sound stimuli are perceived through the ears in the auditory system, while the eyes receive visual information for the vision sensory system. The information that is captured through through these senses is sent to the first region of the cortical sensory system. The top level of the network makes sense of the world based on the given information from all three sensors.

The actual process that takes place in the brain to make sense of the world is very complicated. To this day, the concept of combining different sensory systems is still in its early research stages. Researchers are more focussed on the development of algorithms to emulate each individual sensory system. The research is then continued in the area of pattern recognition or image processing.

According to (Hawkins et al., 2009), the HTM models the neo-cortex as a tree-shaped hierarchy of memory regions, or layers, where each layer learns common sequences of patterns. The HTM also performs a probabilistic prediction function modelled as a form of Bayesian network.

Similar to the neocortex, HTM inputs are connected directly to nodes in the lowest level of the network. The first layer has enough nodes to receive the specific size of the input (based on the size of receptive field). Layers 2 and 3 are layers on the top of

Layer 1 and 2, respectively. These layers receive information from several concatenated nodes of the layers below them. The top layer has only one node. It is in the nodes of the top layers where the output decision is made. A schematic representation of an HTM network is shown in Figure 2.17.

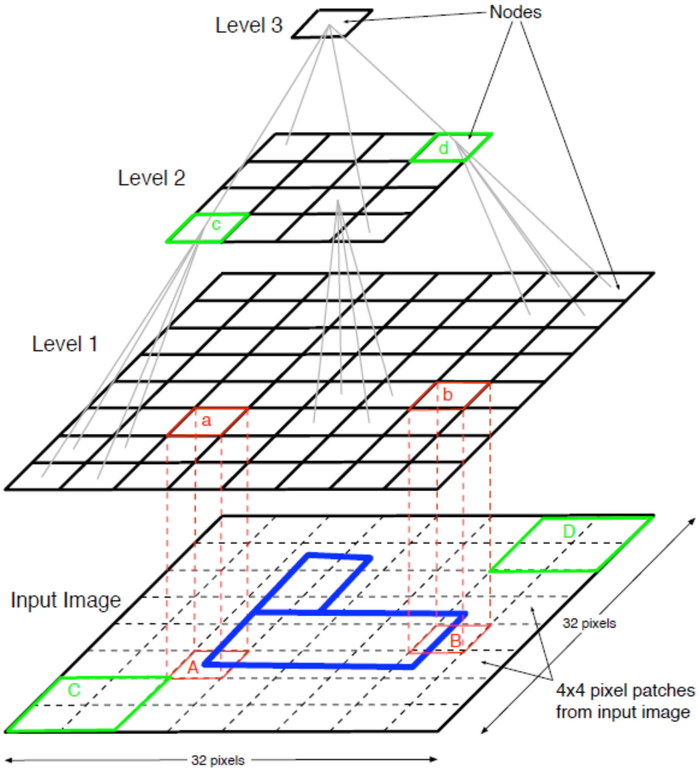


Figure 2.17: HTM network structure (George 2008)

Detailed below are the system requirements — outlined by (Hawkins et al., 2009) — for implementation of the biological concept into a computer algorithm:

a. Sparsification of response. The system should be sparser and more selective to the incoming input.

b. Inhibitory requirements. This requires only a few cells to be active during a feed-forward input.

c. Distributed representations. Every region of the hierarchy passes a distribution of potentially active sequences to its parent regions.

d. Efficient computation. The memory system should use information from previous inputs when making prediction. Both history of inputs and forward predictions are distributed over many states.

e. Cortical layers. Hierarchical models require the feed-forward and feedback path. The feed-forward happens in Layers 3 and 4, and the feedback happens in Layers 2 and 6, while a belief is made in Layer 5.
f. Sequence timing. The sequence memory needs a neural mechanism that can encode the duration of sequence elements in all regions.
g. Memory capacity. This is related to memory allocation. The use of hierarchical sequences of learning should be used repeatedly in different combinations. For example, when a series of small features is stored in the lower layer, it forms bigger features in the upper layers.

2.3.3.1. The Flow of Information in HTM Networks

The biology of the human brain has inspired the implementation of hierarchical layers in HTMs to extract features for visual pattern recognition. The lower layer of HTM networks learns the basic features of an object, which are then used to generate new, more complex features in the higher levels. To build the new features in the higher layers, a set of lower layer features must be concatenated. This process continues until the information from the lowest layer reaches the top layer. According to Hawkin's model, this is the way information flows in the human brain.

As a soft computing framework, which is derived from a simplified model of the human brain, the HTM is considered by some as a new masterpiece in the field of artificial intelligence (Chen et al., 2012). The key process of HTMs is the feature extraction of every single part of the object — spatially, temporally, and hierarchically. Spatial extraction is the process of scanning the input of the network, while temporal extraction groups the most likely patterns into the same group. The hierarchy concept of HTM shows the relationship between the parent and child layer (high level and low level) as shown in Figure 2.17.

Every single part of the visual pattern that comes to the first layer of the network is patched together using a receptive field or coincidence array of size 2x2 pixels. This is called the spatial-pooler, which means that within a time t, a coincidence array will record four bits from the whole pattern. To analyse the whole input image, the coincidence array moves across the pattern. After several times t, the spatial-pooler groups several four bit patterns based on similarity and using a Markov chain. The process of grouping patterns is called temporal-pooling.

The information that is passed to the next layer depends on the degree of belief of the layer below it. Beliefs from child nodes are propagated to their parent nodes. Parent nodes will also propagate their beliefs back to their child nodes to confirm or adjust the child's belief. These processes are unsupervised. HTM learns from time to time, and time itself acts as a supervisor (Hawkins et al., 2009).

2.3.3.2. Spatial and Temporal Pooler Related Works

The concept of temporal and sequential pooling has been studied by many researchers because it is an important component of understanding human intelligence. (Starzyk & Haibo, 2007, 2009) study this biological concept. Even though their works are a bit different — compared to the original HTM as described in (George, 2008a) — in terms of the connection between each neuron and its process, both studies are based on the two types of memory identified in neurobiological research of the human brain, namely: short-term memory (STM) and long-term memory (LTM). In the research, the lower layer, called the pattern recognition layer, uses a modified Hebbian learning algorithm while the old HTM uses Bayesian networks. The WTA (winner takes all) approach is used to decide which stored pattern is similar to the incoming pattern. This WTA is implemented in all layers to reduce the complexity of the learning process. In their simulation, the weights of all the neurons' were set to be a small positive number between 0.001 and 0.01). The WTA itself is the maximum value of the sum of the input's products from the lower layer. It is similar to the general concept of HTM that also uses the WTA to define which stored pattern has the maximum probability of being similar to the incoming pattern. However, they use a threshold to anticipate the possibility of getting more than one WTA. The concept of spatial-temporal and propagation processes in (Starzyk & Haibo, 2007, 2009)'s works are more complex compared to the spatial-temporal concept explained in (Chen et al., 2012), but the concept of using STM and LTM can be considered for use in the HTM algorithm, as long as it meets the neocortex model.

Other studies that replicate the neocortex also use spatio-temporal processes. (Vu-Anh, Starzyk, & Wooi-Boon, 2012) claim that there are three critical problems in spatio-temporal learning: error tolerance, significance of a sequence’s elements, and the memory forgetting mechanism. Solving these problems allows the system to process real-valued, multi-dimensional data streams, continuously. Spatio-temporal neural architectures have two types of memory: STM (short-term memory) and LTM (long-term memory). STM is used as temporal storage of input data (it will decay over time) and has a limited capacity. LTM is built using synaptic modifications of STM. Also, LTM involves several STMs.

2.3.3.3. HTM for Pattern Recognition

The neocortex-inspired HTM has played an important role in the development of algorithms for object or pattern recognition. Many methods have been found to replicate the neocortex’s work as a computer algorithm. However, even though this research area has been studied by various researchers over many years, it is still far removed from achieving the level of the natural brain’s performance (Lei, Xianbin, Xu, & Jianguang, 2009). (Hawkins & Blakeslee, 2005)’s basic understanding of the human brain is used by (Lei et al., 2009) to implement object recognition using HTMs. They use handwritten digits, collected from mail envelopes in Buffalo. Each digit is captured in 16x16 pixels. Euclidean distance is used to measure the similarity, and their results show that a small distance threshold produces more accurate outputs than a longer distance threshold. The researchers also provide the results when using other tools, such as relevance vector machines, neural networks, invariant support vectors, a k-nearest neighbour classifiers (k-NNs), mixture densities, and support vector machines (SVMs) (Lei et al., 2009). However, they do not mention whether the HTM performs well compared to others. The work by (Maltoni, 2011), on the other hand, does provide comparisons between the HTM, nearest neighbour classifiers, multilayer perceptrons, and convolutional neural networks. (Maltoni, 2011) is also successful in implementing the HTM for pattern recognition. His work is available for viewing on NuPIC (Numenta Platform for Intelligence Computing), which is an open source project founded by Hawkins. Maltoni uses three different pattern sets, namely: SDIGIT (machine-printed digit with 16x16 pixels and greyscale images), PICTURE (line drawing in 32x32 pixels with black and white images), and USPS (handwritten in 16x16 pixels with greyscale images) as described in (Ponce et al., 2006). Some parameter values, such as the forget threshold, transition memory, minimum and maximum group sizes among others, have been investigated for each pattern classification to get the best result. The results of Maltoni’s work show that HTM is much more accurate than the three different pattern classification methods, namely SDIGIT, PICTURE, and USPS, and also when compared to NN, MLP, and CN. However, is it shown that the HTM consumes more time to finish the process.

The implementation of HTM in pattern recognition is not only to test whether the HTM performs well compared to other methods, but also to investigate HTM optimisation. The optimisation of HTMs is also investigated in (Bolotova & Spitsyn, 2012). They optimise the temporal grouping process and rename it maximum temporal connection (MTC). This optimisation method is based on the greedy algorithm (GrA). MTC’s recognition rate is 78.6% with clustering and 83.1% without. By using MTC, the recognition rate increased by 3-5%. The temporal grouping process is also optimised using an adaptive neural gas algorithm and a graph clustering technique, as reported in (Charalampous, Kostavelis, Amanatiadis, & Gasteratos, 2012). This research uses colour images from the ETH-80 dataset. This dataset is a collection of 256x256 pixel coloured images. The k-NN and the linear SVM method are used in the top level of the HTM, and the results are compared with those of the proposed method, which is a combination of the SVM method and the principal component analysis (PCA). This optimisation also performs better than k-NN and pure PCA.

In another work, (Kostavelis et al., 2012) also use colour images from the ETH-80 data set as inputs to the HTM. This work adopts saliency detection to release the HTM network from memorising redundant information and to increase the accuracy of classification. The HTM implements the saliency detection procedure using a graph-based visual saliency algorithm (GBVS) before performing the spatial process. This prevents the HTM from memorising redundant information, and means that the HTM is not trained on the whole image, but on just a specific area of the image delivered by the GBVS. The classification rate shows that this method gives a good result, even though it is only about 2% higher compared to k-NN and SVM. In other works, the classification rate of optimised HTMs — using locality-constrained linear coding (LLC) and spatial pyramid matching (SPM) — also increases the classification rate by about 2%–3% compared to the original LLC and SPM models.

(Boone et al., 2010) show that HTM is not as accurate as traditional image analysis and the processing techniques used to diagnose the retina in the detection of diabetes. Their work helps people prevent diabetes by monitoring the eyes of patients. However, one particular implementation called NuPIC is shown to still be limited in its recognition of colour images, and it cannot identify any other objects included in the single image. Nerve diagnosis in the retina not only requires detection of the shape of the retina, but also recognition of the nerve in a retina image. The image of the retina is also influenced by the intensity of light. To improve results, therefore, the image is pre-processed by dividing the single image into smaller greyscale images, and the location of the nerve is manually determined using Matlab. The smaller images are then fed into the HTM input layer. Even though the results show that HTM is able to recognise the nerve in the images, the researchers claim that this case does not show the full benefit of using HTM, which is identified as learning effectively through example.

Many applications implementing HTM networks focus on pattern or object recognition. (Schey, 2008) uses NuPIC to identify song, which is a time series problem. Even though NuPIC is not able to handle the complex problem of song recognition, the data preparation and research methods are good examples of ways to start understanding the kind of data that can be fed into HTMs in future research. In his description of the process to be followed, Schey suggests that readers must first understand the input data to be analysed. In his research, it is about understanding and investigating the digital music format. The second step is preparing the data that will be fed into HTM network and writing the data generation scripts. This is an important step because the input data should be in a format that can be fed into and read by the HTM network. The next step is creating and configuring the HTM network. The standard version of NuPIC does not suit all applications; it requires customisation. The last step is training to see — and then evaluate — the accuracy of the network. If the accuracy is not acceptable, the parameters assigned to each node must be adjusted.

Figure 2.18 shows the example of input that is fed to HTM network in Schey's work.

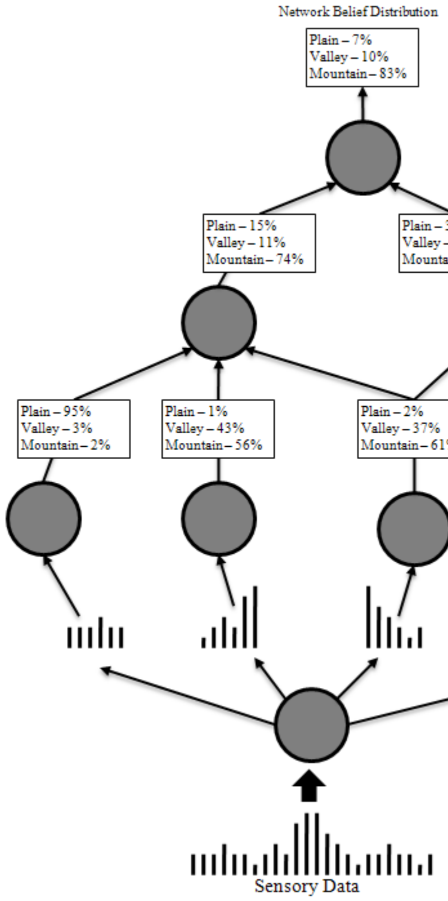


Figure 2.18: The example of input patterns fed to HTM network (Schey 2008)

Another work, this time by (Maxwell, Pasquier, & Eigenfeldt, 2009), adapts the HTM into their proposed framework called hierarchical sequential memory for music (HSMM). They observe that music has the potential to build spatio-temporal hierarchies. They start their work by dividing standard MIDI files into three groups of data, namely: pitch data, rhythmic data, and velocity data. Figure 2.19 shows a HSMM with a four-level network.

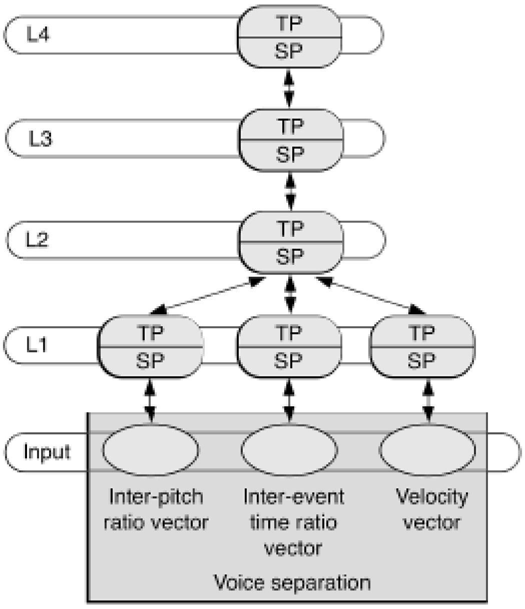


Figure 2.19: Splitting MIDI files to be fed into HTM networks (Maxwell et al., 2009)

The inputs for pitch, rhythm, and velocity are associated with node on L1 (Level/Layer 1) and combined on L2 (Level/Layer 2). The upper layers, L3 and L4, learn the higher-order musical structure. They use Euclidean distance to evaluate the similarity between incoming input patterns and the stored patterns. They claim that the system is able to recognise a simple pattern at L1, and at the higher level, it can recognise larger musical structure like phrases, melodies, and sections.

Another example of the use of HTMs in real time series problems is implemented in (Gabrielsson, Konig, & Johansson, 2012). Their work focuses on creating a profitable software agent to trade in the financial market. They use data from the E-Mini S&P 500 (ES) of 2 August–1 September, 2011. The raw data is aggregated into vectors containing one minute of the trade volume. Each vector contains 10 technical indicators or features, which are fed into HTM networks. The first 70% of data is used for the training, the next 15% is used for validating, and the last 15% is used for testing.

Figure 2.20 shows the use of HTM for classification, based on (Gabrielsson et al., 2012).

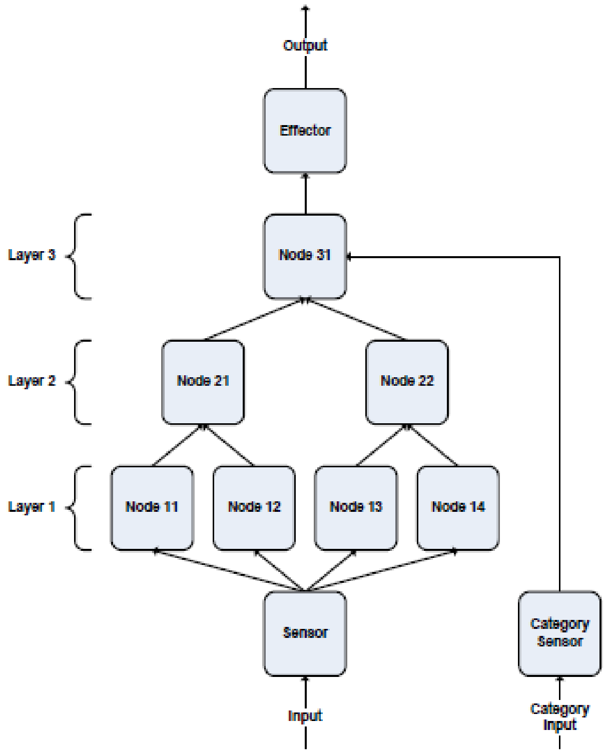


Figure 2.20: HTM network as a classifier of two classes 0 and 1 (Gabrielsson et al., 2012)

In this research, the classifier has two output classes: class 0 indicates the price will not increase by at least two ticks at the end of the next ten-minute period, and class 1 represents that the price will increase. During the training stage, input vectors are fed into Layer 1 of the network; the network is then trained with the classifier, which is fed into Layer 3. During temporal pooling, likelihood is calculated using a Euclidean distance algorithm with a varied distance from 0.001 to 0.5. An ANN is used to compare the result of the proposed HTM network. The study claims it is possible to use HTM to create a profitable trading algorithm for the financial market.

2.4. The General Process of Image Recognition

The image recognition process consists of the following steps (Gurevich & Koryabkina, 2006):

1. Obtaining the original image by scanning, photography, capturing from sensors, etc.
2. Pre-processing the image, which involves:

    a. image normalisation, intensity normalisation, filtering, binarisation, colour conversion, noise elimination, etc.
    b. image segmentation to distinguish objects and regions
3. Image feature extraction
4. Recognition algorithms

A general recognition or classification system is shown in Figure 2.21.

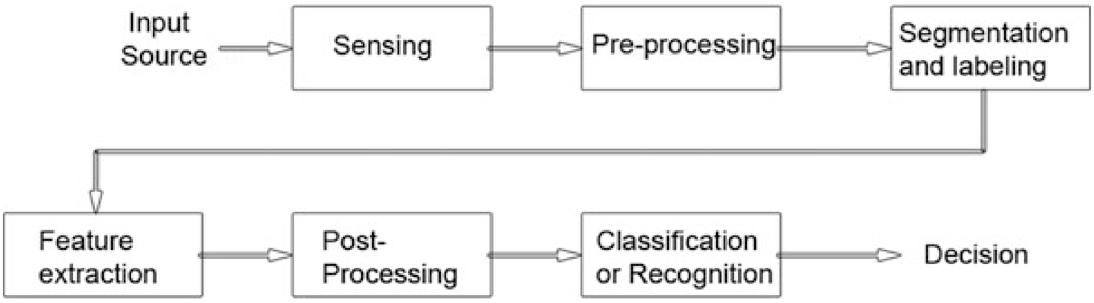


Figure 2.21: A general recognition or classification system (Dougherty, 2013)

2.4.1. Feature Extraction

According to (Tou, 1968), there are four different ways to describe the features of an object, using: physical features, topological features, mathematical features, and statistical features. When humans see objects, they might focus on particular features of an object (e.g. the physical or topological). When observing an apple and an orange, humans can recognise them based on their physical and topological features, but these two ways of feature extraction are in fact arbitrary and subjective.

Feature extraction, on the other hand, is a critical process. It requires choosing features of the image to assist in solving recognition and classification problems. The features of an object or image should be informative. They should contain some significant information to distinguish between classes and to ensure that the recognition and classification processes are conducted properly. Features should be suitable for feeding into the image recognition or classification processes, and for allowing the construction of an image model. They must also belong to the minimal set of the image, be limited in number, and not exceed the gain due to their usage in the solution (Gurevich & Koryabkina, 2006; Kumar & Bhatia, 2014; Nixon & Aguado, 2012c).

(Gurevich & Koryabkina, 2006) divide feature extraction into three types based on the type of image features: binary, greyscale, or colour. (Nixon & Aguado, 2012a, 2012b) separate feature extraction into two levels: low-level and high-level. Low-level feature extraction is the process of extracting the features of an image without spatial relationship information (no shape information). High-level feature extraction extracts the features of an image by extracting the information of shapes. Low-level feature extraction involves edge detection, phase congruency, and localised feature extraction, while the high-level feature extraction involves template matching, feature extraction by low-level features (appearance-based and distribution-based), and the Hough transform. (Kumar & Bhatia, 2014) list the following widely used feature extraction methods: template matching, unitary image transforms, graph description, projection histograms, contour profiles, zoning, geometric moment invariants, Zernike moments, Spline curve approximation, Fourier descriptors, the gradient feature, and the Gabor feature.

Low-level feature extraction methods analyse parts of an image without using shape information. These methods use pixels as the features of the image. Extracting pixels as the features of the image can be used in any type of image (binary, greyscale, or colour). While the high-level feature extraction examines features based on shape information, it first performs a low-level feature extraction before looking at the shape-based features. This subsection presents some feature extraction methods such as template matching and unitary transform.

Template matching is a technique or method used to compare small parts of an image against a defined template (Cha, 2000; Jalil, Basari, Salam, Ibrahim, & Norasikin, 2015; Song, Chen, Chi, Qiu, & Wang, 2007). In general, this technique measures the similarity between tested images and a predefined template. According to (H. Lee, Kwon, Robinson, & Nothwang, 2016; Nezhinsky & Verbeek, 2012; Slot & Gozdzik, 2008; Takahashi, Tanaka, Suzuki, Shio, & Ohtsuka, 2007; Younes, 2010; Zhu, Chen, & Yuille, 2010), the majority of the experimental setups convert images into greyscale or binary images before processing them using the template matching method.

Unitary image transformations are commonly used for bit reduction in signal transmission, but they also can be used in pattern classification, speech and signal processing, and picture encoding (Kekre & Solanki, 1978). Unitary image transformations preserve the information of the data by concentrating most of the image energy into a few of the transformed domain samples. In other words, the images are processed within blocks instead of as a whole image. A transformed image $\breve{A}$ is a multiplication of the unitary matrix U, the original matrix A, and the transposed form of matrix U (Gotze & Sauer, 1993). Discrete sine transform (DST) and discrete cosine transform (DCT) are two examples of sinusoidal image transforms (Clarke, 1983; A. K. Jain, 1979). DCT is used as a features matrix by (Ezoji & Faez, 2011) to reduce the effect of illumination on the images. In this work, an illuminated image, containing dark and bright pixels, provides histogram equalisation before it is applied to the DCT. The output of DCT is a feature matrix that contains the extracted features from the input image.

2.4.2. Similarity Measurement

To achieve recognition, similarity between two images or patterns must be measured. According to (R. Jain, Murthy, Chen, & Chatterjee, 1995), similarity measurement can be categorised into three groups: metric, set-theoretic, and signal-detection theory-based measurement.

The metric-based measurement calculates a pair of points between the tested and reference images. The distance between them represents their dissimilarity. Euclidean distance has been popular in metric-based similarity comparison. The farther the distance between the reference and tested patterns, the more dissimilarity there is. Let x and y be two vectors of size n, where $x = (x^1,x^2,\ldots,x^n)$ and $y = (y^1,y^2,\ldots y^n)$. The Euclidean distance between x and y is given by:

$$d^2(x,y) = \sum_{i=1}^{n}\left(x^i - y^i\right)^2 \tag{5}$$

Euclidean distance has also been used in fuzzy similarity measurement for calculating the distance of all fuzzy sets between two compared images (Ray, 2009).

Even though Euclidean distance has been widely used to measure similarity, it is very sensitive to even small changes or deformations (Liwei, Yan, & Jufu, 2005). LIwei and Yan propose a new method of Euclidean distance called image Euclidean distance (IMED). IMED has some advantages, such as its relative insensitivity to the small deformation, its simple computation process, and the fact it can be embedded in most image recognition techniques. To measure the distance between two images, metric coefficients are calculated with Equation (6).

$$g_{ij} = < e_i, e_j > = \sqrt{< e_i, e_i >}\sqrt{< e_j, e_j >}\,.cos\theta_{ij} \tag{6}$$

where < and > are the scalar product, and θ_{ij} is the angle between e_i and e_j. The IMED between two images is calculated with Equation (7).

$$d^2(x,y) = \sum_{i,j=1}^{mn} g_{ij}\left(x^i - y^i\right)\left(x^j - y^j\right) = (x-y)^T G(x-y) \tag{7}$$

If the vectors have different lengths, *G* is a diagonal matrix; otherwise, G is the identity matrix.

Because the perceptual assessment of a similarity measure has always been the goal for related research areas, (Chou & Hsu, 2011; Wang, Maldonado, & Silwal, 2011) argue that traditional metric-based measures — like peak signal to noise ratio (PSNR) and mean square error (MSE) — lead to inconsistent similarity evaluations and do not reflect human judgement. They investigate a group of similarity measurements to find the difference of luminance, contrast, and structure or geometry, and (Chou & Hsu, 2011) investigate the metric-based distance by combining several measurements. Their measurement uses the mean of intensity of luminance, luminance contrast, and geometrical distribution; while (Wang et al., 2011) evaluate the quality of similarity measures using a structural similarity measure (SSIM). The SSIM index is the similarity product of the luminance, contrast, and structures.

In (Kobayashi, 2016), measurement of similarity is investigated based on three groups of similarity functions involving the mean, the standard deviation, and the correlation of each pair of images. Another metric for similarity is investigated in (Xiaohang & Dianhui, 2004). They calculate the similarity between pairs of images based on a content retrieval technique. This technique does not compare pixel by pixel to determine similarity; instead, it compares region by region. The regions are divided into five: the top-left, top-right, bottom-left, bottom-right, and the centre.

Set-theoretic based measurement compares sets of features between a pair of images or objects (Tversky, 1977; Tversky & Gati, 1978). Tversky proposes the similarity between two features, A and B, is as illustrated in Figure 2.22.

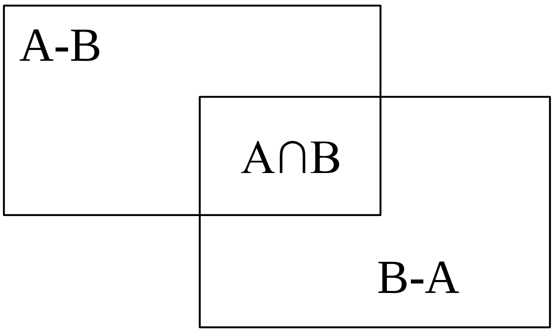


Figure 2.22: An illustration of the relation between two feature sets

By using Tversky's linear similarity model, the similarity *(s)* between objects a and b can be defined as in (Cazzanti & Gupta, 2006):

$$s(a,b) = \emptyset f(a \cap b) {-}{\propto} f(a \backslash b) - \beta f(b \backslash a) \tag{8}$$

where f is a positive saliency function, and θ, α, and β are fixed positive real numbers. Objects a and b are more similar if their intersection is bigger. Improved set-theoretic similarity measurement has been investigated in several works, such as the comparison of fuzzy sets in (F. Zhao & Ma, 2006), the matching of shapes in (Hasanbelliu, Giraldo, & Príncipe, 2014), and the similarity measurement between songs in (Foster, Dixon, & Klapuri, 2015).

Signal-detection theory points out the similarity between objects or patterns based on three different types of task: yes/no tasks, rating tasks, and forced-choice tasks (Stanislaw & Todorov, 1999). A yes/no task gives a 'yes' response if the decision variable is sufficiently high enough, based on the given criterions, and a 'no' response if no signal is presented. (Stanislaw & Todorov, 1999) illustrate the decision distribution as shown in Figure 2.23.

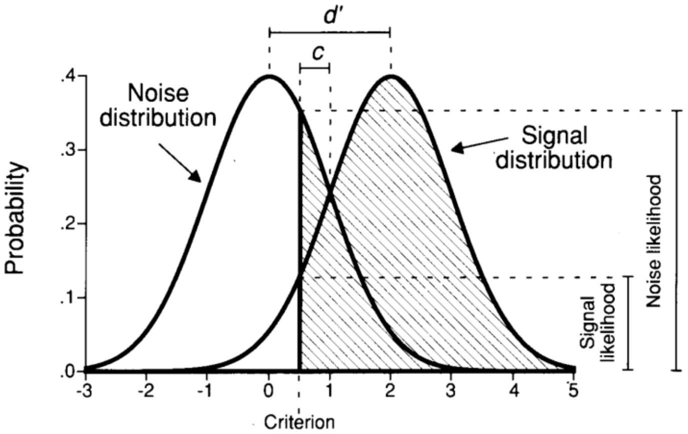


Figure 2.23: The distribution of decision for a 'yes/no' task

A 'yes' response is illustrated by the highlighted signal distribution, while a 'no' response is shown by the noise distribution.

2.4.3. Classification Methods

Classification is the process of allocating a pattern or object to a specific class or group based on the similarity of their attributes. (Costa & Cesar, 2000) state that classification is a general, broad, and not completely developed area. However, classification has played an important role in many areas, from biology to human sciences, and anyone interested in pattern classification should be prepared to accept a broad variety of perspectives.

(Costa & Cesar, 2000) define the basic concepts of classification in four ways as follows:

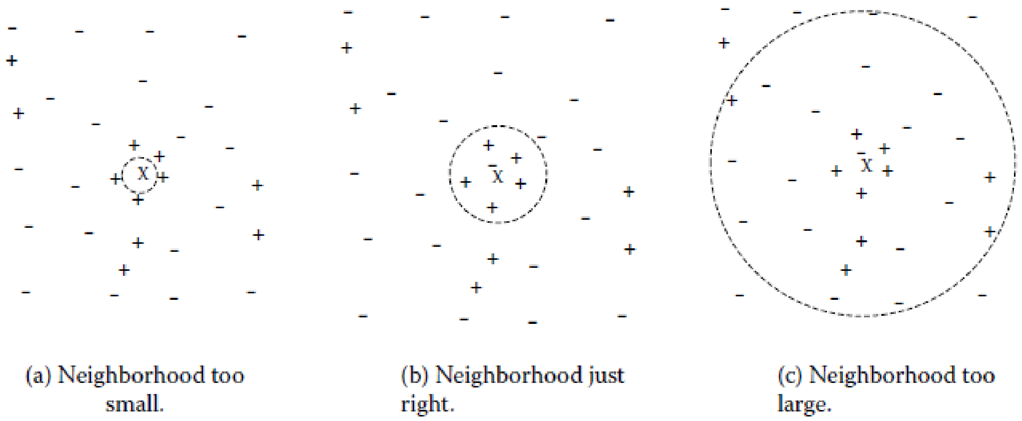
- Conducting a hierarchical classification that makes a class with several sub-classes, each of which inherit the properties of the respective superclass.
- Removing redundancy to anticipate the repetition of memorising or conducting the same tasks.
- The act of assigning classes to objects, which can be supervised or unsupervised.
- Organising a feature space into regions corresponding to several classes. There is the expectation that to treat the additional objects a new training process is not required.

According to (Patil & Patil, 2013), there are three popular classification methods: *k*-nearest neighbours (k-NNs), artificial neural networks (ANNs) and support vector machines (SVMs). Localised versions of these methods are very popular because they offer certain advantages in terms of computation time and performance (Bischl, Schiffner, & Weihs, 2013).

Classification is widely used in fields like pattern recognition and data mining. (Savchenko, 2016) proposes that to solve problems involving feature vectors of fixed size, there are three classification methods that can be used: linear discriminant analysis (LDA), feed-forward multilayer perceptron (MLP), and a support vector machine (SVM).

A performance comparison between k-NN and nested generalised exemplar (NGE) is conducted in (Wettschereck & Dietterich, 1995). The experimental results indicate that the k-NN performance grows linearly with the number of training samples. The k-NN classification also performs well in many situations and applications (Steinbach & Tan, 2009). However, there is no single method that can be called the best for all tasks. It is common that a particular method is the best for specific tasks (Shavlik, Mooney, & Towell, 1991). Another possibility is the combining of classifiers (Xu, Krzyzak, & Suen, 1992).

k-NN classification seeks to find, in the training set, the closest group of *k* objects to the test object (Steinbach & Tan, 2009). The *k* variable plays an important role during classification. Results are sensitive to noise if *k* is too small. On the other hand, parameters from other classes may be included if *k* is too large. Figure 2.24 illustrates the k-NN with various *k* variables, from small to large.



(a) Neighborhood too small.
(b) Neighborhood just right.
(c) Neighborhood too large.

Figure 2.24: k-NN classification with small, medium, and large k *(Steinbach & Tan, 2009)*

SVM is the classification method that finds a hyperplane to separate two classes of given samples. According to (Xue, 2009), SVM is not only a well-performing classification, but it also has high accuracy when predicting future data. Figure 2.25 shows a hyperplane separating two classes with SVM. The hyperplane can be defined by the following discriminant function *g(x)*:

$$g(x) = w^T x + b \qquad (9)$$

where *w* and *b* are the weight vector and bias of the optimal hyperplane, respectively.

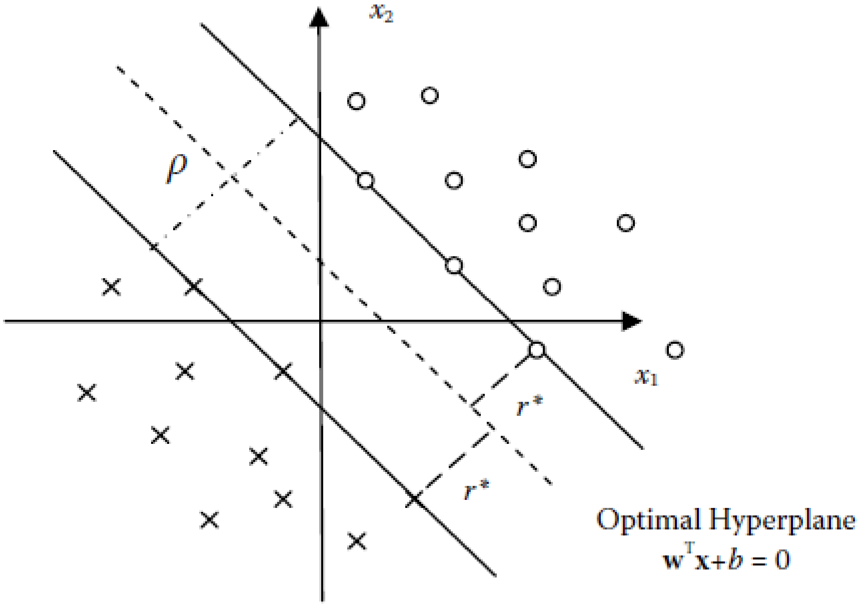


Figure 2.25: An illustration of hyperplane for a linear separable case (Xue, 2009)

The margin of separation ρ is determined by the shortest geometrical distance r from the two classes. It can be calculated as:

$$r = \frac{g(x)}{||w||} \tag{10}$$

The principal component analysis (PCA), also known as an eigenfaces approach in early works, is proposed by (Turk & Pentland, 1991) to solve 2-D face recognition problems. With PCA, the images of faces are converted into a feature space defined by the eigenvectors within the set of faces. The following steps are followed in face recognition using PCA:

1. Calculate the eigenfaces in the set of images.
2. Calculate a set of weights, based on the input image and the eigenfaces, by projecting the input image onto each of the eigenfaces.
3. Check if the image is sufficiently close to the feature space.
4. Classify the weight pattern as either known or unknown.

Figure 2.26 shows an example of a 2-D dataset and its two principal components as demonstrated by (Vidal, Ma, & Sastry, 2016).

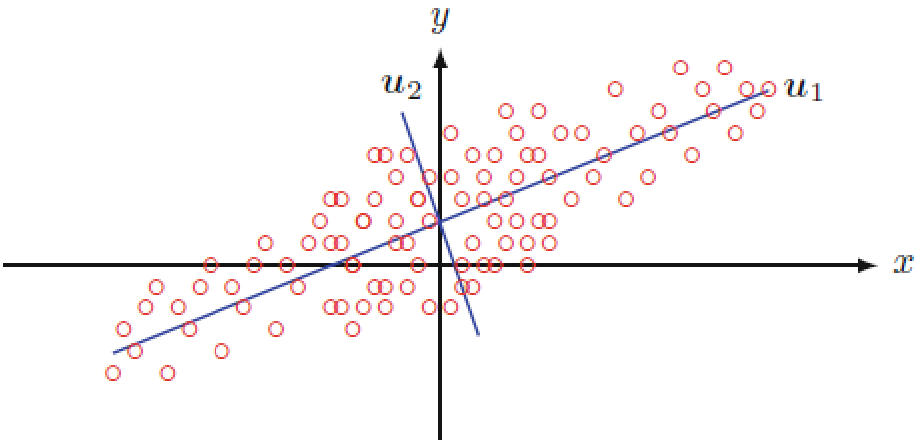


Figure 2.26: An example of 2-dimensional dataset and its two principal components (Vidal et al., 2016)

In Figure 2.26, the two eigenvectors are u_1 and u_2. Eigenvector u_1 shows a large variance within the data, while eigenvector u_2 does not.

In another work, to solve the problem of face recognition, (Belhumeur, Hespanha, & Kriegman, 1997) propose a method based on Fisher's linear discriminant called LDA. This method has lower error rates compared with PCA and is proposed to enhance PCA in face recognition under variations of lighting and facial expressions.

2.5. Summary

The literatures show that the development in the field of artificial intelligence algorithm has been moving very fast. In general, the big picture of the artificial intelligence development since it is introduced has been drawn in Figure 2.5 by (Sze et al., 2017). However, HTM is not included in the picture. Since Hawkin's idea in the book "On Intelligence" was published, HTM has played important rule in brain-inspired algorithm. HTM could be placed in the big picture of artificial intelligence development as illustrated in Figure 2.27.

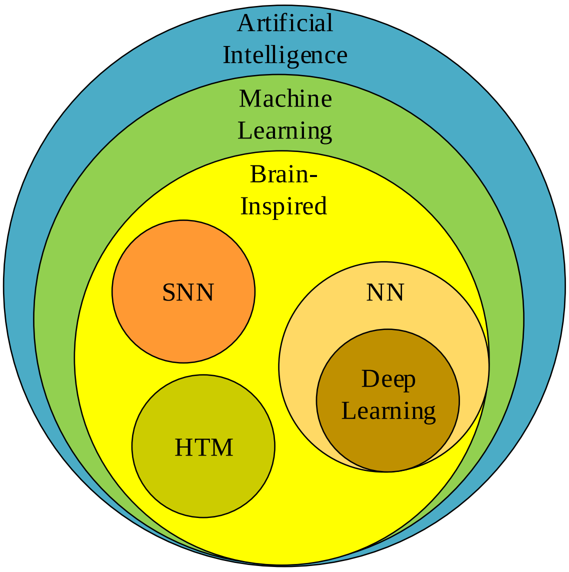


Figure 2.27. Artificial Intelligence, Machine Learning, and Brain-Inspired Algorithm

Majority of intelligence algorithm grows from basic concept of neuron or neural network, even SNN itself could be a part of neural network. A review of the literature reveals that pattern recognition or classification has three important components, namely: feature extraction, recognition, and classification. Since its early development, 2-D images have been used in pattern recognition and classification, and still to this day, the representation of features by pixels is widely implemented in image recognition and classification methods.

HCN and HTM use a receptive field to extract features, even though they both start the scanning process from the lowest level by sensing the pixels' values. In HTM, the probability of lower layer features is propagated to the next layer up. Extracting features gradually — from lower to higher levels, or from smaller to larger areas — may be aligned with the process of how humans see objects and images; however, this concept has not been widely investigated. One of the most prominent objectives of this research is to investigate the concepts of hierarchical feature extraction, similarity measurement, and classification.

Chapter 3
Methodology

The literature review in Chapter 2 shows that there are some significant gaps in the literature regarding feature extraction related to concatenation in pattern recognition applications. The research reported in this thesis aims at understanding the nature of the gaps and finding ways to fill them. A summary of the findings are written below:

1. The use of other patterns' features in feature extraction has not been deeply investigated.
2. The way to combine lower layer's features and to activate a higher layer's patterns in hierarchical networks has not been explored.
3. There is no investigation on position exchange of the reference and tested pattern during feature extraction that can lead the different result.
4. There is no experimental information to investigate several classification process that is started from the closest process to the way humans do.

This chapter presents the methodology required to verify the hypothesis that has been addressed in this thesis. According to (Jha, 2008), there are two types of research method: the first is qualitative and the second is quantitative.

The qualitative method involves two types of data. Firstly, there are empirical materials such as case studies, personal experiences, introspective accounts, life stories, interviews, observational accounts, historical accounts, interactions, and also visual texts. Secondly, there are activities and routines, and also problematic moments and meaning in an individual's life. The data that are used in a qualitative research could be descriptions of situations, events, people, interactions, observations, experiences, beliefs, correspondences, records, and history cases.

The quantitative method, on the other hand, is frequently referred to as hypothesis testing research. This is a common research operation in investigation. The hypothesis drives the research to produce the proof. Based on the above definitions of qualitative and quantitative research methods, it can be seen that this research falls in the area of quantitative research.

(Jha, 2008) states that quantitative research has several stages. It begins with the theory, and continues with a review of previous works. After the review, the hypothesis is generated, and the next step is to collect the data. Once the strategy has been confirmed, the data are then analysed based on the hypothesis. (Crnkovic, 2012), on the other hand, suggests that the research should consist of three stages: the question to be answered, the strategy to get results, and the validation to verify the results. Some examples of (Crnkovic, 2012)'s question-strategy-validation approach are summarised in Table 3.1.

Table 3.1: Research methodology suggested by (Crnkovic, 2012)

Question	Strategy	Validation
Feasibility: Is it possible to do X?	Qualitative model: Report interesting observations	Persuasion: I thought hard about this, and I believe…
Characterisation: What are the characteristics of X?	Technique: Invent new ways to do some of the tasks, including implementation techniques	Implementation: Here is a prototype of a system
Method: Can X be done better?	System: Build a Y	Evaluation: Measure Y, and compare to X
Selection:	Analytical model:	Experience:

Question	Strategy	Validation
How do I decide if X or Y?	Develop structural models that permit formal analysis	Report on use in practice

The guidelines proposed by (Jha, 2008) and (Crnkovic, 2012) are used for guidance in this research. This thesis commences with the theories reviewed in Chapter 2 and the rationale stated in Chapter 1. The following three chapters in this thesis provide an overview of the hierarchical concatenation concept for pattern recognition, feature extraction using a hierarchical concatenation network, and pattern classification using hierarchical concatenation networks. The methodology used in each chapter to address the research question is shown in Table 3.2.

Table 3.2: Research methodology in this thesis

Method	Question	Strategy	Validation
Chapter 4: An overview of the hierarchical concatenation concept for pattern recognition	Feasibility method	Technique	Implementation
Chapter 5: Feature extraction using hierarchical concatenation networks	System	Technique	Implementation and evaluation
Chapter 6: Pattern classification using hierarchical concatenation networks	System	Technique	Implementation

As this research was supposed to replicate the way humans see objects into a computer algorithm, several experiments needed to be conducted to validate the proposed method. The research questions stated in Chapter 1 guided this thesis to focus on feature extraction, concatenation, activation, and similarity measurement and classification. For this objectives, the experiment required datasets. A decision was made to start with simple datasets and increase complexity as the research progressed. Experiments started with simple datasets with the assumption that they had been normalized. Complex datasets included patterns with various levels of noise (extremely rotated or extremely noisy). Once this kind of data was applied to the HCN, its performance would degrade as the tested patterns' complexity. As the thesis focus on hierarchical feature concatenation, the image datasets used in this thesis should represent the different groups of data without considering real noisy image. A data in each group should be different in terms of style. In this stage, the ten digits of numbers with ten different style (font and thicknes) are used to see if the representation of similarity measurement. Once the proposed algorithm is able to show its ability presenting the same digits from different groups with closer percentage of similarity, the validation then moves to publicly available dataset. In this thesis, the handwritten datasets (USPS and MNIST) are used. This dataset contains huge number of digits which are written by many different people. USPS dataset has 7291 train and 2007 test images, while MNIST has 60,000 and 10,000 for training and testing images respectively. That means the style of writing will have many variations. At the end, two existing methods (LDA and PCA) were used to compare the ability of the proposed algorithm when applying small and large datasets.

In term of pattern recognition, the methods in this thesis did not use extremely distorted datasets. For example, the datasets with high degree orientation, large-scale changes, and high noise are not used. They might be used after extensive pre-processing.

3.1. Feature Extraction Using Hierarchical Concatenation Networks

The questions that need to be addressed to implement hierarchical concatenation for pattern recognition and classification are:

- How are the concatenated patterns in the upper layers activated?

- How does the position exchange of a pattern affect the similarity rate between a pattern and its extracted pattern?

To answer the questions above, the feasibility of applying hierarchical concatenation to pattern recognition is verified by implementing and testing a network. A layer-by-layer analysis of experimental results is conducted to test the hypothesis against human perception.

3.2. Pattern Classification using Hierarchical Concatenation Networks

After implementing feature extraction and comparing the pattern and its extracted pattern, another question arises:

- Which combination of HCN's feature extraction method and similarity measures can produce a better performance rate?

To answer this question, several explorations on feature extraction (activation and the position exchange) are conducted. For the quantitative analysis, similarity measurements are investigated in order to see which activation method and which similarity measures will produce better classification rates.

3.3. Summary

The two main research methodology types are qualitative and quantitative. Since the quantitative research method is commonly used in hypothesis testing research, this method is selected to verify the hypothesis in this research. Several questions that need to be answered in the remaining chapters have been listed in Sections 3.1 and 3.2.

Chapter 4
An Overview of Feature Extraction, Similarity Measurement, and Classification using Hierarchical Concatenation Networks

This chapter presents the concept of hierarchical concatenation networks (HCNs) applied to pattern recognition, which is based on the way humans make sense of the world using vision as reference. The idea comes from the process we use to recognise visual patterns. To recognise an object, we rapidly scan it, looking for key features. The process is known as saccadic eye movement. The brain then uses unique information contained in such features to identify the complete object. This concept gives the idea to replicate the process within computer programming by looking at the way how computers scan the image. The discussion in this chapter begins with the process of concatenation — the process of linking things together in a series — from a human perspective, and continues with the implementation of a computer algorithm. It then concludes with a brief description of the classification process involved.

In general, the process of feature extraction using HCN is shown in Figure 4.1.

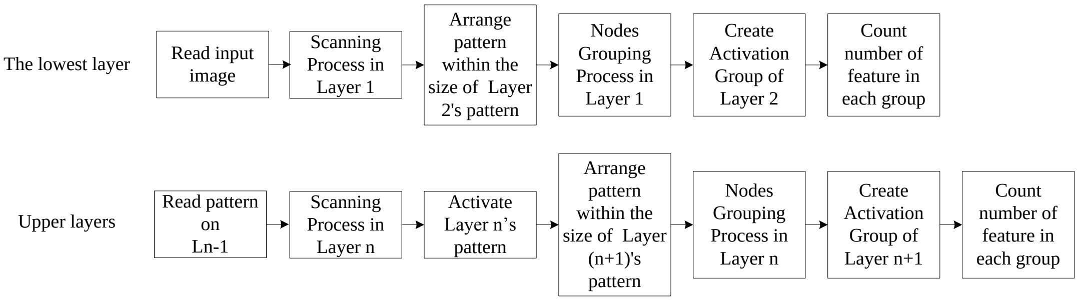

The lowest layer
Read input image
Scanning Process in Layer 1
Arrange pattern within the size of Layer 2's pattern
Nodes Grouping Process in Layer 1
Create Activation Group of Layer 2
Count number of feature in each group
Upper layers
Read pattern on Ln-1
Scanning Process in Layer n
Activate Layer n's pattern
Arrange pattern within the size of Layer (n+1)'s pattern
Nodes Grouping Process in Layer n
Create Activation Group of Layer n+1
Count number of feature in each group

Figure 4.1: A general flow diagram of HCN for feature extraction

4.1. Hierarchical Concatenation from the Human Perspective

To explain hierarchical concatenation, the two patterns shown in Figure 4.2 are used as examples. Both images are simple 2-D representations of familiar objects. It would not be difficult for an adult human to identify the image on the left (in Figure 4.2) as a monkey's head, and the image on the right as some kind of fruit. Even animals (e.g. cats, dogs, rats, etc.) can identify various objects. An animal's brain often performs in a similar way to the human brain when recognising objects.

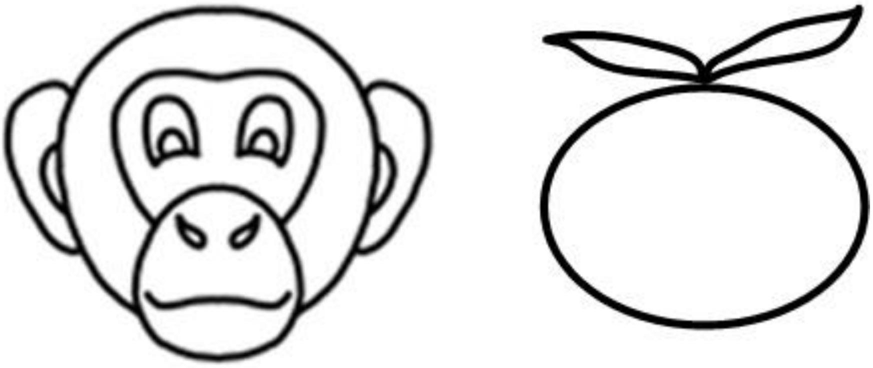


Figure 4.2: 2-D image examples

The explanation in this chapter uses the image of a monkey's head as a reference image and the image of a fruit as a test image.

In humans, real objects are reflected as an image on the retina through the cornea and crystalline as discussed in Herman (Herman, 2016). To be able to focus on this image on the retina, light is needed. Without light, objects cannot be reflected onto the retina. A comprehensive work of eyes' behaviour when interpreting the world, including the focussing process of the corneal system, is presented by Nishino (Nishino & Nayar, 2006).

The eye can only focus on a specific object with a specific coverage area. Other objects outside the focus area are not clearly reflected, or they are perceived as blurred images (Herman, 2016). The behaviour of the eye in terms of words recognition is a letter-by-letter scan, illustrated below:

U n i v e r s i t y

For example, a sighted and literate person to decide what that word is, the eyes' focus might move from the first letter to the last. At time *t=1,* when the eyes are focusing on the first letter (i.e. the letter 'U'), the eyes might not able to recognise other letters at the same time. The eyes would focus sequentially in time to the rest of the letters. When focus is at the end of the letters, the whole word is recognised as 'University'. By the decision could be made with the word's meaning as 'an institution to conduct teaching and learning at the highest level in several programs studies and authorised to confer both undergraduate and graduate degrees' (www.dictionary.com).

The process of reading the above word may be similar to the process of recognising the two 2-D images in Figure 4.2. When reading, the combination of letters forms a word that has a specific meaning, and when recognising images, the combination of features also forms a pattern that has meaning.

The eyes' rapid movement to see the features of an object is known as a saccade. To recognise objects, research shows that a specific extraction process from the whole pattern takes place (Chernyak & Stark, 2001; Holland & Komogortsev, 2013; Kresevic, Marinovic, Johnston, & Arnold, 2016; Mata, Morales, Romero, & Rubio, 2015; Wong, 2014; M. Zhao, Gersch, Schnitzer, Dosher, & Kowler, 2012). The process of recognition is conducted by extracting individual features from the given object and then comparing them to features that have been seen in the past. To identify the objects in Figure 4.2, features — like the eyes, nose, mouth, ears, leaves, and other curves — are firstly individually identified and then combined in the brain to be recognised as a monkey's head, or a fruit, or something else.

Figure 4.3 shows the labels for individual features of both the images in Figure 4.2. These features must be recognised and assembled (concatenated) before a decision can be made on what the object is. The way the human eye examines individual letters as the features, and then combines the known letters as a word, is similar to the image recognition process for Figure 4.3.

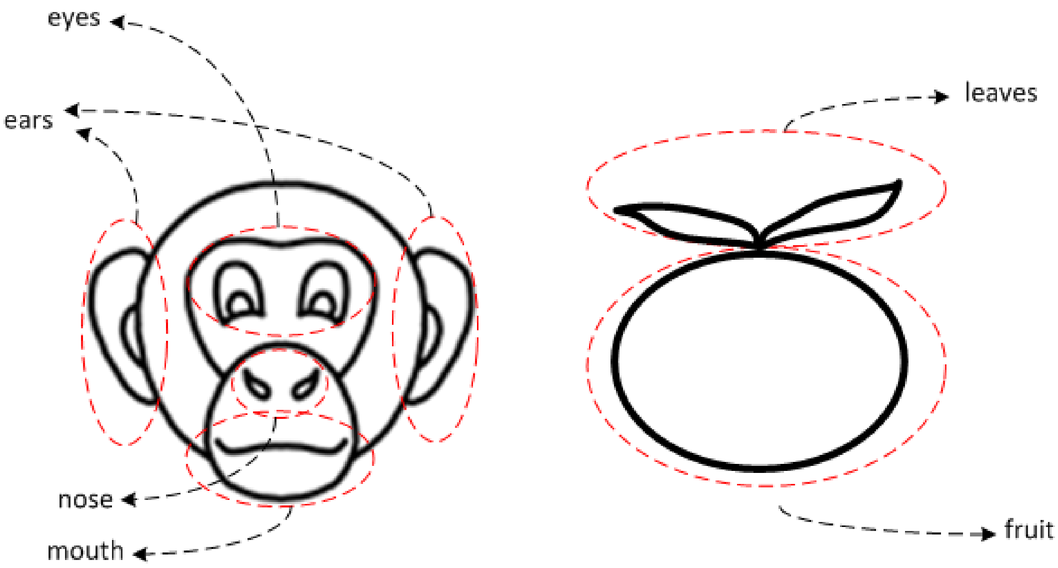


Figure 4.3: Example of feature extraction as part of the object recognition process

Vertebrates perform all the described tasks involved in object-recognition very easily and almost unconsciously. This is because each individual part of the object has previously been seen. Even though the right-side image can be easily identified as a fruit, deciding what kind of fruit it is remains difficult to establish. This is because the information provided by the edged image is not enough to make a decision. The monkey image has more features, and hence more information, to assist the decision.

It is not easy for computer algorithms to replicate that kind of process. Some feature extraction algorithms, which are supposed to achieve this, remain far removed from matching the ability of the vertebrate's brain. This research is one way of exploiting the ability of computers to process extracted and concatenated information hierarchically.

Computers-based vision systems can capture every single feature of an image (i.e. the image's pixels) and extract their value (Gupta & Panchal, 2011; Mahalanobis, Shilling, Muise, Hines, & Neifeld, 2016; Tirunelveli, Gordon, & Pistorius, 2002; Wu, Zhang, Wei, & Ozcan, 2016). By knowing the size of an image, computer algorithms can analyse every single pixel of the image sequentially from the top left to the bottom right of the image area. This process is analogous to saccades in vertebrates' visual systems.

For example, if the size of a colour image is 32x32 pixels, then the total number of pixels is 1024. Each pixel stores values in the range 0 to 255, and the pixel's value range indicates the colour intensity of the image. In this thesis, the images are black and white. If one pixel represents the smallest part or feature of the image, its individual value is limited and does not provide sufficient information for a computer algorithm to make a decision regarding the given whole image. Pixels are simply a collection of numbers (0 and 1 for black and white images).

In Chapter 2, several pattern recognition methods are presented. Since the early development of these algorithms, researchers have tried to enhance the capability of existing recognition or classification methods, while a pixel of an image is still used to represent the smallest feature of the image.

Figure 4.4 shows the structure of a HCN with four layers. By using the image that measures 32x32 pixels, Layers 1, 2, 3, and 4 have 256, 64, 16, and 4 nodes, respectively. The node in Layers 2 to 4 is the concatenation of the four nodes in its lower layer.

In this thesis, nodes are used to temporarily store patterns in the network's layers after scanning. The node in Layer 1 is the concatenated four pixels of the image, while the node in Layers 2 to 4 is the concatenated four nodes of Layers 1 to 3, respectively.

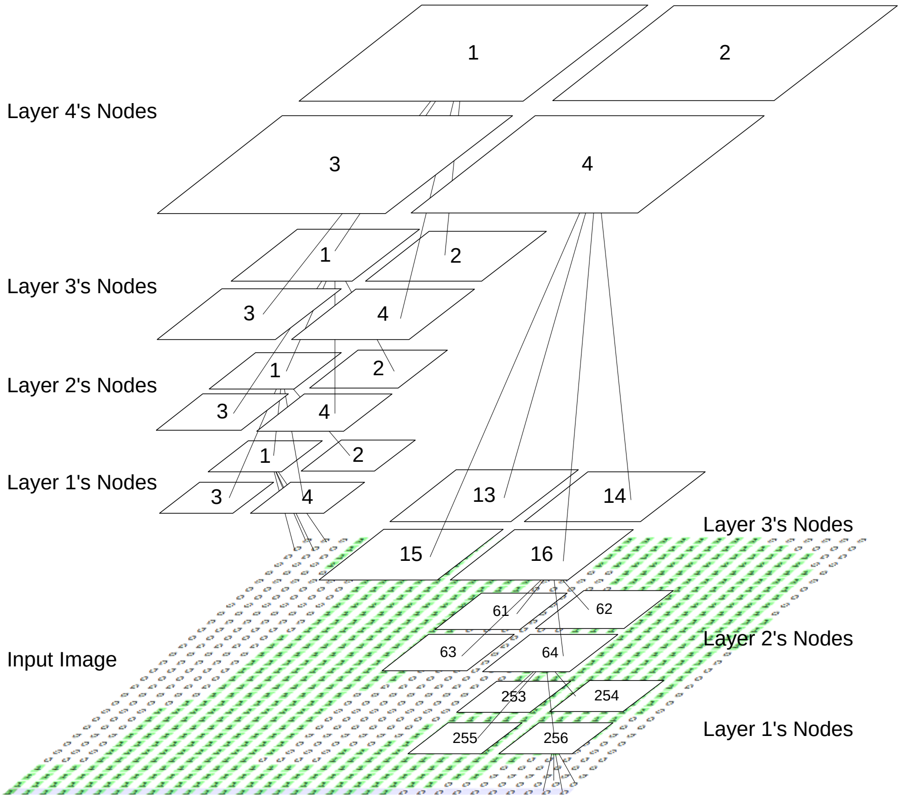
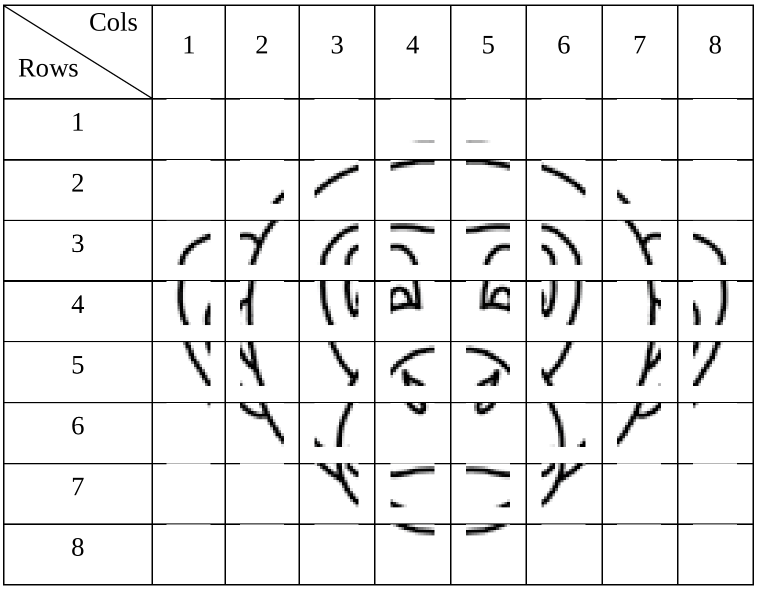


Figure 4.4: Structure of HCN

Let's assume the network has only two layers (Layers 1 and 2). In Table 4.1, the image has been divided into 64 parts (organised in an array of eight rows and eight columns). It is assumed that these 64 parts are each the smallest division of the image in Layer 1. In a real image, the smallest part is represented by a pixel.

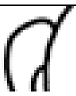
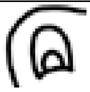
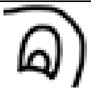
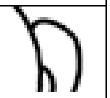
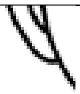
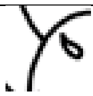
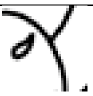
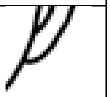
Table 4.1: Example of an image that has its smallest features divided into 64 parts



Cols Rows	1	2	3	4	5	6	7	8
1								
2								
3								
4								
5								
6								
7								
8								

Table 4.2 shows an assumption of patterns in Layer 2 or in the highest level of the two-layer network. Each feature in Layer 2 emerges from the concatenation of the features in Table 4.1. The concatenation means combining four nodes of Table 4.1 (within a 2x2 square node) to form one node of Table 4.2. This is the basic idea behind this process.

Table 4.2: Image from Figure 4.2 after it has been divided into 16 parts

Cols / Rows	1	2	3	4
1				
2				
3				
4				

The purpose of the research in this thesis is to investigate and explore the possibility of hierarchical concatenation for image recognition and classification. As explained previously, the basic idea is illustrated in Tables 4.1 and 4.2. As the features in Table 4.2 are the concatenation of features in Table 4.1, Table 4.2's features are the features on the highest layer and may provide further information to support the decision regarding recognition or classification.

Even though the features in the upper layer are the concatenation of the lower layer's features, the upper layers' features in this thesis are not directly combined. This is because each 4-square cell from Table 4.1 forms a cell of features in Table 4.2. There is an activation process, which needs to be conducted to activate the upper layer's features. For example, the partial image in Table 4.2, located at position (1, 1), is the product of the concatenation features in Table 4.1, located at (1, 1), (2, 1), (1, 2), and (2, 2). However, the representative feature in this cell should be the product of an activation process. The activation process of the upper layers' features is explained in Sub-Chapter 4.1.2.2.

The following subsections present the process of concatenation from both the point of view of a human (a high-level perspective) and a computer algorithm.

4.1.1. The Scanning Process from a High Level Perspective

This subsection uses the image of the monkey's head in Figure 4.2 to explain the scanning and grouping processes. This example image uses a network with two layers for concatenation. The layers above Layer 2 conduct the same process. The algorithm also implements sequences of the scanning process. Sequential scanning in this thesis means shifting the input image circularly by one column to the left (see Figure 4.6). The example in Figure 4.5 shows that the image has eight columns and therefore eight sequences. It also shows the whole image within the first sequence. All sequences are shown in Figure 4.6.

The smallest parts of the image in Table 4.1 are used as the input for Layer 1 as shown in Figure 4.5. The smallest parts of the image in Figure 4.5 form the basis of the features in Layer 2. The red square in Figure 4.5 is a coincidence array that measures 2x2 square cells. The coincidence array in layer 1 is used to scan the features in Layer 1 by concatenating the features of the four cells within it. Upper layers' coincidence arrays do scanning in their layer. Scanning features means recording the features that are covered by the coincidence array. To scan all the features in Layer 1, the coincidence array is shifted across the image from coordinate (1, 1) to (8, 8).

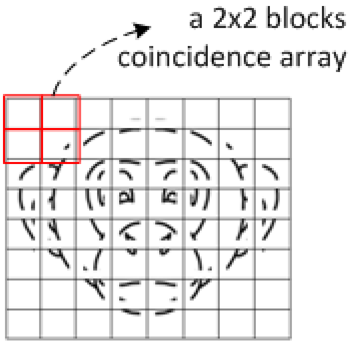


Figure 4.5: Coincidence array used to scan each single part of the pattern

Scanning is the process of extracting features from the whole pattern using coincidence arrays. Figure 4.5 shows a coincidence array about to start scanning an image from the first node to the last node. The coincidence array moves from the top-left to the bottom-right of the image. Each single movement of the coincidence array records four features that are temporarily stored in an associated node that is covered by the coincidence array. There are 16 nodes in Layer 1 (see Table 4.2). The upper layer has four nodes as shown in Table 3.

Table 4.3: Image from Figure 4.2 after it has been divided into four parts

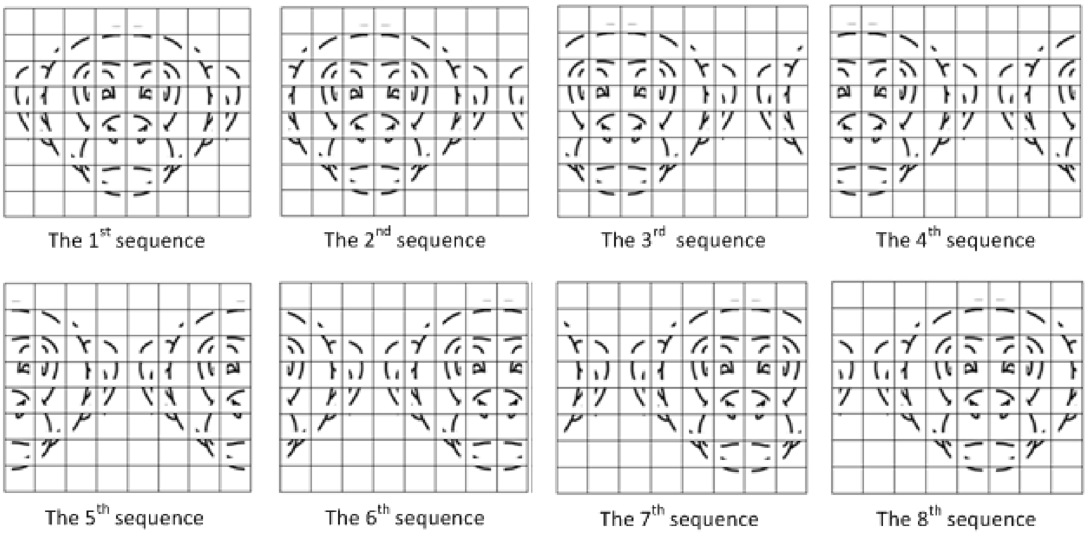
Cols Rows	1	2
1		
2		

The number of nodes in each layer *(nNodesL$_n$)* is shown in Equation 4.1. As the image is a square image, the image's height and width is the same. Equation 4.1 requires the width or height of the input image to calculate layer *n (widthL$_n$)*.

$$nNodesL_n = \left(\frac{widthL_n}{2}\right)^2 \tag{4.1}$$

By applying Equation 4.1, the movement of the coincidence array has no overlap between nodes. This thesis chooses not to implement overlapping within the coincidence array movement, but instead adopts the sequential scanning process presented below.

Figure 4.6 shows the sequence of image input, which shifts the image by one column to the left during scanning. The first sequence image in Figure 4.6 is the original input into Layer 1 of the HCN. In this example, the total number of features that are stored in each node is eight — one per every shift of the image.



The 1st sequence

The 2nd sequence

The 3rd sequence

The 4th sequence

The 5th sequence

The 6th sequence

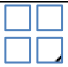
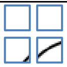
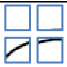
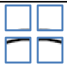
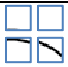
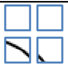
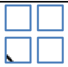
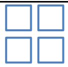
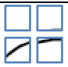
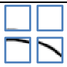
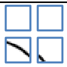
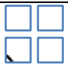
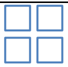
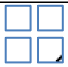
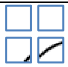
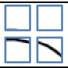
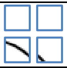
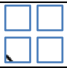
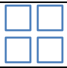
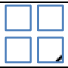
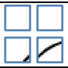
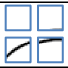
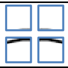
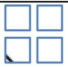
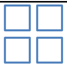
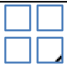
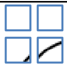
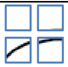
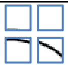
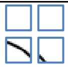
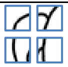
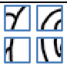
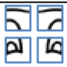
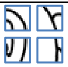
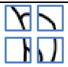
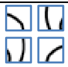
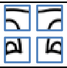
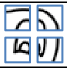
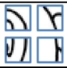
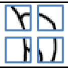
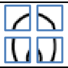
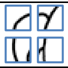
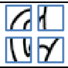
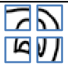
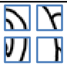
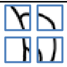
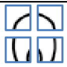
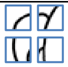
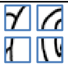
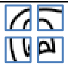
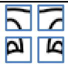
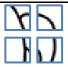
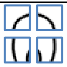
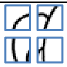
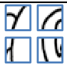
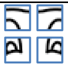
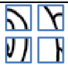
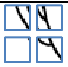
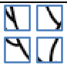
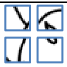
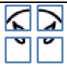
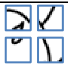
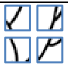
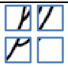
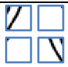
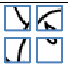
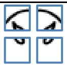
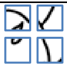
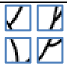
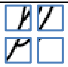
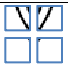
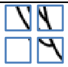
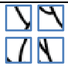
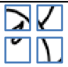
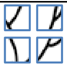
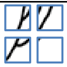
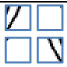
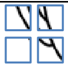
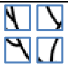
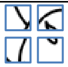
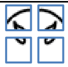
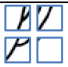
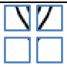
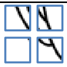
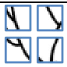
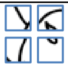
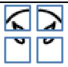
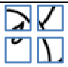
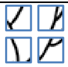
The 7th sequence

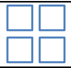
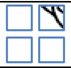
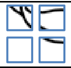
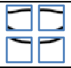
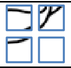
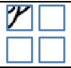
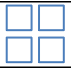
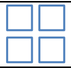
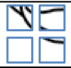
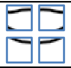
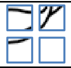
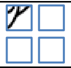
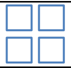
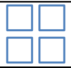
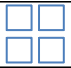
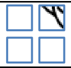
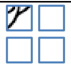
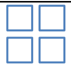
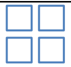
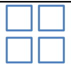
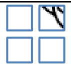
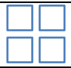
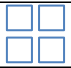
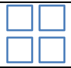
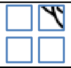
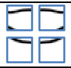
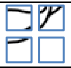
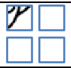
The 8th sequence

Figure 4.6: The sequences of patterns

Table 4.4 shows the features within the nodes of Layer 1, which are also the outputs of scanning process. For every sequence, a node receives four features. Some nodes have similar features after scanning, i.e. Node 16 has the same features at the first, second, and third sequence.

Table 4.4: Scanning process

Nodes	Sequences							
	1	2	3	4	5	6	7	8
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								

Nodes	Sequences							
	1	2	3	4	5	6	7	8
13								
14								
15								
16								

Implementation of a sequential scanning process allows each node to see more features, instead of one feature with no sequences. The differences between implementing a sequence and not implementing a sequence are presented in Chapter 5. Both methods give different results.

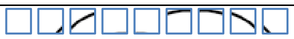
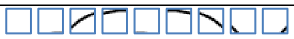
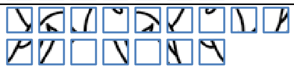
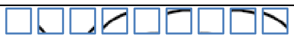
4.1.2. The Grouping Process

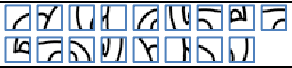
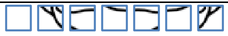
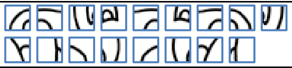
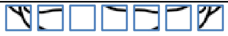
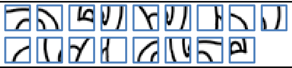
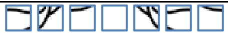
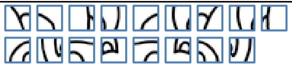
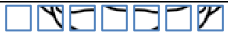
Grouping is conducted after the scanning process. It involves two processes. The first process is the grouping of the features in each node based on patterns' similarity. These are called node groups. The second process is the grouping of four nodes that will activate the upper layer's patterns. These are called activation groups. The following two subsections describe in detail each of these processes.

4.1.2.1. Node Groups

Node grouping reduces memory use by removing redundant features in each node. Table 4.5 shows the group of unique features in each node after removing all repeats. As shown in Table 4.4, after the scanning process, Node 1 has 32 features. Features that appear more than once are removed from the group, and after grouping, Node 1 has nine unique features.

Table 4.5: Group of unique features in each node

Nodes	Group of Features	No. of Pat	Nodes	Group of Features	No. of Pat
1		9	9		16
2		9	10		16
3		9	11		16
4		9	12		16

Nodes	Group of Features	No. of Pat	Nodes	Group of Features	No. of Pat
5		17	13		7
6		17	14		7
7		17	15		7
8		17	16		7

Scanning and grouping in Layer 1 are part of the features extraction process in the proposed network. Layer 1 is the lowest layer in the network that does not conduct the activation process, which activates features on a layer. Layer 1's features are a product of direct concatenation of the four smallest input features within the coincidence array.

4.1.2.2. Activation Groups

Activation grouping determines the features' activation groups in an upper layer. The research in this thesis has investigated two ways of activation called HCN-I and HCN-II. The differences between them are presented in Chapter 5. The explanation in this chapter uses HCN-I. Since the number of features in four concatenated nodes may not be similar, the upper layer's patterns are represented by the number of unique features within the nodes. For example, if a node has three unique features, the feature in this node is represented by binary '111' or decimal '7'.

The example in Figure 4.7 shows how Node 1 in Layer 2 is the concatenation of Nodes 1, 2, 5, and 6 from Layer 1.

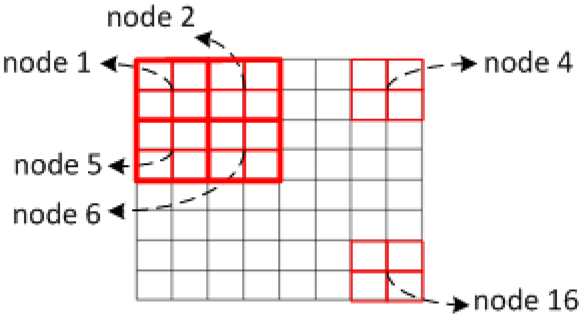
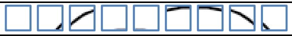
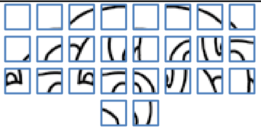
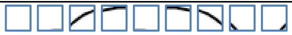
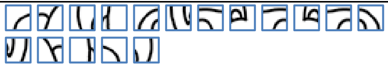
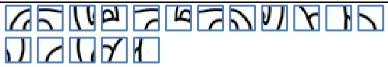


Figure 4.7: The concatenated nodes

Table 4.6 shows the unique features in Nodes 1, 2, 5, and 6. It should be noted that Nodes 1 and 2 have the same features, while Nodes 5 and 6 share the same set.

Table 4.6: The activation group of concatenated nodes

Nodes	Group of Unique Features in Each Node	Activation Group to Activate Node 1 of Layer 2	Number of Patterns in the Group
1			26 patterns
2			
5			
6			

To create a concatenated group of features, the union operation is performed as described in Equation 4.2.

$$n1 \; \cup n2 \; \cup n5 \cup n6 = \{p : p \in n1 \; or \; p \in n2 \; or \; p \in n5 \; or \; p \in n6\} \tag{4.2}$$

A closer inspection of the features in each node (as shown in Table 4.5) shows that nodes within the same row have the same patterns. This is because of the horizontal movement of the coincidence array from left to the right over the image's area.

By performing a union operation, the features in Nodes 1, 2, 5, and 6 are grouped to create only 26 unique patterns. This group is then propagated in Layer 2 to activate Layer 2's features.

4.1.3. Activating the Upper Layer's Patterns

So far in our example, the scanning process in Layer 1 produces the features shown in Table 4.4. These features are then grouped as shown in Table 4.5. The concatenated nodes within the coincidence array create the activation group as shown in Table 4.6. This activation group is used to activate the upper layer's features. There is a difference between the activation process for a reference image and the activation process for a tested image. A reference image uses its own activation group, while a tested pattern will use the activation group of the reference image. The figure below shows Node 1 in Layer 2 of the monkey's head as a reference image.

After the scanning process in Layer 2, the features in this layer must be activated. The two activation methods can be imagined as when we see an object and we find an eye as a part of that object. At this time we can recognize the eye if we have seen it previously. We could not make the final decision of what the object is until we finish scan all parts of the object. The eye itself is a form of several shapes and constructed from several materials. The collection of information of shapes and materials lead the temporary decision be pointed to an eye. The experiment in this thesis was focused on the shapes of the images, hence the two proposed activation methods were established to find the possibility of representing the upper layer’s features. The activation features in Node 1 of Layer 2 are shown in Figure 4.8.

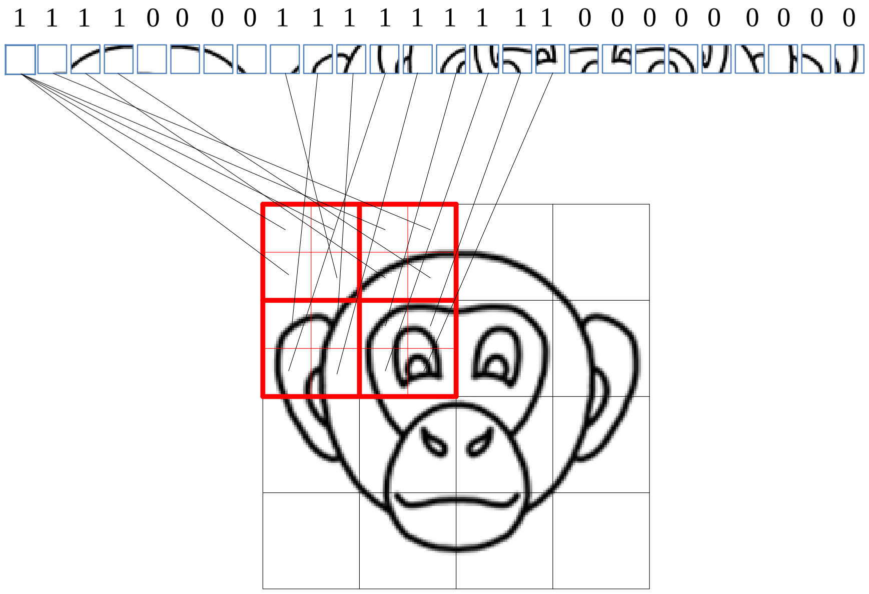


Figure 4.8: An illustration of an upper layer's feature

When referring to Table 4.6, Node 1 in Layer 2 is the concatenation of Nodes 2, 5, and 6 of Layer 1. The activation group for this node has 26 different features. The features in the upper layers’ nodes are represented as binary numbers. Hence, the features have the same number of bits as the number of features within the activation groups. A bit will be active if the coincidence array sees a similar feature to the features within the activation group. Active bits are indicated by ‘1’ and inactive bits are ‘0’.

Node 1 of Layer 2 is used in this example. Table 4.7 shows all nodes in Layer 2, where each node is the concatenation of four Layer 1 nodes within the coincidence array of Level 2.

Table 4.7: Layer 2 nodes, which are concatenated from Layer 1 nodes

Layer 1 Concatenated Nodes	Layer 2 Nodes
1, 2, 5, 6	1
3, 4, 7, 8	2
9, 10, 13, 14	3
11, 12, 15, 16	4

After the features in Node 1 are activated, the process continues, activating features in the rest of the nodes. The activation process explained above is based on Figure 4.5 and applied to the first sequence of the input image in Figure 4.6. The remaining sequences as shown in Figure 4.6 then follow the same process.

4.1.4. Similarity Measurements

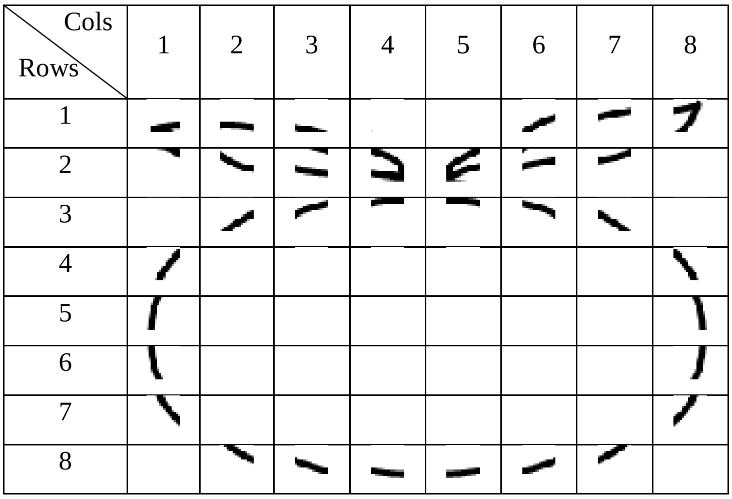
Similarity measures are important when distinguishing two objects or patterns. Some similarity measures have been presented in Chapter 2. In this thesis, the similarity between two patterns is measured in a simple way.

For example, let's assume there are two vectors: A = [0 4 8 12 13 14 15] and B = [0 1 5 7 8 10 14 15]. The similarity between the vectors is calculated by counting the number of similar elements in each vector, A and B, and then dividing the largest element number between the two vectors. In this example, there are '2' similar elements between A and B, which are '0' and '15', and the largest element number is '8' (from vector B). Hence, the percentage of similarity between vector A and B is (2/8)*100 or 25%.

The similarity between the images in Figure 4.2 — the reference image of the monkey's head and the tested image of the fruit — is calculated node by node in both Layer 1 and Layer 2. Before measuring the similarity between the two images, the features of the fruit must be extracted using HCN.

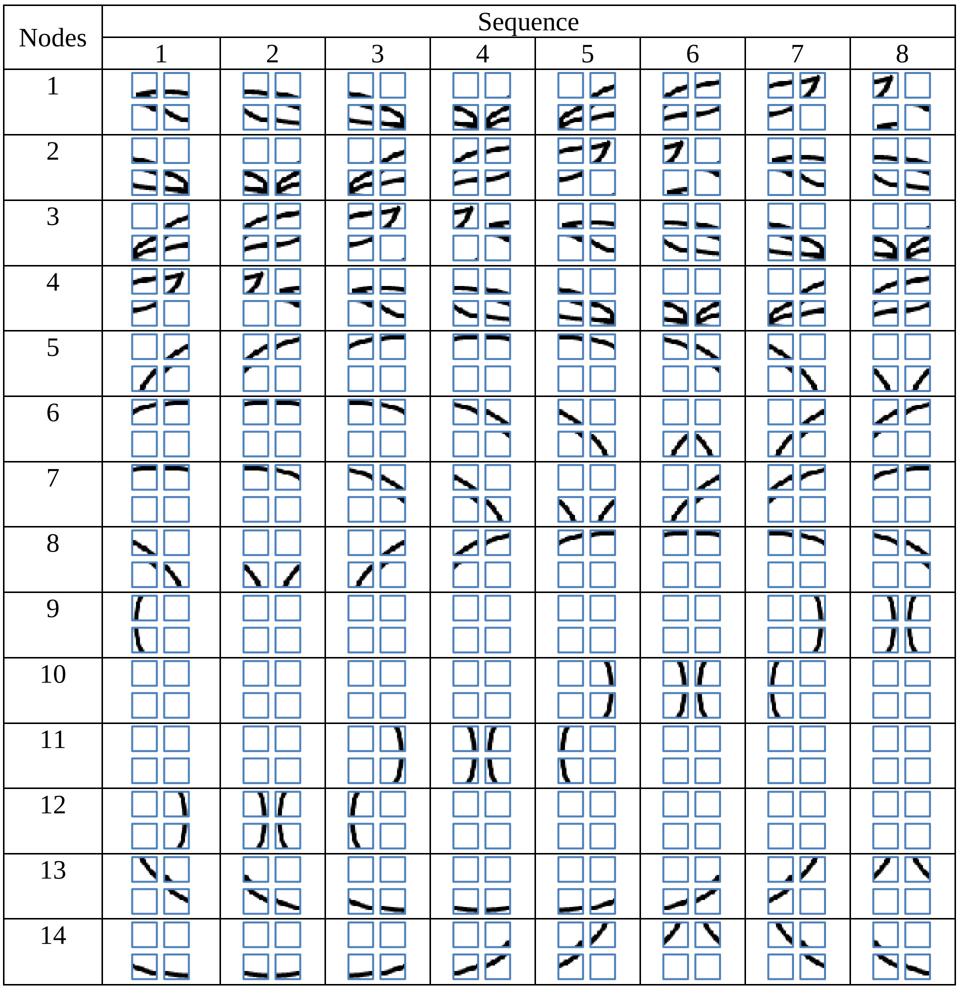
Table 4.8 shows the features of the image presented to Layer 1.

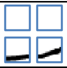
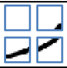
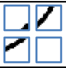
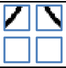
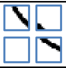
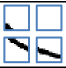
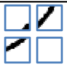
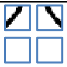
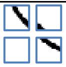
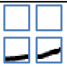
Table 4.8: Example of a tested image presented to Layer 1


Cols
Rows
1
2
3
4
5
6
7
8

After performing the scanning process on Layer 1, the features of each node in every sequence are as shown in Table 4.9.

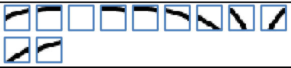
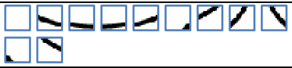
Table 4.9: Tested image's features after scanning in Layer 1


Nodes
Sequence
1
2
3
4
5
6
7
8
9
10
11
12
13
14

Nodes	Sequence							
	1	2	3	4	5	6	7	8
15								
16								

HCN then performs node grouping on the tested image. Activation groups for the higher layers are not required, because the features in the upper layers use the activation groups from the reference image. The unique features in each node of the tested image are shown in Table 4.10.

Table 4.10: Group of unique features in each node of the tested image

Nodes	Group of Features	No. of Pat	Nodes	Group of Features	No. of Pat
1		13	9		5
2		14	10		5
3		15	11		5
4		15	12		5
5		11	13		11
6		11	14		11
7		9	15		12
8		12	16		12

The features of the fruit image are shown from a high-level perspective in Table 4.11, and represent Layer 2 with four nodes.

Table 4.11: Layer 2 perspective of the fruit image

Rows \ Cols	1	2
1		
2		

To activate the features in Layer 2 of the fruit image, the activation group is required from Node 1 in Layer 2 of the reference image (the monkey's head). The process of features activation is similar to the process that is conducted using the monkey's head as shown in Figure 4.9.

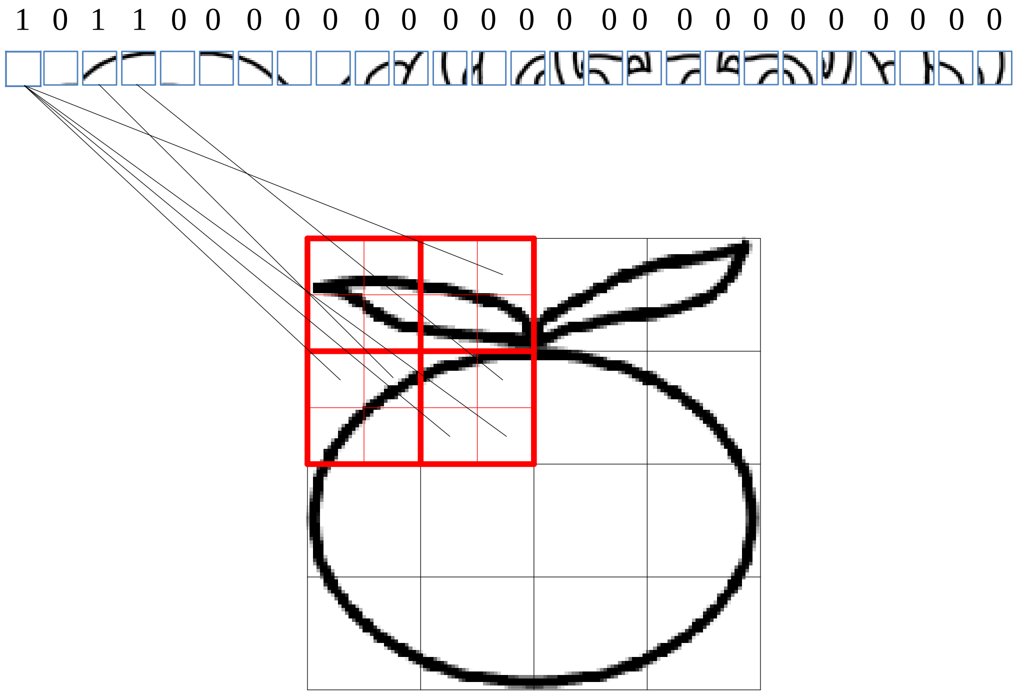


Figure 4.9: Features activation in Node 1, Layer 2 of the tested image

In the first sequence, the first node in Layer 2 now contains a feature '10110000000000000000000000'.

The explanation of similarity measurement in this chapter uses the first node of both a reference image and a tested image as the example. Table 4.12 shows the process to calculate the percentage of similarity between the first node of the tested image and the first node of the reference image.

Table 4.12: Example of similarity measurement between Node 1 of reference image and Node 1 of tested image

First node in Layer 2 of Tested Pattern	‘**1**1**110000**111111111**000000000**’
First node in Layer 2 of Learnt Pattern	‘**1**0**110000**000000000**000000000**’
Similarity in bits	16 bits
Similarity in %	(16/26)*100 = 62%

To count the total percentage of similarity between the two images, the similarity calculation has three steps. These steps are:

1. Similarity node by node (*sim_node),* which compares similarity node by node between the reference image (R) and tested image (T) (Equation 4.3).
2. Similarity layer by layer *(sim_Layer),* which calculates the average of the nodes’ similarity in each layer (Equation 4.4).
3. Total similarity *(sim_tot),* which takes the average similarity from all layers (Equation 4.5).

$$sim\_node_n = \frac{num\_sim\_features(R,T)_n}{Total\ Number\ of\ Bits} \tag{4.3}$$

$$sim\_Layer_L = \frac{\sum_1^{num\_node_L} sim\_node_n}{num\_node_L} \tag{4.4}$$

$$sim\_tot_{(R,T)} = \sum_{L=1}^{L=m} sim\_Layer_L \tag{4.5}$$

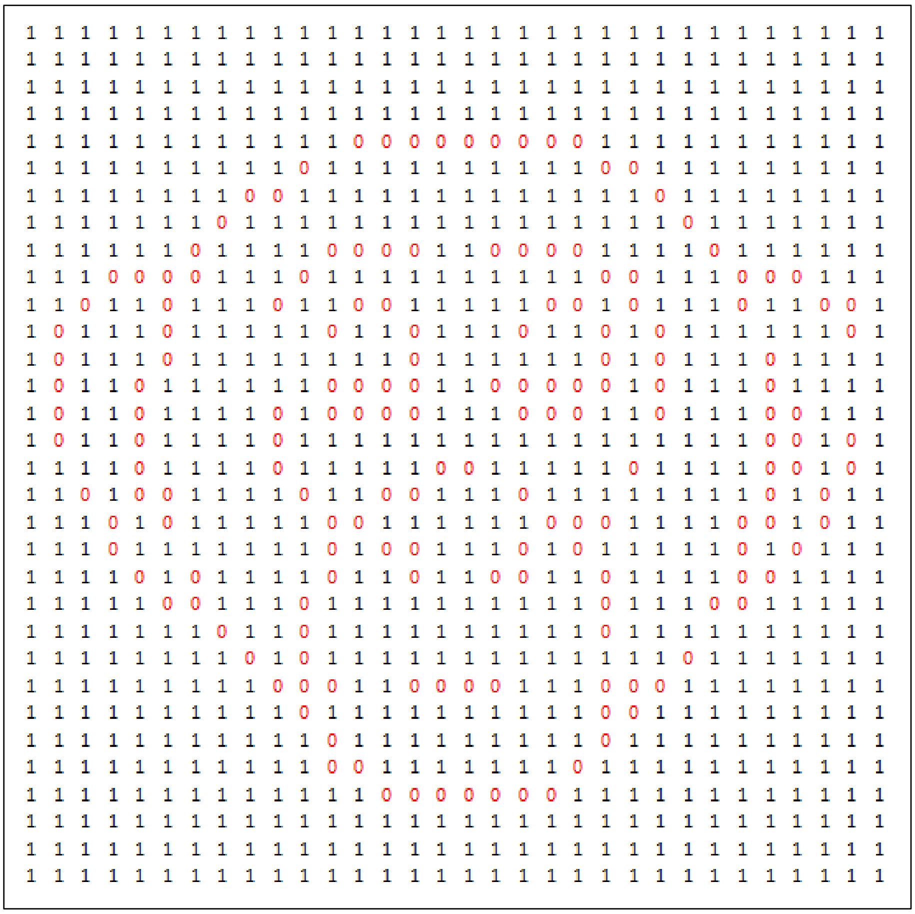
4.2. Applied Hierarchical Concatenation Algorithm

The structure of the hierarchical concatenation network (HCN) is shown in Figure 4.4, while an illustration of how the detailed features of an image are seen from a human perspective is presented in Section 4.1. This section will present the implementation of a HCN to extract the features from each image and measure their similarity. Once again, the image of the monkey’s head and the image of a fruit will be used as the reference image and tested image, respectively.

The feature extraction process using HCN is shown in Figure 4.1, and this acts as the guide to splitting the features from both the reference and tested patterns. The following example presents the feature extraction process for the reference image first and then the tested image. After features from both images are extracted from all layers, the similarity measures are presented.

4.2.1. Feature Extraction of a Reference Image

Before an image is applied to a HCN, it needs to be converted into a binary image. The image of the monkey's head has the binary features shown in Figure 4.10.



```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 0 0 0 0 1 1 1 1 0 1 1 1 1 1 1
1 1 1 0 0 0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 0 0 0 1 1 1
1 1 0 1 1 0 1 1 1 0 1 1 0 0 1 1 1 1 1 0 0 1 0 1 1 1 0 1 1 0 0 1
1 0 1 1 1 0 1 1 1 1 1 0 1 1 0 1 1 1 0 1 1 0 1 0 1 1 1 1 1 1 0 1
1 0 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 1 1 1 0 1 1 1 1
1 0 1 1 0 1 1 1 1 1 1 0 0 0 0 1 1 0 0 0 0 0 1 0 1 1 1 0 1 1 1 1
1 0 1 1 0 1 1 1 1 0 1 0 0 0 0 1 1 1 0 0 0 1 1 0 1 1 1 0 0 1 1 1
1 0 1 1 0 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1
1 1 1 1 0 1 1 1 1 0 1 1 1 1 1 0 0 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1
1 1 0 1 0 0 1 1 1 1 0 1 1 0 0 1 1 1 0 1 1 1 1 1 1 1 1 0 1 0 1 1
1 1 1 0 1 0 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0 0 1 1 1 1 0 0 1 0 1 1
1 1 1 0 1 1 1 1 1 1 1 0 1 0 0 1 1 1 0 1 0 1 1 1 1 1 0 1 0 1 1 1
1 1 1 1 0 1 0 1 1 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 1 1 0 0 1 1 1 1
1 1 1 1 1 0 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 0 1 1 1 1 1
1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0 1 1 1 0 0 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Figure 4.10: Binary features of the reference image

Each bit in the binary image is a feature of the HCN input. The HCN then conducts the scanning process to extract the image's features from Layer 1 using Algorithm 4.1.

Algorithm 4.1: Scanning in Layer 1

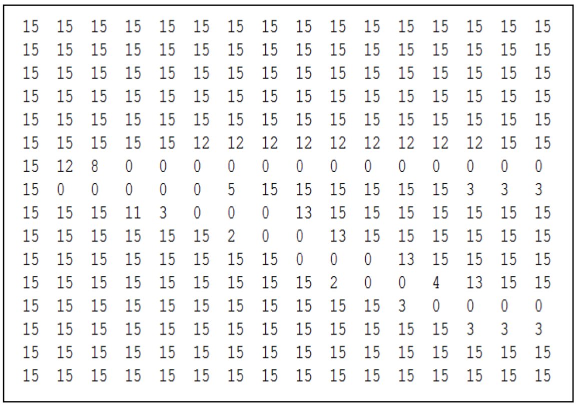
```
For k from 1 to nSeq
nNodes$L_1$ ← 1
    For i from 1 to nRows$L_1$-1
        For j from 1 to nCols$L_1$-1
            pix1 ← pat$L_n$(l)(i, j)
            pix2 ← patLn(l)(i, j+1)
            pix3 ← patLn(l)(i+1, j)
            pix4 ← patLn(l)(i+1, j+1)
            pLn ← [pix1 pix2 pix3 pix4]
            dec ← 0
            Loop m from 1 to 4
                dec ← dec + p$L_n$(m) * (2 ^ (4-m))
            End Loop m
            nodeL1(l)(nNodesL1, k) ← dec
            nNodesL1 ← nNodesL1 + 1
        End Loop j
    End Loop i
End For k
```

Within every sequence *(k)* from 1 to 32 (if the width of the image is 32, *nSeq will be 32*), each pixel of the image is captured using a 2x2 pixels coincidence array (pix1, pix2, pix3, and pix4). One movement of the coincidence array captures one feature of a node in Layer 1. A four-bit feature is then converted to a decimal value to make it easy for further calculations. After one sequence, Layer 1 produces a vector with 256 elements. The 256 elements in that vector represent the features of 256 nodes in Layer 1. There are 256, 64, 16, and 4 nodes in Layers 1, 2, 3, and 4, respectively. The numbers of nodes in Layer *n* *(n_nodeL_n)* can be calculated using Equation 4.6.

$$n\_nodeL_n = \left(\frac{width\_PatL_{n-1}}{2}\right)^2 \tag{4.6}$$

Where *$width\_PatL_{n-1}$* is the width of pattern that comes to Layer *n*.

The pattern in Layer 1, after the first sequence of the scanning process, is shown in Figure 4.11. This pattern is then used as the input to Layer 2 for its first sequence.



```
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
15  15  15  15  15  12  12  12  12  12  12  12  12  12  15  15
15  12  8   0   0   0   0   0   0   0   0   0   0   0   0   0
15  0   0   0   0   0   5   15  15  15  15  15  15  3   3   3
15  15  15  11  3   0   0   0   13  15  15  15  15  15  15  15
15  15  15  15  15  15  2   0   0   13  15  15  15  15  15  15
15  15  15  15  15  15  15  15  0   0   0   13  15  15  15  15
15  15  15  15  15  15  15  15  15  2   0   0   4   13  15  15
15  15  15  15  15  15  15  15  15  15  15  3   0   0   0   0
15  15  15  15  15  15  15  15  15  15  15  15  15  3   3   3
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
```

Figure 4.11: Layer 1's pattern of the reference image at the first sequence

At the end of the scanning process, each node in Layer 1 has 32 features. Some of these features are repeated, but using the HCN node group process, the repeated features are removed. Each node then contains only unique features. The table below shows the unique features in each node.

Table 4.13: Unique features of nodes in Layer 1

Nodes	Features	Nodes	Features	Nodes	Features
1 - 16	[15]	97 - 112	[5 7 8 9 10 12 14 15]	193 - 208	[0 1 2 3 7 10 11 15]
17 - 32	[15]	113 – 128	[0 3 5 7 10 11 13 14 15]	209 - 224	[4 7 9 10 13 14 15]
33 - 48	[3 6 11 12 13 14 15]	129 - 144	[1 3 4 9 10 11 12 13 14 15]	224 - 240	[3 7 11 15]
48 - 64	[3 6 7 9 11 13 14 15]	145 - 160	[1 2 5 6 7 9 10 11 12 13 14 15]	241 - 256	[15]
65 - 80	[3 5 6 7 8 9 11 12 13 14 15]	161 - 176	[1 3 5 6 7 8 9 10 11 14 15]		
80 - 96	[2 3 5 6 7 9 10 11 13 14 1]	177 - 192	[5 6 7 10 11 13 14 15]		

As shown by the flow diagram in Figure 4.1, an activation group is created after performing the node grouping process. The algorithm to create the activation group in Layers 2 to 4 is shown in Algorithm 4.2.

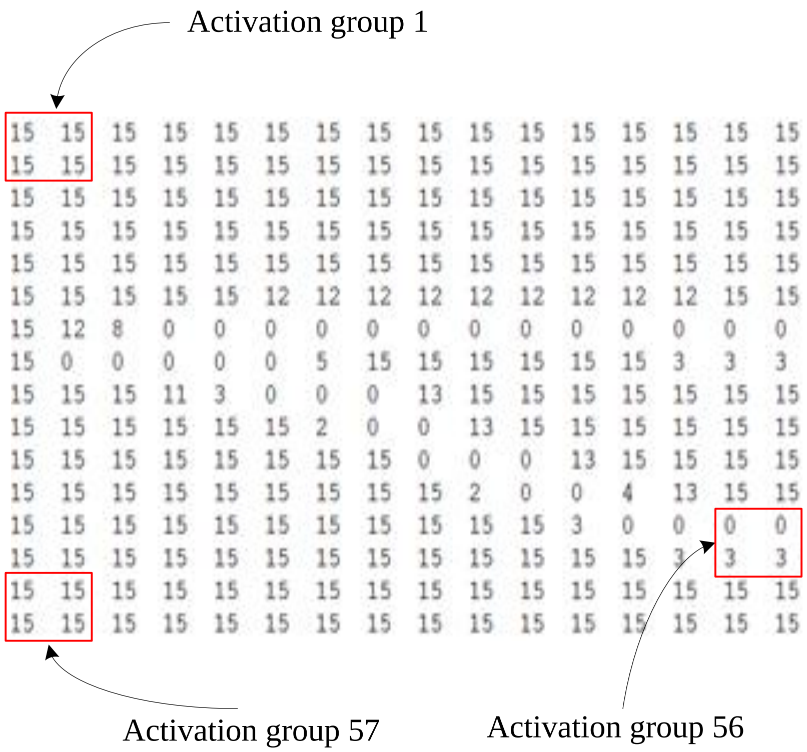
Algorithm 4.2: Creating an activation group

```
For k from 1 to nSeq
  $nNodesL_{n+1}$ ← 1
  For i from 1 to $nRowsPatL_n$-1 increase by 2
    For j from 1 to $nColsL_n$-1
      Act_GroupLn(l,nNodesLn) ← node_Ln(l,j) U node_Ln(l,j+1) U
      node_Ln(l+1,j) U node_Ln(l+1,j+1)
      nNodesLn ← nNodesL2+1
    End For j
  End For i
End For k
```

An activation group is created in every sequence. The activation group that is created in Layer *Ln=1* has as many the numbers of nodes in Layer 2 *($nNodesL_{n+1}$)*. Each activation group of nodes in *Ln+1* is a product of the union process of the four nodes in Ln within Ln+1's coincidence array. Using Equation 4.6, the number of nodes in Layer 2 is calculated as 64 nodes. Hence, there are 64 activation groups required to extract Layer 2's features.

Figure 4.12 shows the activation groups used to activate Layer 2's features within its nodes. The red squares in Figure 4.12 indicate the activation group, which is based on four concatenated nodes of Layer 1. The first activation group has four elements of the same value, '15', so the repeated values are then removed to show only one value of '15'. The activation group 61 has elements '3', '3', 15', and '15', which after removal of the repeated values, will show only two values: '3' and '15'.

After the activation groups are created, the scanning process continues to pattern in Layer 2. The scanning process of Layers 2 to 4 is conducted using Algorithm 4.3. The activation process for the higher layers' features is also shown in Algorithm 4.3, Lines 10 to 21.



Activation group 1
Activation group 57
Activation group 56

15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
15	15	15	15	15	12	12	12	12	12	12	12	12	12	15	15
15	12	8	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	5	15	15	15	15	15	15	3	3	3
15	15	15	11	3	0	0	0	13	15	15	15	15	15	15	15
15	15	15	15	15	15	2	0	0	13	15	15	15	15	15	15
15	15	15	15	15	15	15	15	0	0	0	13	15	15	15	15
15	15	15	15	15	15	15	15	15	2	0	0	4	13	15	15
15	15	15	15	15	15	15	15	15	15	15	3	0	0	0	0
15	15	15	15	15	15	15	15	15	15	15	15	15	3	3	3
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15

Figure 4.12: Activation groups for Layer 2's feature extraction

Algorithm 4.3: Scanning Process in layer *n (2 – 4)*

```
For k from 1 to nSeq
    $nNodesL_n \leftarrow 1$
    For i from 1 to $nRowsL_n$-1
        For j from 1 to $nColsL_n$-1
            $pix1 \leftarrow patL_n(l)(i, j)$
            $pix2 \leftarrow patLn(l)(i, j+1)$
            $pix3 \leftarrow patLn(l)(i+1, j)$
            $pix4 \leftarrow patLn(l)(i+1, j+1)$
            $pLn \leftarrow [pix1\ pix2\ pix3\ pix4]$
            For m from 1 to numActivationGroup_$L_{n-1}$
                For n from 1 to length_$pL_n$
                    If $pL_n$ = ActivationGroup_$L_{n-1}(l,nNodesL_n)(m)$
                        $tempPatL_n \leftarrow 1$
                    End If
                End For n
            End For m
            $dec \leftarrow 0$
            For m from 1 to numActivationGroup_Ln-1
                $dec \leftarrow dec + tempPatL_n (m) * (2 \wedge (numActivationGroup\_L_{n-1}-m))$
            End For m
            $nodeLn(l)(nNodesLn, k) \leftarrow dec$
            $nNodesLn \leftarrow nNodesLn + 1$
        End For j
    End For i
```

Algorithm 4.3: Scanning Process in layer *n (2 – 4)*

```
End For k
```

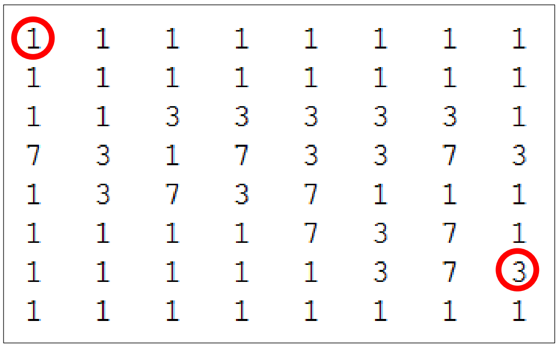
The scanning process in the upper layers is similar to the process in Layer 1; however, while Layer 1 uses its coincidence array to capture pixels, Layers 2 to 4 use their coincidence arrays to record nodes within the layer below them. These layers also require the activation process to activate their features.

For example, the activation of features in Nodes 1 and 56 of Layer 2. Figure 4.12 shows that Node 1 has features [15 15 15 15]. Using Algorithm 4.2, which includes a union operation on Line 4, the activation group of this node is calculated as ‘15’. Because this node has only one unique feature, this node’s features are represented as a vector [1]. The process for the reference pattern is illustrated in Figure 4.8, and the process for the tested pattern is shown in Figure 4.9.

Node 56 has features [0 0 3 3], however due to its activation group, this becomes two features [0 3]. These features are represented by a vector [1 1], and using this vector, the value of the features in this node is ‘3’. The value of the features in Nodes 1 and 56 at the first sequence are indicated with red circles in Figure 4.13.

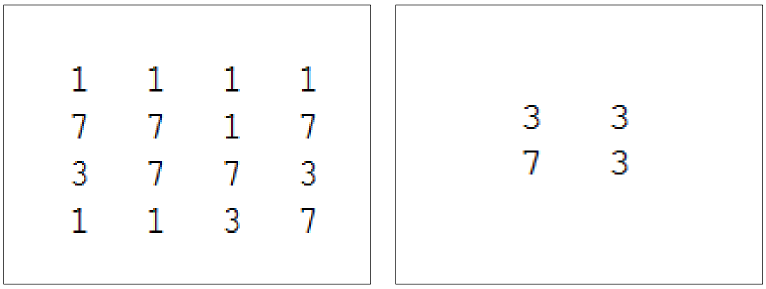
As shown in Figure 4.11, the width of the input pattern for Layer 2 is 16 columns; therefore, using Equation 4.6, the number of nodes in this layer is calculated as 64. The nodes are then arranged within the size of Layer 3’s input pattern. Figure 4.13 shows the output pattern of Layer 2 that can be used as the input pattern of Layer 3.

Layers 3 and 4 perform the same process. The input for Layers 3 and 4 is the output of Layers 2 and 3, respectively. After scanning and conducting the activation process, the patterns in Layers 3 and 4 at the first sequence are presented in Figure 4.14.



```
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
1   1   3   3   3   3   3   1
7   3   1   7   3   3   7   3
1   3   7   3   7   1   1   1
1   1   1   1   7   3   7   1
1   1   1   1   1   3   7   3
1   1   1   1   1   1   1   1
```

Figure 4.13: Layer 2’s patterns for the reference image at the first sequence



```
1   1   1   1
7   7   1   7
3   7   7   3
1   1   3   7
```

(a)

```
3   3
7   3
```

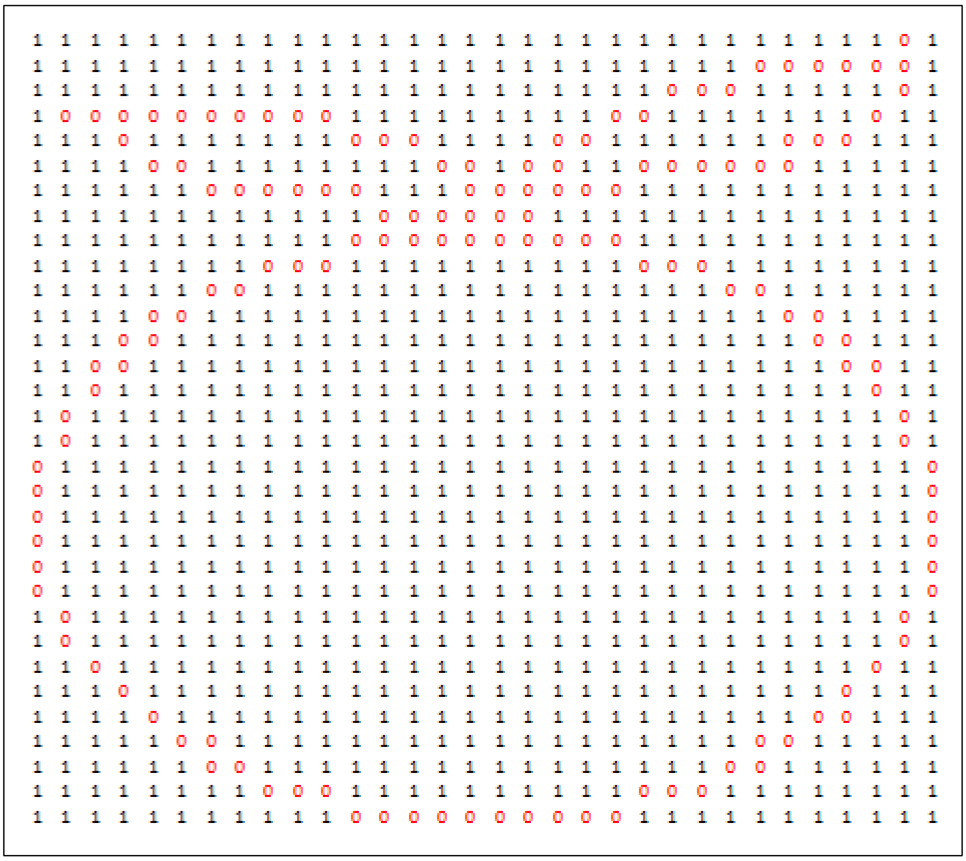
(b)

Figure 4.14: Layer 3's pattern (a) and Layer 4's pattern (b) for the reference image at the first sequence

4.2.2. Feature Extraction of a Tested Image

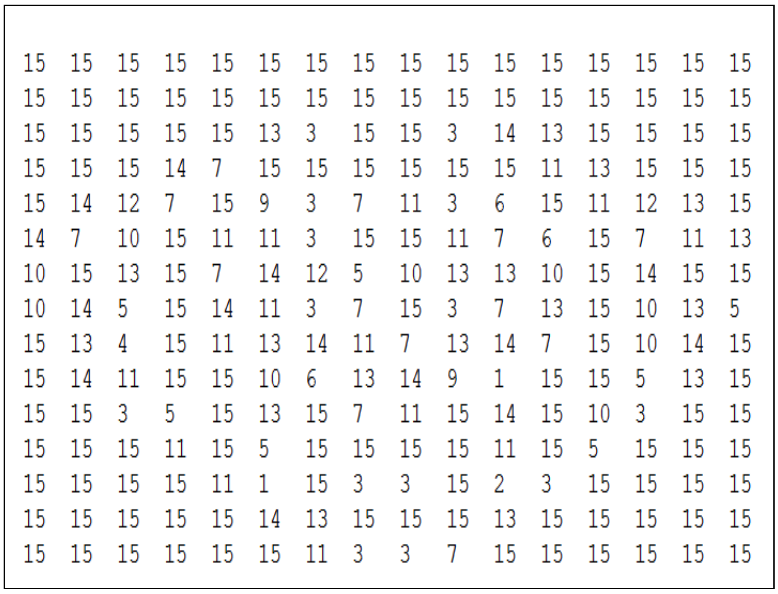
After extracting all the nodes’ features from the reference image, the feature extraction process continues on the tested image. An important part of feature extraction using HCN is that the tested image does not need to create the activation group; instead, it uses the activation group of the reference image to activate its features. Other processes such as scanning and node grouping are similar.

Using the fruit image in Figure 4.2 as the tested image, Figures 4.15 to 4.18 show its patterns at Layers 1, 2, 3, and 4 after conducting the scanning and grouping process at the first sequence.



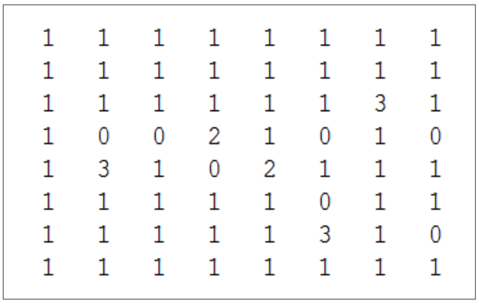
```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 0 1
1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 0 1 1
1 1 1 0 1 1 1 1 1 1 1 0 0 0 1 1 1 1 0 0 1 1 1 1 1 1 0 0 0 1 1 1
1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 0 1 0 0 1 1 0 0 0 0 0 0 1 1 1 1 1
1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1
1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1
1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1
1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1
1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1
1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1
1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1
1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
```

Figure 4.15: Binary features of the tested image



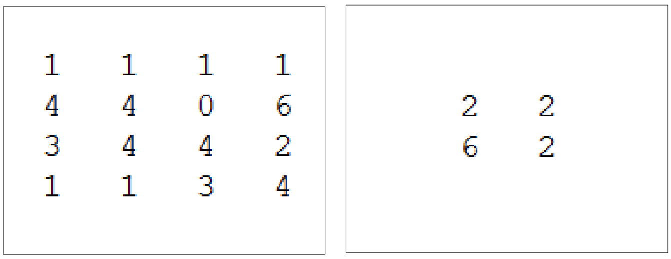
```
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
15  15  15  15  15  15  15  15  15  15  15  15  15  15  15  15
15  15  15  15  15  13  3   15  15  3   14  13  15  15  15  15
15  15  15  14  7   15  15  15  15  15  15  11  13  15  15  15
15  14  12  7   15  9   3   7   11  3   6   15  11  12  13  15
14  7   10  15  11  11  3   15  15  11  7   6   15  7   11  13
10  15  13  15  7   14  12  5   10  13  13  10  15  14  15  15
10  14  5   15  14  11  3   7   15  3   7   13  15  10  13  5
15  13  4   15  11  13  14  11  7   13  14  7   15  10  14  15
15  14  11  15  15  10  6   13  14  9   1   15  15  5   13  15
15  15  3   5   15  13  15  7   11  15  14  15  10  3   15  15
15  15  15  11  15  5   15  15  15  15  11  15  5   15  15  15
15  15  15  15  11  1   15  3   3   15  2   3   15  15  15  15
15  15  15  15  15  14  13  15  15  15  13  15  15  15  15  15
15  15  15  15  15  15  11  3   3   7   15  15  15  15  15  15
```

Figure 4.16: Layer 1's pattern in the tested image at the first sequence



```
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
1   1   1   1   1   1   3   1
1   0   0   2   1   0   1   0
1   3   1   0   2   1   1   1
1   1   1   1   1   0   1   1
1   1   1   1   1   3   1   0
1   1   1   1   1   1   1   1
```

Figure 4.17: Layer 2's pattern in the tested image at the first sequence



```
1   1   1   1
4   4   0   6
3   4   4   2
1   1   3   4
```

(a)

```
2   2
6   2
```

(b)

Figure 4.18: Layer 3's pattern (a) and Layer 4's pattern (b) in the tested image at the first sequence

4.2.3. Similarity Measurement of Reference and Tested Images

As mentioned in Subsection 4.1.4, there are three steps that must take place to measure the similarity between the reference image and the tested image. This calculation uses Equations 4.3, 4.4, and 4.5.

Figures 11, 13, 14, 16, 17, and 18 are examples of features within nodes at the first sequence. By shifting the input image as many the patterns' width, each node has 32 features (image with 32x32 pixels). Similar features are then reduced so that the node has a group of unique features.

Using Equation 4.3, the ratio of similarity of nodes between the reference pattern and the tested pattern can be calculated. Table 4.14 shows the similarity of each node (Nodes 1 to 256) in Layer 1. The first node is located on the first row and in the first column, and the last node is on the last row and in the last column.

Table 4.14: Similarity of nodes between reference image and tested image at Layer 1

Node similarity in Layer 1															
25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	33.3	100.0	100.0
14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.3	16.7	25.0	25.0	25.0	50.0	100.0
42.9	42.9	42.9	28.6	28.6	28.6	14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.3	14.3
25.0	28.6	28.6	28.6	28.6	28.6	28.6	28.6	28.6	28.6	14.3	14.3	14.3	14.3	14.3	14.3
33.3	33.3	33.3	33.3	33.3	22.2	11.1	11.1	11.1	11.1	11.1	11.1	11.1	11.1	11.1	11.1
30.0	30.0	30.0	30.0	30.0	40.0	40.0	40.0	40.0	40.0	40.0	30.0	30.0	20.0	30.0	30.0
40.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0	40.0	40.0	40.0	40.0
40.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0	40.0	30.0	30.0	30.0	30.0	50.0	50.0	50.0
27.3	27.3	27.3	36.4	45.5	45.5	36.4	36.4	54.5	54.5	45.5	45.5	45.5	45.5	45.5	45.5
41.7	41.7	33.3	33.3	25.0	25.0	41.7	41.7	33.3	33.3	33.3	33.3	33.3	33.3	33.3	33.3
40.0	40.0	40.0	40.0	40.0	30.0	10.0	10.0	20.0	20.0	20.0	30.0	30.0	30.0	30.0	30.0
40.0	40.0	40.0	20.0	20.0	20.0	20.0	20.0	20.0	40.0	40.0	40.0	40.0	28.6	28.6	28.6
16.7	16.7	16.7	16.7	16.7	16.7	16.7	16.7	16.7	16.7	16.7	50.0	66.7	66.7	66.7	66.7
25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0
25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0
100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

Using Equation 4.4, the similarity percentage in Layer 1 *(sim_Layer$_1$)* is 35.18%. Layer 2 shows a 56.68% similarity, which is detailed in Table 4.15.

Table 4.15: Similarity of nodes between reference image and tested image in Layer 2

Node similarity in Layer 2							
25.0	25.0	25.0	25.0	25.0	25.0	25.0	50.0
33.3	33.3	33.3	33.3	33.3	33.3	33.3	100.0
25.0	25.0	40.0	50.0	50.0	50.0	50.0	40.0
25.0	25.0	25.0	25.0	25.0	25.0	25.0	25.0
50.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0
50.0	66.7	66.7	100.0	25.0	25.0	25.0	25.0
100.0	100.0	100.0	100.0	100.0	100.0	50.0	50.0
100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

The similarity of nodes in Layers 3 and 4 is shown in Tables 4.16 and 4.17, respectively.

Table 4.16: Similarity of nodes between reference image and tested image at Layer 3

Node similarity in Layer 3			
25.0	25.0	25.0	33.3
0.0	33.3	33.3	33.3
33.3	40.0	28.6	28.6
100.0	100.0	100.0	40.0

Table 4.17: Similarity of nodes between reference image and tested image in Layer 4

Node similarity in Layer 4	
0.0	33.3
40.0	33.3

The similarity percentage between the image of the monkey’s head and the image of the fruit is 42.43% in Layer 3 and 26.67% in Layer 4. This shows that HCN is discriminative towards the higher layers compared to the lower layers.

The similarity rate in Layer 2 increases by slightly more than 1.5 times. From Layer 2, the similarity rate goes down by about 15% in Layer 3 and by double in Layer 4. Therefore, after taking an average of the layers’ similarity using Equation 4.5, the similarity percentage between the two images is 39.24%. The similarity rate between the sample images is different when the position of the reference image and the tested image is interchanged. After interchanging the position of the reference pattern and the tested pattern to extract the features, the similarity rate is 64.27%. This difference is discussed in Chapter 5.

4.3. Feature Extraction Test with Simple Images

A preliminary work using a hierarchical concatenation network (HCN) is tested in (Ramli & Ortega-Sanchez, 2015) using simple images as shown in Figure 4.19.

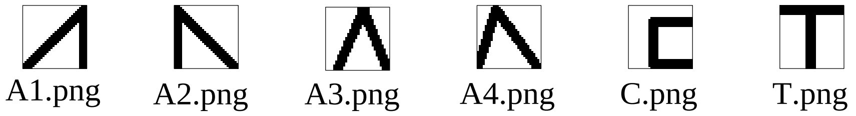
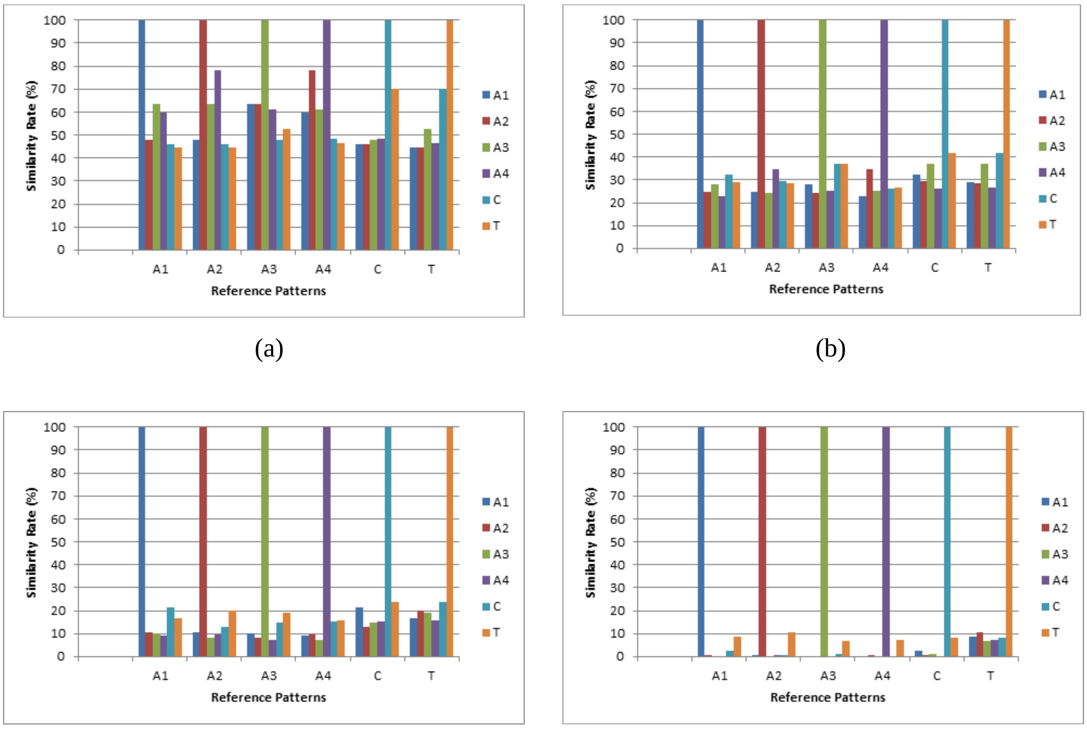


Figure 4.19: Six simple patterns to test using a HCN

The HCN used in this experiment has only two layers. The work investigates the difference between the coincidence array movement using overlap and no overlap during the scanning process. The movement of the coincidence array using overlap means that the coincidence array moves across the pattern's area one pixel at a time. Without overlap means that the coincidence array moves across the pattern by the size of itself, and this method of movement is used in the HCN (as explained at the beginning of this chapter) because it produces better results than the overlap method.

This subsection presents the use of a four-layer HCN to test the simple patterns in Figure 4.19. Using these images as the reference and tested pattern, the similarity rates between them are as shown by the graphs in Figure 4.20.



(a)

(b)

(c)

(d)

Figure 4.20: Similarity rate in (a) Layer 1, (b) Layer 2, (c) Layer 3, and (d) Layer 4

Within all layers, the similarity rate between the reference image and itself as the tested image is 100%, while the rates goes down significantly from the first layer to the top layer between other pairs of reference and tested images.

4.4. Classification with HCN

This subsection provides a brief overview of how classification might be conducted. The ability to recognise given objects or patterns is the ability to classify them into existing groups that have some similarity of features. Humans and mammals might show this ability in either a supervised or unsupervised learning process. A supervised learning process is when the subject requires constant feedback while the learning is taking place. For example, when kids first start to learn their numbers, they must be supervised. Parents may say the name of a number out loud and also show its shape. After several repetitions, kids are able to recognise numbers based on their name and shape.

Unsupervised learning is undertaken alone. Let’s use the example of a cat that can recognise the difference between milk and water. The cat’s owner has never taught him to understand which one is which, but the cat can recognise the difference. The cat does not understand the meaning of milk or water from a human point of view, but she can understand the difference of colour, smell, and taste.

The two examples above show that both supervised and unsupervised learning can help intelligent creatures recognise objects. These learning processes have been studied for decades by the artificial intelligence community (C. H. Lee & Yang, 2007; C. S. G. Lee & Lin, 1992; Manohar, Kumar, & Kumar, 2016; Sapkal, Kakarwal, & Revankar, 2007; Wosiak, Zamecznik, & Niewiadomska-Jarosik, 2016). As presented in Chapter 2, researchers have tried to develop machine algorithms that replicate intelligent behaviour.

Similarity measurement is the basic process to quantify the difference between a given pattern and patterns that were previously learned. The general process of similarity measurement involves comparing all features that form the whole pattern. The pattern’s features can be lines, curves, dots, etc. As per the discussion at the beginning of this chapter, intelligent creatures might recognise objects by identifying and concatenating smaller features. Figure 4.2 illustrates how easy it is to identify two images as a monkey’s head based on their features.

Once the recognition process is complete, the developed method must prove its ability to perform classification. Figure 4.6 shows the classification of objects based on their shape. Let's assume the classification method needs a supervisor to label the class of each object. When a new object is presented, its features are compared to the features within each labelled class. The shape that has the closest similarity of features to a specific class is assigned to that class.

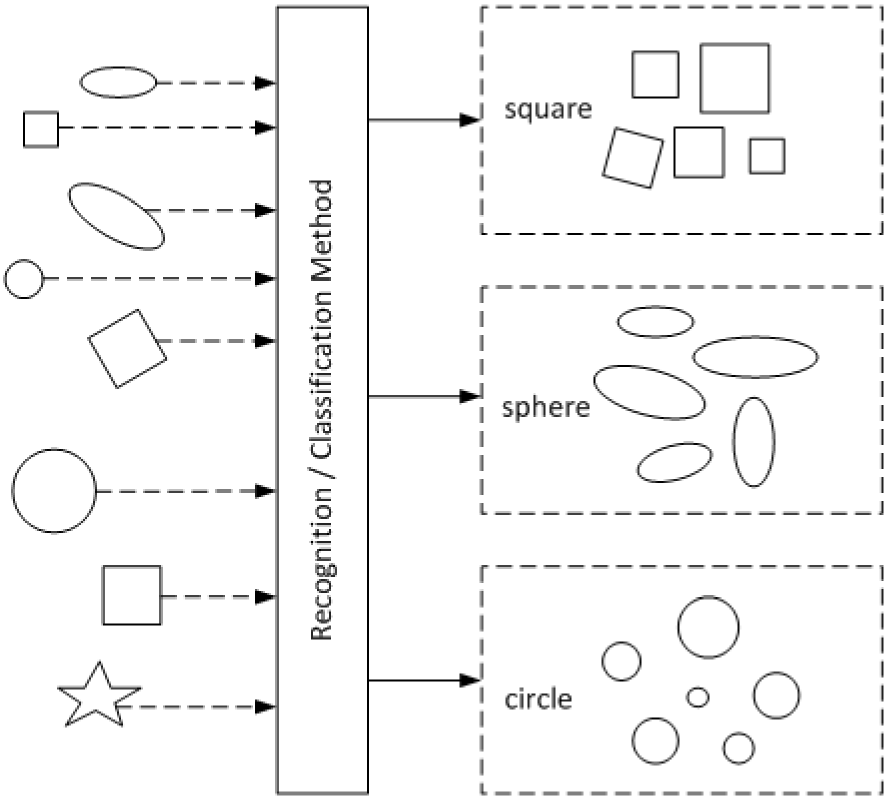


Figure 4.21: Pattern classification based on geometric similarity

What happens to the star shape? It does not have any class. This means the classification method has never learnt the star shape. Even though the star shape has no class, the method still produces a similarity measurement. The star shape has a small part of the square's lines that can assign it to the class of square with some small value of similarity.

There three two ways to classify objects or images into available classes in this thesis. The first is called classification by union. Classification by union replicates the process that is explained at the beginning of this subsection. Firstly, only the unique features of the group's images are retained, and then a given image's features are compared to the unique features of that group. When a given image shows the highest number of similar features to a particular group, it will be assigned to that group. The second method measures the similarity of the tested image’s features against the average of similarity of features of each class. The class with the closest similarity to the given image is chosen for that image. This classification method is called classification by average. The last investigation has been conducted based on the distance measures. Further investigation and exploration into the way the tested pattern is classified into a specific class using HCN is presented in Chapter 6. Feature extraction is implemented in Chapter 5.

Chapter 5
Feature Extraction using Hierarchical Concatenation Networks

Chapter 4 discusses the general concept of the hierarchical concatenation method to extract an image's features and its similarity measurements, node by node and layer by layer. By increasing the coverage area of the coincidence array from layer to layer, the similarity rate between the reference pattern and tested pattern becomes discriminative (Ramli & Ortega-Sanchez, 2015). This indicates that the extracted features in the lower layers have a higher similarity rate than the higher layers, because the lower layers perceive smaller features (compared to the higher layers). In the highest layer, features are represented in their largest form, which causes the perception of the features in the different objects to be discriminative. Similarity is measured by calculating the average similarity rate within all layers.

This thesis explores feature extraction and patterns activation within the network's layers. There are two different ways to activate the representation of the upper layers' patterns, and these are discussed in this chapter. These two feature extraction methods are called HCN-I and HCN-II.

After presenting the possibility of using hierarchical concatenation to extract features in the previous chapter, this thesis now investigates the designed network's ability to recognise objects or patterns. As the recognition process of the way how humans see is difficult to understand, attempting to replicate it as a computer algorithm requires specimens to test the algorithm's ability. The specimens should represent identified objects or patterns from a human's point of view. If the specimens exist and are familiar in the world, the performance of the computer algorithm can be clearly compared to the way humans see.

There are many complex datasets with high-level distortion, but those kinds of dataset need pre-processing before feeding to the specific algorithm. The proposed algorithm in this thesis excludes complex pre-processing as it is supposed to proof the ability of the proposed algorithm to perform recognition or classification. The experiments in this thesis used two kinds of datasets. The first is 'digit dataset' that was produced using freeware and the second is publicly available handwritten (USPS and MNIST) datasets. These datasets were used in the experiments due to: they are easy for human to identify them compared with others that could lead misinterpretation, the proposed algorithm was designed to perform the concatenation process during feature combining, and the proposed algorithm needs to show its ability in recognition by using the specimens which are familiar to human. Other similar datasets with high-level distortion for example high degrees rotated shaped or other kinds of noisy images were not used in this thesis.

The 'digit dataset' is chosen in the experiment as it is easy to investigate if the proposed algorithm is able to achieve the objective of recognition or not. For this reason the dataset is produced using free software called 'imagemagick', which can be downloaded from www.imagemagick.com. With this software the less distorted images can be produces, for example the images can have has different style (font, thickness, and tilt). The images are drawn on 32 x 32 pixels of canvas with no noise within the canvas.

For the purpose of this investigation, each class of numbers contains 10 members. This is because the human can accurately recognise shapes with different thicknesses, positions, sizes, and angles. This thesis seeks to reproduce this ability by pre-processing the inputs before applying the designed algorithm. This chapter presents the HCN algorithm that must to prove its ability to follow the concept of features recognition as presented in Chapter 4.

In Chapter 4, the features of a monkey's head are used to extract a fruit's features. The pattern that is used to extract the other pattern is called the reference pattern, while the extracted pattern is called the tested pattern. In this example, the image of the monkey's head is the reference pattern, and the image of the fruit is the tested pattern.

Feature extraction using HCN extracts the features of a pattern using another pattern's features. The extraction process is conducted hierarchically from the lowest to the highest layer. On the lowest layer, the smallest extracted features are the pixels itself that represented in binary numbers. Features on upper layers are the concatenation of features from the below layer. On the other word, the combination of active features at the lower layer represent the upper layers' features. Within the earlier experiment as discussed in (Ramli & Ortega-Sanchez, 2015, 2016), when a feature of the tested pattern is similar to a feature of the reference pattern within the correspond node, the feature of upper layer that represented by assigning value '1' on the correspond bit. After several investigation this method represented the same feature for different comparisons. For example, the first comparison between a tested pattern *tp* that represented by vector *[1 2 3 4]* and a reference pattern *rp = [1 2 3 4]* produce a new feature on upper layer represented by vector *[1 1 1 1]*. The elements of that vector is assigned by '1' due to each feature of the tested pattern is similar to the feature of the reference pattern. Let us see the second comparison. If a tested pattern has *[2 3 5 6]* and a reference pattern has also have *[2 3 5 6]* then a feature on upper layer will be *[1 1 1 1]*. Even though this method is still able used in recognition or classification process but it influence the results and time consumption to run the algorithm.

After several experiments, the second method was defined as another way of upper layer's feature activation. In this method, the occurrence of a feature was considered to play important role to activate upper layers' features. By using the first comparison above, each reference pattern has one time of occurrences. By having the same feature between the tested and reference pattern, the upper layer's feature will be *[1*1 + 2*1 + 3*1 + 4*1]* which is '10'. Therefore, the upper layer's feature of the second comparison above will be *[2*1 + 3*1 + 5*1 + 6*1]* which is equal to '16'. With this second method, two different pattern comparisons show two different feature representations on upper layer. The first method in this thesis is then called as HCN-I and the second method is HCN-II. The detail algorithm of both are presented in sub section 5.1.

In other investigation of the feature activation process, it can be seen that the way the HCN extracts features makes the similarity rate between the two patterns different when their positions as the reference and tested image are interchanged. This is an important find in this research and will be discussed further in this chapter.

The remaining section in this chapter will present:

1. the HCN-I and HCN-II feature extraction methods,
2. the concept of ‘many to one’ and ‘one to many’,
3. the results of an investigation into the different feature extraction methods, using sequential and non-sequential scanning, and the possibility of involving the middle area of the image.

5.1. HCN-I and HCN-II Feature Extraction

HCN-I and HCN-II apply the same feature activation method to Layer 1, because this layer is directly connected to the input (pre-processed input). They are different on upper layers. The HCN-I activates the feature by giving value ‘1’ if the coincidence array sees the feature in it is similar to the feature of reference pattern, while the HCN-II activates the feature by multiplying the numbers of seen feature by the coincidence array with its weight.

Figure 5.1 shows an example of HCN-I feature extraction and activation in Layers 1 and 2.

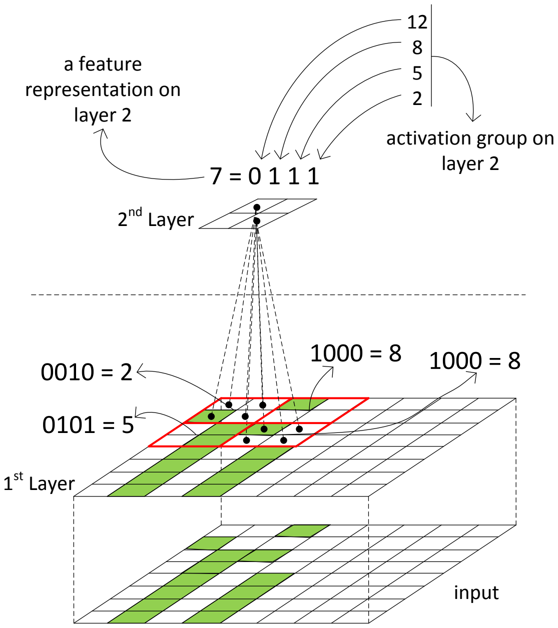


Figure 5.1: Feature extraction and activation in HCN-I

As Layer 1 is connected to the input (pre-processed input), its nodes contain the concatenation of four bits of the input. For example, Layer 1's first node in Figure 5.1 is the concatenation of binary '0010', which is represented as a '2' in decimal value. This means that the first node in Layer 1 has a feature represented as '2'. All the features in Layer 1's nodes are used to activate the features in Layer 2. The activation process for the features in each node requires activation grouping (as discussed in Chapter 4.1.2.2). In this example, there are only three active bits in the activation group ('0111'). This vector represents a feature in the first node of Layer 1 as decimal value '7'. If the input is similar to the reference pattern, all bits of the activation group are active. The first node in Layer 2 is '1111', or '15' as a decimal value.

In the above example, the first node of Layer 2 is '7', which is different to the reference pattern's first node in Layer 2. Hence, the similarity between the first node of input and the reference pattern can be measured by ratio or distance. The measurement using ratio gives the similarity rate (7/15)x100% = 46.67%, while the distance measurement is $\sqrt{(7-15)^2}$ = 8.

The HCN-II has a different activation process. It uses weight to represent the upper layers' features. Figure 5.2 illustrates feature extraction and activation in Layers 1 and 2 of HCN-II.

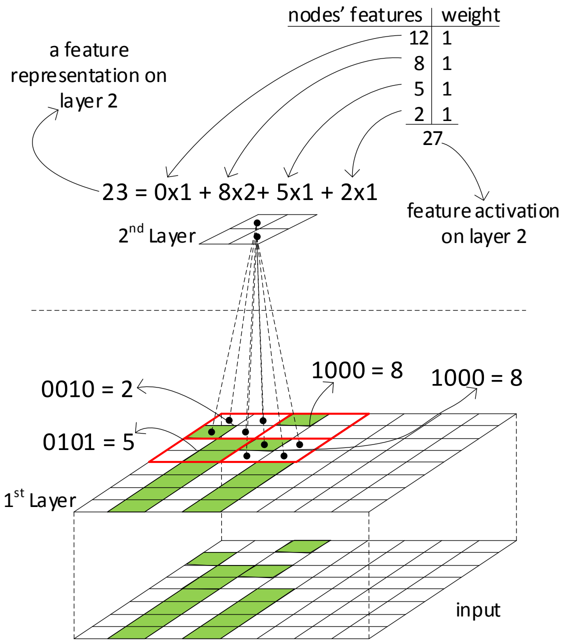


Figure 5.2: Feature extraction and activation in HCN-II

The activation group (or activation feature) in HCN-II can be calculated by multiplying the number of the feature itself with the number of its occurrences within the concatenated node. For example, in Figure 5.2, the activation group is '27', which comes from '12x1'+'8x1'+'5x1'+'2x1', and the input has a feature of '23' in the first node on Layer 2. Hence, similarity by ratio is (23/27)x100% = 85.18%, and distance is $\sqrt{(23-27)^2}$ = 16.

The above example calculations of similarity and distance show that the difference between HCN-I and HCN-II is 38.51% in similarity and 8 in distance. Therefore, the results of both HCN methods are presented in this chapter, and Chapter 6 explores their classification abilities with these methods.

5.2. 'Many To One' And 'One To Many' Methods, Sequential and Non-Sequential Scanning, and Overlapping the Middle Area of Images

The difference in the rate of similarity when the position of the reference and tested image is interchanged is the idea behind the 'many to one' and 'one to many' methods. This section presents their differences.

In this thesis, feature extraction using hierarchical concatenation is used to separate the attributes of a second pattern based on the features of the first, or vice versa, and is shown in Figure 5.3.

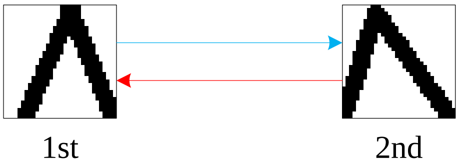


Figure 5.3: An example of two patterns that can extract each other

From a human point of view, the patterns will seem similar when comparing the first pattern to the second pattern, and vice versa. The HCN, however, produces different results when the reference pattern and tested pattern are interchanged. In Figure 5.3, the blue line represents the first pattern being used as reference, while the red line represents the second pattern being used as reference. Following the direction of blue line, the HCN algorithm produces a similarity rate between these two patterns of 45.24%. This rate decreases to 30.95% when the second pattern is used as reference (following the direction of the red line).

Based on this difference, the position of the reference patterns when extracting the tested patterns, or vice versa, plays an important role. Further investigation using a dataset is conducted in this thesis to study the difference between the two positions. Figure 5.4 illustrates the interchanging positions between a set of reference and tested patterns.

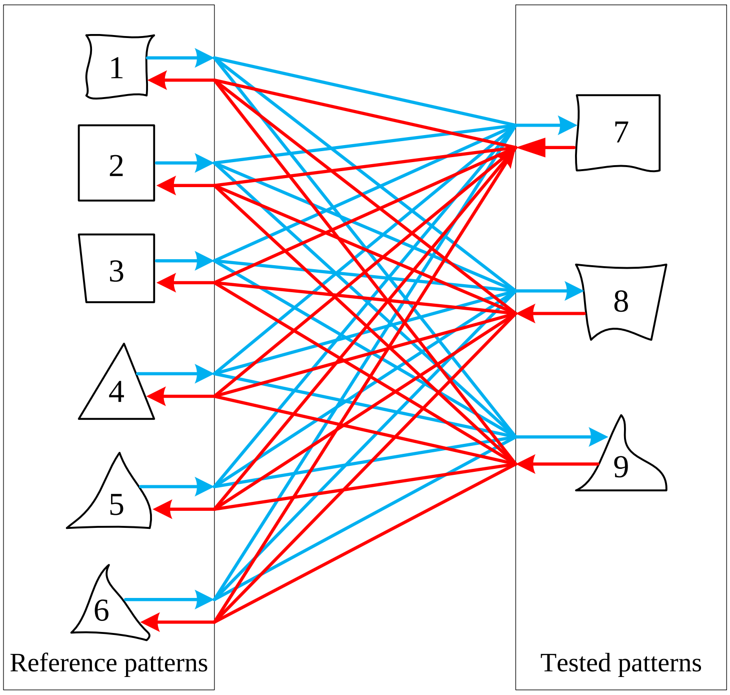


Figure 5.4: Representation of exchanging positions between reference and tested patterns where blue and red lines indicate source of extraction

Figure 5.4 shows six patterns on the left-hand side and three patterns on the right-hand side. Let's assume that the left-hand side patterns are the reference patterns, and the right-hand side patterns are the tested patterns. The blue lines in this figure represent the feature extraction process from the reference patterns' point of view, meaning feature extraction is based on reference patterns. This method is called 'many to one'. In this method, each reference pattern extracts the tested patterns' features using the features of the reference pattern. The reverse process is represented by the red lines, which indicate the activation process from the tested patterns' point of view. This is called the 'one to many' method, and in this method, the tested patterns extract the features of the reference patterns based on the tested patterns' features.

Using the 'many to one' method, the six patterns on the left-hand side see all tested patterns based on their features. For example, Pattern 1 extracts the tested pattern by searching for its vertical and horizontal lines (its features) within the first tested pattern. If Pattern 1 sees similar features in the tested pattern (1), those features are then activated within the tested pattern (see Section 4.1.3 in Chapter 4). Correspondingly, the tested pattern (1) has its own features based on the reference pattern (1). These two patterns are then compared to calculate the similarity rate between them. Patterns 2 to 6 follow the same process, and the similarity rates for each pair of patterns (1–7 to 6–9) are shown in Table 5.1. In the ‘one to many’ method, on the other hand, the three patterns on the right-hand side use their own features to activate the six reference patterns. The similarity rates for each pair of patterns (7–1 to 9–6) are also shown in Table 5.1.

Table 5.1: Similarity rates when position of reference pattern and tested pattern is interchanged

Reference Patterns Extract Tested Patterns	Similarity Rate (%)	Tested Patterns Extract Reference Patterns	Similarity Rate (%)
1 – 7	32.53	7 – 1	47.40
1 – 8	39.13	7 – 2	45.45
1 – 9	30.40	7 – 3	45.24
2 – 7	76.23	7 – 4	36.78
2 – 8	57.38	7 – 5	39.38
2 – 9	50.19	7 – 6	38.81
3 – 7	51.17	8 – 1	39.00
3 – 8	44.12	8 – 2	32.19
3 – 9	43.44	8 – 3	31.20
4 – 7	31.11	8 – 4	35.80
4 – 8	29.25	8 – 5	33.59
4 – 9	32.22	8 – 6	29.17
5 – 7	33.46	9 – 1	22.25
5 – 8	32.89	9 – 2	24.10
5 – 9	27.66	9 – 3	24.88
6 – 7	25.43	9 – 4	31.94
6 – 8	24.36	9 – 5	31.31
6 – 9	27.80	9 – 6	28.19

In Table 5.1, each pair of patterns, 6–8 and 8–6 (highlighted in light green), show that the similarity rate between them is different when their position to extract the features is exchanged. If Pattern ‘6’ uses its own features to extract the features of Pattern ‘8’, the similarity rate between them is 24.36%. On the other hand, when Pattern ‘8’ extracts the feature of Pattern ‘6’, the similarity rate is 29.17%. Another example in Table 5.1, which shows a difference in similarity rates when positions are interchanged, is highlighted in light blue. In this example, the left-hand side pattern (‘6’) uses its features to extract the features of Pattern ‘9’, and on the right-hand side, the features of Pattern ‘9’ are used to extract the features of Pattern ‘6’. This difference in the similarity rate occurs because the features of the extracted pattern are activated based on the features of the extracting pattern (see Section 4.1.3 in Chapter 4).

Another illustration of using the ‘many to one’ and ‘one to many’ methods to extract features is as follows:

Consider a person arriving at a place where some people are wearing clothes of different colours (see Figure 5.5). A decision must be made as to which person he should join, based on the colour of the clothes he himself is wearing.

From the point of view of the person who has just arrived, his group partner is to be chosen based on the clothes they are wearing. This is the ‘one to many’ concept. This person examines each single part of his clothes and compares them against the clothes of every other person’s clothes in the room. Calculating the highest similarity rate between his clothes (shirt, pants and shoes) and the clothes of others will allow him to decide which group partner to join.

The people already in the room must draw the attention of the newcomer based on their costumes’ similarity. In Figure 5.5, one of the people in the room asks the new person to join him because their pants are similar, while others say similar things based on the rate of similarity between their clothes. This is the ‘many to one’ concept. Using the ‘many to one’ method, the decision is made by the person in the room who has the highest rate of similarity to the newcomer’s clothes.

Both of these methods actually perform the same feature extraction process: scanning, grouping, and activation, as described in Chapter 4, but the reason this thesis must explore the methods further is that they produce different results (as presented in Table 5.1). The differences between these two methods are presented by the results in the next sections.

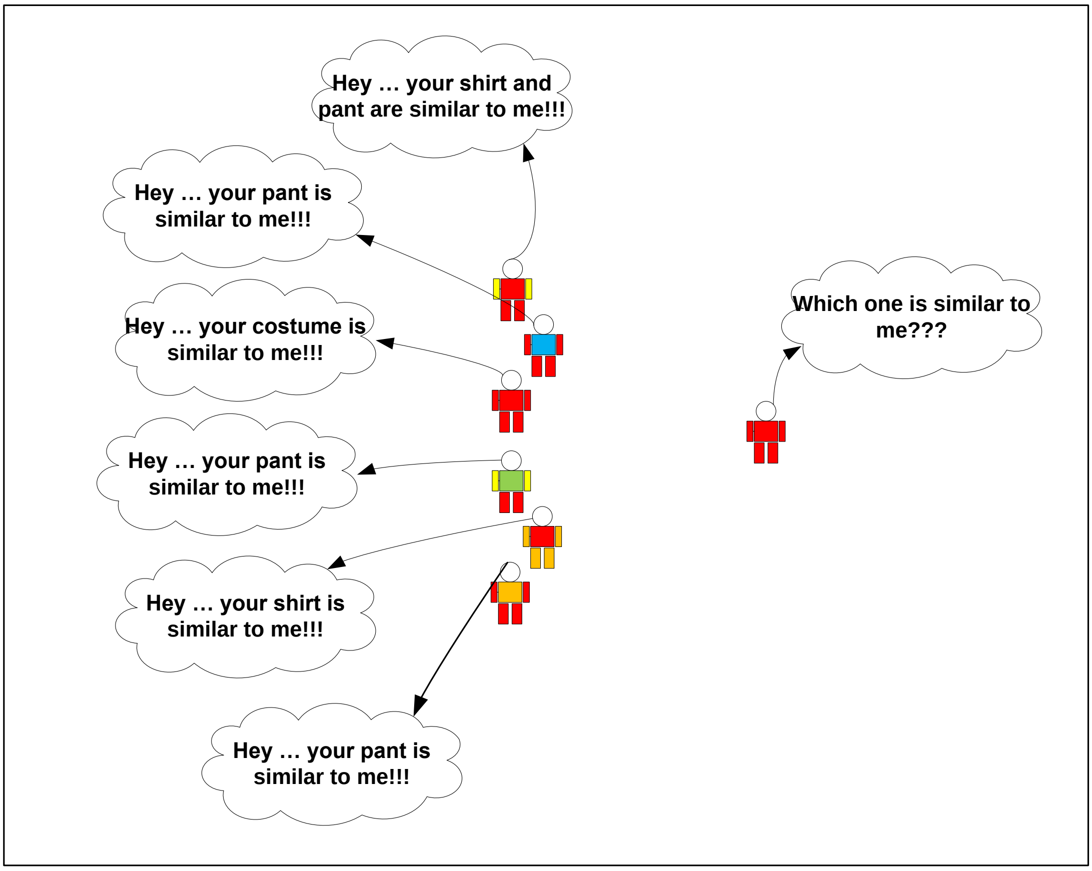

Hey … your shirt and pant are similar to me!!!
Hey … your pant is similar to me!!!
Hey … your costume is similar to me!!!
Hey … your pant is similar to me!!!
Hey … your shirt is similar to me!!!
Hey … your pant is similar to me!!!
Which one is similar to me???

Figure 5.5: ‘Many to one’ and ‘one to many’ concept

Using the digits dataset, which is meaningful from a human perspective, this chapter presents some experiments. The experiments implement sequential scanning (as presented in Chapter 4) as well as non-sequential scanning, and the results are investigated further by involving the middle area of the image during the scanning process.

The concept of sequential scanning is discussed in Chapter 4. Implementation using non-sequential scanning does not shift the image during scanning. A case where overlapping is used in the middle area of the image is also considered so that any improvement in performance rate can be assessed. The algorithm presented in Chapter 4 uses the whole image area in its early implementation of hierarchical concatenation. In the following example, overlapping the middle area means that feature extraction is also performed within the middle area of the image, as presented in Figure 5.6.

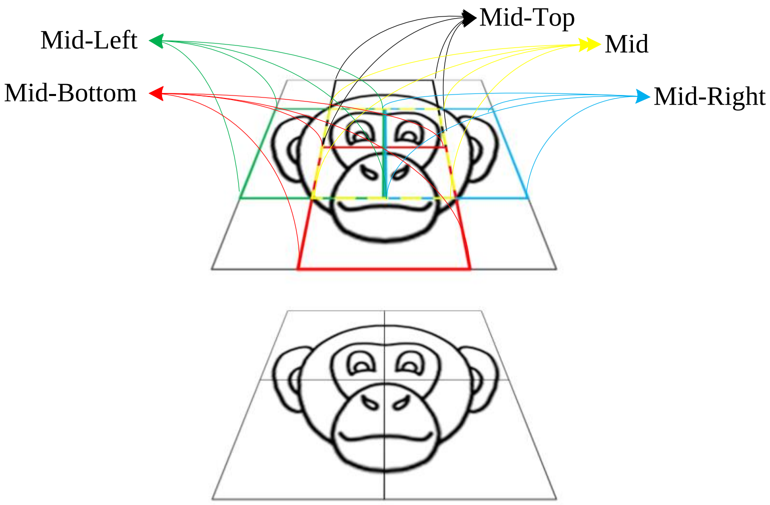


Figure 5.6: Overlapping the middle area of image

The image at the bottom of Figure 5.6 shows the scanning area of the image with no overlapping. The horizontal and vertical lines in the middle of the image indicate the borders of the scanning area. The coincidence array does not cover the pixels on those lines. The image at the top of Figure 5.4 illustrates the overlapping area in the middle of image. This overlap produces five divisions, which are labelled as ‘mid-centre’, ‘mid-top’, ‘mid-left’, ‘mid-bottom’, and ‘mid-right’. The ‘mid-right’ division covers the image area indicated by the yellow arrows. The black arrows represent the middle area covered by the ‘mid-top’ division, and the ‘mid-left’ location is shown by the green arrows. The ‘mid-bottom’ and ‘mid-right’ divisions are the areas indicated by the red and blue arrows, respectively. Each division has one quarter of the image size, or 16x16 pixels.

The different results — with and without overlapping the middle part of the images — are explored in the next subsections. The investigation uses both numbers and letters for the purpose of recognition. The experiments are setup to test the HCN’s ability to extract features using both the ‘many to one’ method and the ‘one to many’ method. The capability of these methods to recognise the tested patterns is also investigated by implementing overlapping in the middle area of the pattern. If a tested pattern is supposed to have a higher similarity rate with its own group, the failure of the HCN method to recognise or place the tested patterns into the correct group is measured as the basis for its performance.

Classification is another application of the hierarchical concatenation concept presented in this thesis. Classification is the ability of the network to classify patterns into groups based on hierarchical feature extraction, and this is presented in Chapter 6.

5.3. The Exploration of Recognition

To explore the concept of the hierarchical concatenation network (HCN), a dataset is produced using ‘imagemagick’ and used during the experiment (Curtin, 2013). This thesis investigates the results produced by the HCN when recognising members from each set of digits, and then determines its ability to place patterns from the same class into the highest similarity rate.

Table 5.2 shows 10 groups of digits, where each digit ranges from ‘0’ to ‘9’. Each group contains 10 members from the dataset of numbers and is produced using Algorithm 5.1.

Algorithm 5.1: Producing Image

```
For digit from 0 to 9
  For i from 1 to 10
      Font ←fontlist.lib
      Convert size into 32x32 xc:white  -font(i) -pointsize 28 fill black -gravity center -draw tmp.png
      Convert tmp.png –fuzz 0% -trim +repage tmp2.png
      Convert tmp2.png –resize 32x32! tmp3.png
      Convert tmp3.png digits_i.png
  End for i
End for digit
```

The digit dataset has 10 classes from 0 to 9, and each class has 10 members that are represented in the second row as 1 to 10. The image is drawn in black within a white background measuring 32x32 square pixels. The font style is chosen from the font library with the font size of 28.

Table 5.2: Ten groups of digits

Member	Groups									
	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	0	1	2	3	4	5	6	7	8	9
2	0	1	2	3	4	5	6	7	8	9
3	0	1	2	3	4	5	6	7	8	9
4	0	1	2	3	4	5	6	7	8	9
5	0	1	2	3	4	5	6	7	8	9
6	0	1	2	3	4	5	6	7	8	9
7	0	1	2	3	4	5	6	7	8	9
8	0	1	2	3	4	5	6	7	8	9
9	0	1	2	3	4	5	6	7	8	9

In the discussion that follows, the patterns in Table 5.2 are identified using their digit name followed by their group number. For example, the first ‘0’ in Group 0 is identified as ‘0_0’, while the fifth ‘3’ in Group 3 is identified as ‘3_5’.

The following section presents the experiments’ results within the following frameworks:

- Similarity rate between a given tested pattern and the whole dataset.
- Sorting the position of the datasets based on their similarity to the tested pattern and in a descending order.

HCN feature activation (HCN-I and HCN-II) is discussed for each of the following methods:

1. The ‘many to one’ method using:
    a. non-sequential and sequential scanning over the whole image area, and
    b. an overlapping middle area during feature extraction.
2. The ‘one to many’ method using:
    a. non-sequential and sequential scanning over the whole image area, and
    b. an overlapping middle area during feature extraction.

5.3.1. Recognition using the ‘Many to One’ Method

A brief description of the ‘many to one’ method is provided at the beginning of this chapter. The first experiment implements a hierarchical concatenation feature extraction without sequential scanning. This is conducted by setting the number of the sequence *(nSeq)* as ‘1’ (see Algorithm 4.1 in Chapter 4).

Table 5.3. shows a comparison of the first 10 digits with the highest similarity rate between the tested and reference patterns. The first 10 sets of 10 digits (100 numbers) are then compared against each one of the tested patterns. The rows in Table 5.3 show, for each tested pattern, the 10 digits with the highest similarity rate as calculated by the algorithm using non-sequential scanning. Feature activation in the upper layer uses the HCN-I method.

Table 5.3: ‘Many to one’ method with no sequential scanning in HCN-I.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	**0**	1	**0**	**0**	1	6	**0**	**0**	**0**	7
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	7	4
2	**2**	1	1	1	**2**	**2**	9	1	1	7
3	**3**	1	1	1	7	1	**3**	7	0	0
4	**4**	**4**	1	1	1	1	**4**	1	7	1
5	**5**	**5**	1	1	**5**	0	4	0	4	8
6	**6**	4	5	0	1	1	1	0	1	3
7	**7**	1	1	**7**	**7**	**7**	1	**7**	**7**	1
8	**8**	0	4	1	1	3	4	1	**8**	1
9	**9**	1	1	**9**	0	0	1	0	**9**	7

The digits with blue borders are the numbers that are recognised correctly. As each group of numbers has 10 digits, if the first 10 highest similarities between the reference and tested patterns are from the group of the tested pattern, then the recognition rate of the tested pattern is 100%. In Table 5.3, only 37 out of 100 digits are correctly recognised by the HCN across the whole image area with no shifting of the image patterns during scanning.

The next step involves the middle area of the image during scanning. Overlapping the middle area increases the total number of nodes in each layer. If each middle division measures 16x16 pixels and the number of nodes in each layer is calculated using Equation 4.1 in Chapter 4, 64 more nodes are added to Layer 1 in each division. Layers 2 to 4 are increased by 16, 4, and 1 node, respectively. When combining the entire middle area of the image (five divisions), each layer has five times the number of additional nodes. In this instance, Layers 1 to 4 have 576, 144, 80, and 9 nodes, respectively.

Table 5.4: ‘Many to one’ method involving middle area of pattern with no sequential scanning in HCN-I.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	$\boxed{0}$	1	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	1	$\boxed{0}$
1	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	0	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	7	0
2	$\boxed{2}$	0	1	1	$\boxed{2}$	0	1	0	7	1
3	$\boxed{3}$	0	$\boxed{3}$	1	1	1	0	0	0	7
4	$\boxed{4}$	1	1	$\boxed{4}$	1	1	1	0	1	7
5	$\boxed{5}$	0	1	1	$\boxed{5}$	1	6	$\boxed{5}$	1	0
6	$\boxed{6}$	0	0	1	5	1	$\boxed{6}$	0	$\boxed{6}$	1
7	$\boxed{7}$	$\boxed{7}$	1	1	1	0	1	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$
8	$\boxed{8}$	0	3	$\boxed{8}$	1	1	1	0	1	3
9	$\boxed{9}$	0	1	1	0	1	7	0	$\boxed{9}$	$\boxed{9}$

It can be seen that when including the middle area of the images during scanning to extract features, non-sequential scanning does not give better results (Table 5.4). It actually exhibits a similar performance to when the additional middle area is not used. These results indicate that involving the middle divisions of an image does not improve the HCN’s ability to recognise the tested patterns.

The following investigation implements sequential scanning, which is presented in Chapter 4. Table 5.5 shows the performance of the HCN using the ‘many to one’ method along with sequential scanning.

Table 5.5: ‘Many to one’ method with sequential scanning in HCN-I.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	1	6	0	0	0	0
1	1	1	1	1	1	1	1	1	0	0
2	2	2	2	2	2	2	2	9	2	2
3	3	3	3	3	3	1	0	3	8	3
4	4	1	4	4	1	4	4	1	4	1
5	5	5	1	5	5	0	0	3	5	7
6	6	0	6	5	1	5	6	1	5	6
7	7	7	7	7	7	1	7	1	7	7
8	8	8	8	8	3	3	0	8	3	9
9	9	0	9	9	1	9	9	7	0	9

When implementing sequential scanning for feature extraction of the tested patterns based on the reference patterns, recognition performance almost doubled to 64%. This indicates that sequential scanning, when applied to the ‘many to one’ method, produces better results than non-sequential scanning Although sequential scanning gives a better performance of recognition, the time it takes to run the algorithm is 18 times longer than without sequential scanning.

The following table (Table 5.6) presents the results when the middle area of an image is included within the sequential scanning process.

Table 5.6: ‘Many to one’ method involving middle area of pattern with sequential scanning in HCN-I.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	1	0	0	1	1	1	0
1	1	1	1	1	1	1	0	1	0	1
2	2	2	2	2	2	2	2	2	7	0
3	3	3	0	1	3	3	3	1	3	1
4	4	1	4	0	1	1	4	1	4	1
5	5	1	0	1	5	5	1	0	1	0
6	6	0	6	1	1	6	5	5	5	1
7	7	1	7	7	7	1	7	1	7	7
8	8	0	8	8	8	3	3	8	1	3
9	9	0	9	1	9	9	9	9	1	1

When involving the middle area of the image, the ability of the algorithm to extract a tested pattern’s features and compare their similarity decreases to 55%. This indicates that using the ‘many to one’ feature extraction method in HCN-I has the highest similarity rate for the first 10 reference patterns (at 64%).

Table 5.7 presents the first 10 most similar reference patterns (compared to the tested patterns) using HCN-II feature extraction with the ‘many to one’ method.

Table 5.7: ‘Many to one’ method with no sequential scanning in HCN-II.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	$\boxed{0}$	$\boxed{0}$	6	9	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	6	8	9
1	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	7	$\boxed{1}$	7	7
2	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	9	$\boxed{2}$
3	$\boxed{3}$	$\boxed{3}$	$\boxed{3}$	8	8	$\boxed{3}$	$\boxed{3}$	5	$\boxed{3}$	$\boxed{3}$
4	$\boxed{4}$	$\boxed{4}$	$\boxed{4}$	$\boxed{4}$	$\boxed{4}$	$\boxed{4}$	$\boxed{4}$	1	1	6
5	$\boxed{5}$	3	$\boxed{5}$	8	$\boxed{5}$	6	6	3	8	$\boxed{5}$
6	$\boxed{6}$	5	$\boxed{6}$	5	$\boxed{6}$	$\boxed{6}$	0	3	$\boxed{6}$	$\boxed{6}$
7	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	1	$\boxed{7}$
8	$\boxed{8}$	$\boxed{8}$	3	$\boxed{8}$	$\boxed{8}$	$\boxed{8}$	3	$\boxed{8}$	$\boxed{8}$	$\boxed{8}$
9	$\boxed{9}$	$\boxed{9}$	$\boxed{9}$	0	$\boxed{9}$	$\boxed{9}$	$\boxed{9}$	$\boxed{9}$	2	$\boxed{9}$

By sorting the first 10 highest similarity rates between the reference pattern and the tested pattern, HCN-II using the ‘many to one’ feature extraction method without sequential shifting of the images gives 70 out of 100 digits in the correct place. Let’s see the results in Table 5.8 when the middle area of image is included in the measurement.

Table 5.8: ‘Many to one’ method involving middle area of pattern with no sequential scanning in HCN-II.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	6	9	0	9	6
1	1	1	1	7	7	1	7	7	7	7
2	2	2	2	2	2	9	7	9	9	0
3	3	3	3	3	3	5	5	5	1	5
4	4	4	4	4	7	1	7	7	3	7
5	5	5	5	5	3	6	6	5	3	3
6	6	5	5	6	6	6	5	6	0	8
7	7	7	7	2	7	7	7	7	2	7
8	8	8	8	8	8	3	5	6	6	6
9	9	9	9	2	9	9	0	2	2	8

When involving the middle area of the images, HCN-II does not improve the placement of reference patterns. Only 54% of digits are located close to the tested patterns. This indicates that inclusion of the middle area of the image, which increases feature comparison, does not enhance the similarity rate of patterns within the same group.

Table 5.9 presents the investigation in HCN-II with sequential shifting and when the middle area of the image is not involved.

Table 5.9: ‘Many to one’ method involving middle area of pattern with sequential scanning in HCN-II.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	6	9	0	0	0	6	8	9
1	1	1	1	1	1	1	7	1	7	7
2	2	2	2	2	2	2	2	2	9	2
3	3	3	3	8	8	3	3	5	3	3
4	4	4	4	4	4	4	4	1	1	6
5	5	3	5	8	5	6	6	3	8	5
6	6	5	6	5	6	6	0	3	6	6
7	7	7	7	7	7	7	7	7	1	7
8	8	8	3	8	8	8	3	8	8	8
9	9	9	9	0	9	9	9	9	2	9

The implementation of sequential scanning increases time consumption by 40 times, but the number of correct digits placed close to the tested patterns is 70 out of 100 as shown in Table 5.9. This number is similar to when HCN-II is implemented without sequential scanning. This indicates that HCN-II does not improve the correct placement of the reference patterns when shifting the image by as many pixels as the number of sequences.

Let’s look at the results when the middle area of the image is included during the scanning process along with sequential implementation as shown in Table 5.10.

Table 5.10: ‘Many to one’ method involving middle area of pattern with sequential scanning in HCN-II.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	6	9	0	9	6
1	1	1	1	7	7	1	7	7	7	7
2	2	2	2	2	2	9	7	9	9	0
3	3	3	3	3	3	5	5	5	1	5
4	4	4	4	4	7	1	7	7	3	7
5	5	5	5	5	3	6	6	5	3	3
6	6	5	5	6	6	6	5	6	0	8
7	7	7	7	2	7	7	7	7	2	2
8	8	8	8	8	8	3	5	6	6	6
9	9	9	9	2	9	9	0	2	2	8

Table 5.10 shows the similarity measurements for all the tested patterns within the dataset and that the number of the first 10 closest digits to the tested patterns is similar to using sequential and non-sequential scanning in HCN-II. Both have 54 out of 100 patterns. Sequential scanning consumes more time than non-sequential scanning as shown in Table 5.11.

Table 5.11: Comparison of time consumption and number of correct placements between HCN-I and HCN-II using 'many to one' method

Feature Extraction	Time Consumption (s)	Number of Correct Placements
HCN-I:		
- No sequential scanning	7.06	37
- Sequential scanning	127.32	64
- No sequential scanning with middle area	17.03	37
- Sequential scanning with middle area	308.27	55
HCN-II:		
- No sequential scanning	35.47	70
- Sequential scanning	91.97	70
- No sequential scanning with middle area	6.84	54
- Sequential scanning with middle area	201.02	54

Upon evaluation of the performance of the 'many to one' method for feature extraction, it can be seen that the implementation of sequential scanning in HCN-I produces better results compared to non-sequential scanning. However, HCN-II provides more correct placements of the reference patterns when compared with the tested patterns. HCN-II gives the same number of placements (70 out of 100) when either including or excluding the middle area of the image. Opposite to HCN-I, HCN-II gives better results when sequential scanning is not implemented. This means that HCN-II is 3.6 times faster (see Table 5.11, 125.32s / 35.47s) than HCN-I in terms of time consumption.

5.3.2. Recognition using the 'One to Many' Method

Experiments similar to those reported in Section 5.2.1 are now conducted to analyse the results produced by the 'one to many' method. Table 5.7 presents the performance of this method without sequential scanning.

Table 5.12: ‘One to many’ method without sequential scanning in HCN-I

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	6	0	0	0	6	9	5	9
1	1	1	1	1	1	1	4	1	7	7
2	2	2	2	2	2	9	2	9	2	0
3	3	8	3	8	3	3	0	6	3	3
4	4	4	4	1	4	4	6	8	6	7
5	5	6	8	3	8	9	5	8	5	0
6	6	6	6	6	3	5	5	5	6	0
7	7	7	7	7	7	7	7	7	7	7
8	8	3	3	8	8	3	8	8	8	3
9	9	9	9	9	9	9	9	9	0	2

When extracting and recognizing the similarity of tested and reference patterns, performance improves slightly to 61% when using non-sequential scanning with the ‘one to many’ method; this is compared to when the ‘many to one’ method is used.

Table 5.8 shows the results when the middle area of the images during non-sequential scanning is included.

Table 5.13: ‘One to many’ method involving middle area of pattern without sequential scanning in HCN-I.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	6	9	0	0	9	0	5
1	1	7	1	7	1	4	1	7	1	7
2	2	2	2	9	2	2	0	2	0	2
3	3	3	3	3	5	8	3	0	8	6
4	4	1	4	8	4	6	7	3	2	6
5	5	5	0	5	3	8	3	6	5	0
6	6	6	6	5	5	6	5	6	5	5
7	7	7	7	7	7	7	7	2	7	2
8	8	8	8	3	3	3	8	3	8	0
9	9	9	9	9	9	9	9	2	2	2

Similar to the ‘many to one’ method, the ‘one to many’ method with no sequential scanning and including the middle area of image shows a lower performance than when the middle area is not involved. Its performance rate decreases by about 5% to 56%.

The following table shows the recognition results when using the ‘one to many’ method and scanning the whole image sequentially.

Table 5.14: ‘One to many’ method with sequential scanning in HCN-I

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	6	0	0	6	9	0
1	1	1	1	1	1	1	1	0	1	4
2	2	2	2	2	2	2	2	2	2	9
3	3	8	3	3	3	3	8	3	3	6
4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	6	6	5	3	3	5
6	6	6	5	6	6	5	6	6	3	6
7	7	7	7	7	7	7	7	7	2	7
8	8	8	8	8	8	3	3	3	8	6
9	9	9	9	9	9	9	9	0	2	9

When implementing sequential scanning, the performance rate goes up significantly to 75%. Table 5.10 shows the recognition results when involving the middle area of image.

Table 5.15: ‘One to many’ method involving middle area of pattern with sequential scanning in HCN-I

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	6	0	9	6
1	1	1	1	1	1	1	0	1	7	1
2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	8	5	3	9
4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	6	5	6	5	5
6	6	6	6	5	5	6	5	6	6	6
7	7	7	7	7	1	7	7	2	1	7
8	8	8	8	8	8	3	8	3	3	3
9	9	9	9	9	9	9	9	2	9	9

When including the middle area of the image, this method improves the performance rate slightly by 3% to 78%. The results of the investigation into similarity measures in HCN-II are shown in the following tables. Table 5.16 shows the results of implementing HCN-II without sequential scanning and without inclusion of the middle area of the image.

Table 5.16: ‘One to many’ method without sequential scanning in HCN-II

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	9	9	6	6	$\boxed{0}$	$\boxed{0}$	0
1	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	7	7	7	7	2	7
2	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	3	1	7	$\boxed{2}$	7	9
3	$\boxed{3}$	$\boxed{3}$	$\boxed{3}$	$\boxed{3}$	3	5	6	5	8	5
4	$\boxed{4}$	$\boxed{4}$	1	7	$\boxed{4}$	7	6	6	1	$\boxed{4}$
5	$\boxed{5}$	$\boxed{5}$	3	$\boxed{5}$	$\boxed{5}$	$\boxed{5}$	$\boxed{5}$	3	3	3
6	$\boxed{6}$	5	$\boxed{6}$	5	$\boxed{6}$	5	3	1	3	0
7	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	1	$\boxed{7}$	3	2
8	$\boxed{8}$	3	$\boxed{8}$	3	5	$\boxed{8}$	$\boxed{8}$	6	5	0
9	$\boxed{9}$	$\boxed{9}$	$\boxed{9}$	7	2	0	8	7	1	2

When feature activation is implemented in HCN-II, only 47 digits are placed close to their tested patterns. In this case, the scanning process excludes the middle area of the image. Table 5.17, on the other hand, shows the results when the middle area of the image is involved using non-sequential scanning.

Table 5.17: ‘One to many’ method involving middle area of pattern without sequential scanning in HCN-II.

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	6	9	0	9	6
1	1	1	1	1	7	7	1	7	7	7
2	2	2	2	2	2	9	7	9	9	0
3	3	3	3	3	3	5	5	1	5	6
4	4	4	4	7	7	7	4	1	4	1
5	5	5	5	5	3	6	6	5	3	3
6	6	5	5	6	5	6	6	6	0	3
7	7	7	7	2	2	7	7	7	7	7
8	8	8	8	8	8	3	5	6	3	6
9	9	9	9	9	2	9	0	2	2	7

The number of correct digits placed close to the tested patterns increases by 8 digits to 55 out of 100. The results in Table 5.18 are when feature extraction is implemented in HCN-II without involving the middle area of the image and when sequential scanning is used.

Table 5.18: ‘One to many’ method with sequential scanning in HCN-II

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	6	7	7	8	2	0	0	0
1	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	7	7	7	7	2	7
2	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	$\boxed{2}$	3	1	7	$\boxed{2}$	7	9
3	$\boxed{3}$	$\boxed{3}$	$\boxed{3}$	$\boxed{3}$	$\boxed{3}$	5	6	5	8	5
4	$\boxed{4}$	$\boxed{4}$	1	7	$\boxed{4}$	7	6	6	1	$\boxed{4}$
5	$\boxed{5}$	$\boxed{5}$	3	$\boxed{5}$	$\boxed{5}$	$\boxed{5}$	$\boxed{5}$	3	3	3
6	$\boxed{6}$	5	$\boxed{6}$	5	$\boxed{6}$	5	3	1	3	0
7	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	$\boxed{7}$	1	$\boxed{7}$	3	2	$\boxed{7}$
8	$\boxed{8}$	3	$\boxed{8}$	3	5	$\boxed{8}$	$\boxed{8}$	6	5	0
9	$\boxed{9}$	$\boxed{9}$	$\boxed{9}$	7	2	0	8	7	1	2

Here, the number of digits correctly placed within the same row decreases by 10 numbers to 45%. The results show that implementing sequential scanning does not improve the number of correct placements.

Table 5.19. shows the results of the investigation that implements sequential scanning when the middle area of the image is involved.

Table 5.19: ‘One to many’ method involving middle area of pattern without sequential scanning in HCN-II

Test Pats	Reference Patterns									
	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	6	9	0	9	6
1	1	1	1	1	7	7	1	7	7	7
2	2	2	2	2	2	9	7	9	9	0
3	3	3	3	3	3	5	5	1	5	6
4	4	4	4	7	7	7	4	1	4	1
5	5	5	5	5	3	6	6	5	3	3
6	6	5	5	6	5	6	6	6	0	3
7	7	7	7	2	2	7	7	7	7	7
8	8	8	8	8	8	3	5	6	3	6
9	9	9	9	9	2	9	0	2	2	7

Within Table 5.19, the number of correct placements is 55 out of 100. This indicates that there is no improvement when implementing sequential scanning in the HCN-II feature extraction. Further exploration of HCN-II is summarised in Table 5.20.

Table 5.20: Comparison of time consumption and number of correct placements between HCN-I and HCN-II using 'one to many' method

Feature Extraction	Time Consumption (s)	Number of Correct Placement
HCN-I:		
- No sequential scanning	3.58	61
- Sequential scanning	22.82	75
- No sequential scanning with middle area	8.69	56
- Sequential scanning with middle area	57.17	78
HCN-II:		
- No sequential scanning	1.68	47
- Sequential scanning	49.67	45
- No sequential scanning with middle area	3.9	55
- Sequential scanning with middle area	111.94	55

Table 5.20 indicates that HCN-I is better than HCN-II in terms of the correct placement of reference patterns related to the tested patterns. HCN-I produces 78 correct placement out of 100 patterns when involving the middle area of the image with sequential scanning.

5.4. Discussion and Summary

Feature extraction using hierarchical concatenation seeks to implement a mechanism similar to the one humans use for feature extraction and object recognition. HCN-I using the 'one to many' method produces the best results, because features are extracted from the tested patterns' point of view; however, it consumes 1.6 times more time than HCN-II, which has the best result with the 'many to one' method. In both methods ('many to one' and 'one to many'), the tested patterns find their match in a similar way to how a person selects a partner based on the similarity of their clothes (see Figure 5.3). Feature extraction from a pattern uses another pattern's features.

By selecting one number ('0_5', '1_6', '2_0', '3_7', '4_6', '5_5', '6_0', '7_0', '8_6', and '9_0') from each group, this thesis compares feature extraction in a hierarchical concatenation network between the 'one to many' method and the 'many to one' method. Both methods will be implemented for the purpose of classification in Chapter 6.

This chapter presents the results of pattern recognition within a hierarchical concatenation network. A tested pattern extracts a reference pattern using its own features. Its compares each reference pattern to itself and sorts the comparison results. For all experiments, the 10 highest similarity rates are presented.

Even though both methods show the concept of feature extraction based on a group of reference patterns, not all of the 10 highest similarities are correct matches.

For example, in Table 5.14, only tested pattern ‘4_6’ was 100% successful in extracting all elements from Group 4, while other groups also produced several misrecognitions. For example, when ‘0_5’ is used as the tested pattern, the algorithm allocates number ‘6_5’ in seventh position and ‘0_4’ in Position 8. From a human perspective, these results are not correct

Figure 5.7 shows the tested and reference patterns with coincidence arrays overlaid. The red square in these figures represents the coincidence array in Layer 1, while the blue, green and yellow squares represent the coincidence arrays in Layers 2, 3, and 4, respectively.

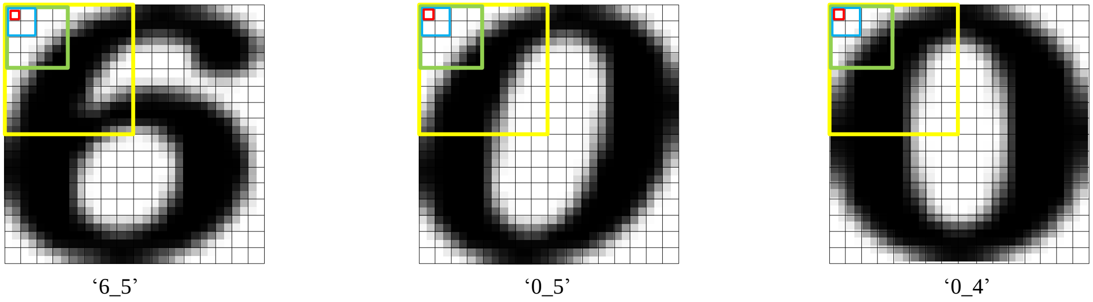

‘6_5’ ‘0_5’ ‘0_4’

Figure 5.7: Numbers within coincidence arrays. Image '0_5' is the tested pattern, and '6_5' and '0_4' are reference patterns

When the features are extracted hierarchically from layer to layer within the coincidence arrays, the feature similarity rates between the pair ‘0_5’ and ‘0_4’ and the pair ‘0_5’ and ‘6_5’ are shown in Table 5.12.

Table 5.21: Similarity rates within layers for pairs '0_5 - 0_4' and '0_5 - 6_5'

Tested Pattern		Similarity Rate (%)	
		Reference Patterns	
Layer	‘0_5’	‘0_4’	‘6_5’
	1	71.55	61.99
	2	54.49	66.13
	3	16.44	17.50
	4	0.13	0.66
	Average Rate	35.65	36.57

The similarity rates for ‘0_5’ and ‘0_4’ are 10% higher than for ‘0_5’ and ‘6_5’ in Layer 1. In Layers 2, 3, and 4, on the other hand, the similarity rates for pair ‘0_5’ and ‘6_5’ are higher than the pair ‘0_5’ and ‘0_4’. As the similarity calculation is based on an average of all layers, the second pair (‘0_5’ and ‘6_5’) shows a better similarity rate percentage than the first pair (‘0_5’ and ‘0_4’).

In summary, this chapter analyses the ability of a hierarchical concatenation network to perform feature extraction and measure the similarity rate between a pair of patterns based on their features. Both the ‘one to many’ method and the ‘many to one’ method have their own benefits in terms of recognition performance and the time it takes to extract the patterns. Both methods are discussed further in terms of pattern classification in the next chapter.

Chapter 6
Pattern Classification using a Hierarchical Concatenation Network (HCN)

6.1. Introduction

The recognition process using HCN compares the features of the tested image against the features of the reference image. The different ways to extract features using a HCN are presented in Chapter 5. However, in brief, the HCN extracts the features of an image in a hierarchical process from layer to layer. Features are also extracted using a coincidence array in each layer.

This chapter presents a classification method, which is inspired by the way humans classify objects and which will improve the performance of HCN classification. Classification cannot be separated from recognition. To classify an object into a specific class, out of several available classes, a classification method should be able to recognise the given object first. Classification is then based on the highest recognition rate between the given object and a class.

Chapter 5 presents two feature activation frameworks (HCN-I and HCN-II), each implementing two different ways to extract the features (using the ‘many to one’ and ‘one to many’ methods). This chapter further explores these methods and investigates their results. The results of the classification rate will enable a decision as to which method bests represent the HCN.

The way classification is explored in this research, using HCN feature extraction, is shown in Figure 6.1.

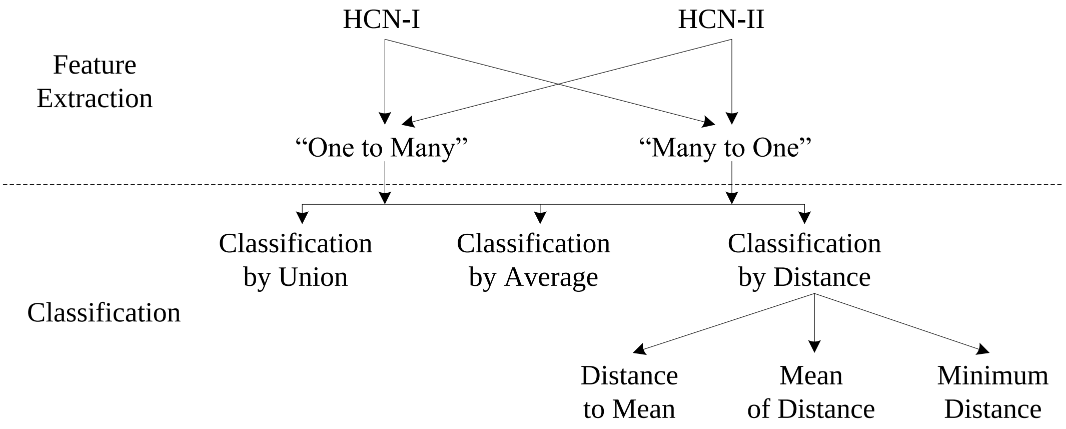


Figure 6.1: Classification exploration using HCN feature extraction

The difference between HCN-I and HCN-II is the way in which the upper layers' features are activated. HCN-I activates the upper layers' features based on the similarity of the lower layers' concatenated features with the activation group represented by bits of vector. HCN-II multiplies the features within the concatenated features with their frequencies, and also activates the upper layers' features based on the threshold value.

HCN is a feature extraction method, which extracts the features of a pattern using other patterns' features. In Chapter 5, this research finds that the the way in which features are extracted (i.e. by using the 'one to many' method or the 'many to one' method) plays an important role; this is because the method implies the position of the tested and reference patterns. The 'one to many' method is when the tested pattern extracts its own features before using its own features to extract the features from the reference pattern, while the 'many to one' approach is when the reference pattern extracts the features of the tested pattern.

Once both patterns (tested and reference) are extracted, the process continues to the classification step. Figure 6.2 shows that there are three different methods of classification. The first method is called ‘classification by union’. This method unifies the features of all reference patterns from the same class, then measures the similarity between the unified features and the features of the tested pattern. The second method is ‘classification by average’. This method measures the similarity rate between all tested patterns and reference patterns in the same class, and then calculates the average of those similarities. The last method is ‘classification by distance’, which is divided into three categories: ‘distance to mean’, ‘mean of distance’, and ‘minimum distance’. ‘Distance to mean’ measures the distance between the features of the tested patterns and the average of the reference patterns’ features within the same class. ‘Mean of distance’ measures each single distance between the tested and reference patterns within the same class, and then calculates the average of those distances. The ‘minimum distance’ is similar to the ‘mean of distance’, as it measures the distance between every pair of tested and reference patterns, but at the end, it chooses the minimum distance.

Before presenting the classification method using HCN, this thesis explores the pattern recognition process from a human perspective by illustrating the features that form the patterns in the following example. Let’s assume that the pattern in Figure 6.2 is an unknown pattern. In other words, it does not have a specific meaning. It can be recognised by examining its features.

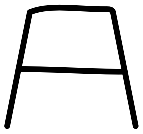


Figure 6.2: An example of a pattern

This pattern is formed by some features. It has two lines that stand almost vertical. The edges of the lines are connected at the top by a horizontal line, while the other ends at the bottom are not. There is another horizontal line that connects the two vertical lines at around their middle point.

If the patterns in Figure 6.3 are to be recognised, the patterns in Figure 6.2 could be used as reference. The tested patterns are examined in terms of their feature similarity to the reference pattern in Figure 6.2.

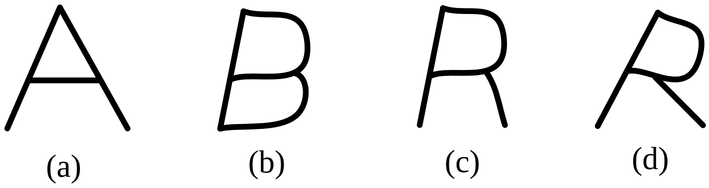


Figure 6.3: Examples of some other patterns

Looking at the features of Figure 6.3.(a), it has two standing lines, which are joined together at their top-end. It has one horizontal line, which connects the two standing lines. The other ends of the standing lines are open. It can be concluded that Figure 6.3 (a) has similar features to the pattern in Figure 6.2. Their similarity rate, however, would be less than 100%, because they are not exactly the same.

The next tested pattern is shown in Figure 6.3 (b). This pattern has one vertical line on its left side and two curved lines on its right side. The top-end of the vertical line connects with the top curve, and the bottom-end of the vertical line joins with the bottom curve. Based on these features, the similarity rate between the pattern in Figure 6.3 (b) and the pattern in Figure 6.2 is smaller than the similarity rate between Figure 6.3 (a) and Figure 6.2. The same feature similarity comparison can be applied to the remaining two patterns in Figure 6.3 ((c) and (d)).

The classification process groups a recognised object with other objects showing similar features. As the human brain is very powerful at recognition and classification, this chapter presents an idea that sprung from the three hypothetical questions below:

1. Can the classification process be conducted by comparing the features of a tested pattern against the group of features that belong to all patterns within a class (i.e. union features)?
2. Does it make a difference to performance if the classification process is conducted using the average similarity rate for each pair of tested patterns and all patterns within a class?
3. How is the performance rate affected when a distance measure (standard Euclidean distance) is used for classification?

To answer these hypothetical questions, classification in this thesis is divided into the three different methods discussed previously. The ‘classification by union’ method is used to answer the first question; the ‘classification by average’ method is used for the second question; and the ‘classification by distance’ method is used for the last question. These methods are used to group similar images into the same class.

With these hypothetical questions in mind, experiments are conducted using datasets to explore the ability of the HCN to classify tested patterns into a class of reference data. The experimental datasets are the datasets used in Chapter 5, being USPS (United State Postal Service) data, which is accessed from (http://statweb.stanford.edu/~tibs/ElemStatLearn/data.html), and the MNIST datasets, which are accessed from (https://github.com/myleott/mnist_png). The datasets used in Chapter 5 are produced using the imagemagick software (www.imagemagick.com).

The remainder of this chapter presents the concept, the algorithm, and the experimental results for proposed methods HCN-1 and HCN-2, as well as validation and discussion of the HCN network’s performance by exploring several similarity measures. Both HCN-1 and HCN-2 in this chapter implement the ‘one to many’ and ‘many to one’ method for feature extraction.

Feature extraction using HCN is presented in Chapter 5. In this section, the use of this feature extraction for classification is presented. It involves classification by union, average, and distance measure.

6.2. HCN Classification by Union Operation

As the name implies, in the ‘classification by union’ method, the features of a class are defined using the union operation. Figure 6.4 illustrates some patterns that have been classified according to the similarity of their features.

In a HCN network, each pattern is extracted using feature extraction, and this is presented in Chapter 5. In Figure 6.4., after feature extraction takes place, the features of the patterns in Class 1 are labelled from ‘1’ to ‘9’, while the features of the patterns in Class 2 are labelled from ‘a’ to ‘g’.

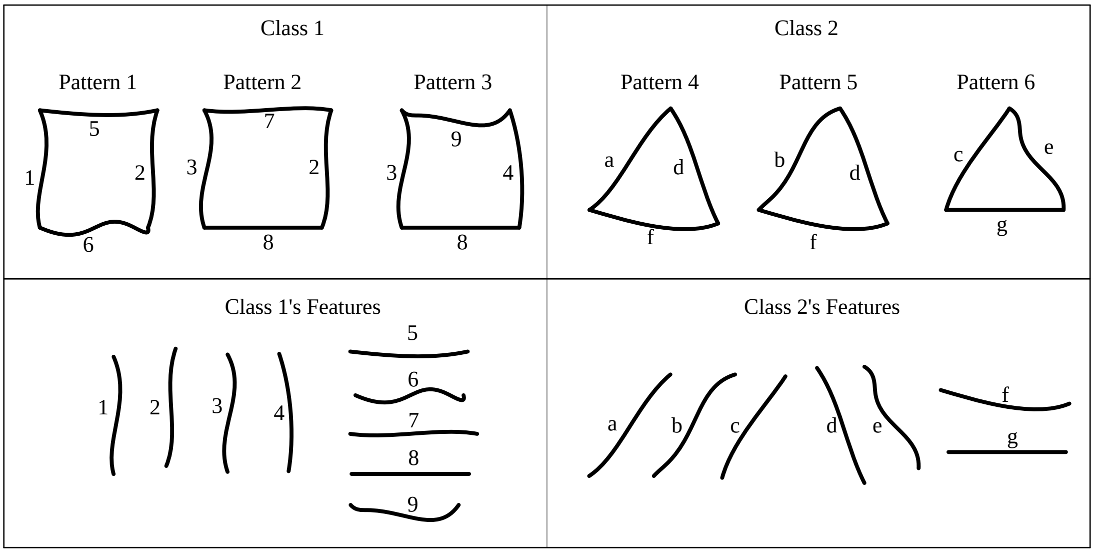


Figure 6.4: Classes of patterns with their class features

The features of a class are the product of a union process that results in classes with unique features. Different classes could show similar features, as shown in the above example, where Feature ‘g’ in Class 2 is similar to Feature ‘8’ in Class 1. With this in mind, Pattern 6 in Class 2 might show a certain similarity rate to Pattern 3 in Class 1.

Figure 6.5 shows an example image, and its features when classified.

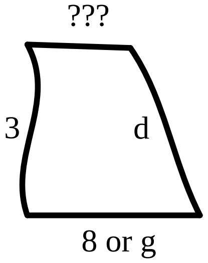


Figure 6.5: An example of a pattern to be classified

Figure 6.5 shows a pattern with some features that define it. Feature ‘3’ is a part of Class 1, and Feature ‘d’ is a part of Class 2. Another of its features could be part of either class. and the feature labelled with the question marks does not belong to any of the known classes.

The classification process by union compares the features of a given pattern to the unique features in each available class. The tested pattern in Figure 6.5 has a 50% similarity rate with Class 1, when looking at Features ‘3’ and ‘8’, and has also a 50%

similarity rate to Class 2, when looking at Features ‘d’ and ‘g’. The feature labelled with the question marks could have a certain similarity rate with Features ‘5’, ‘6’, ‘7’, ‘8’, and ‘9’ from Class 1, and also with Features ‘f’ and ‘g’ from Class 2. Based on these possibilities, the given tested pattern in Figure 6.5 shows a certain similarity rate with both classes. Also based on the highest similarity rate to the classes, it can be classified into Class 1 or Class 2.

In general, the HCN union classification process follows the steps below:

1. Extraction of features as discussed in Chapter 5.
2. Union operation to eliminate similar features of the reference images within the same class.
3. Calculation of the similarity rate between the tested images and all available classes of the reference patterns.
4. Classification of the tested image into a class of images based on the highest similarity rate between the tested image and the class of reference images.

The method to extract the features of the reference and tested image is presented in Chapter 5. Each tested or reference pattern extracts all their pairs; hence the features of each extracted tested or reference pattern are different.

6.2.1. Union Operation

The main objective of the ‘classification by union’ approach is the grouping together (union) of the same features into a node. This process is described in Algorithm 6.1.

Algorithm 6.1: Union operation for the reference pattern’s features in the same class

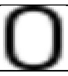
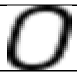
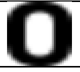
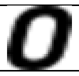
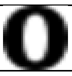
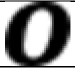
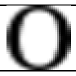
```
1:  For each tested pattern (n_TPat), set the class number (class) to be 1
2:  For all reference patterns (n_RPat), all members in each class (nPatClass)
3:  For all layers (Ln, 1 to 4)
4:  For each node in the current layer (nNodesL_n), perform union operation of
    the same node for all class members:

        $nodeLnC_R = \bigcup_{i=1}^{nPatClass} nodeLn_i$        (6.1)

3:  Repeat Line 4 until the last node in the current layer, Line 3 until the last
    layer, and Line 2 until the last reference pattern.
4:  Increase the class number (class = class +1), and repeat Line 2
```

Using Algorithm 6.1, the union operation is conducted as many times as the number of reference patterns *(n_RPat)*. For the ‘one to many’ method, each reference pattern is extracted by all the given tested patterns *(n_TPat)*, while for the ‘many to one’ method, each tested pattern is extracted by all the available reference patterns. The number of reference patterns in each class *(n_PatClass)* is defined in advance. For the explanation of the algorithm in this chapter, 100 digit datasets are used. Each class has 10 different shapes of digit; the first seven of them are used as reference images and the remaining three as tested images. This means *n_PatClass* is defined as 7.

For all members in the same class *(class)*, the features in the equivalent node’s number *(nNodesLn)* are assigned to the union operation. Therefore, a class of reference patterns that are extracted by a tested pattern *(tp)* have unique features in each node *($nodeLnC_R$)* for every layer *(Ln)*.

The sequential implementation of feature extraction in HCN-I (using ‘one to many’) gives a better result than a non-sequential implementation in HCN-I (using ‘one to many’). Table 6.1 shows an example of using union operation to determine the features of Class ‘0’. In this example, Class ‘0’, which is extracted by pattern ‘0_7.png’, contains seven patterns. Table 6.1 shows the extracted features of the reference patterns in each node of Layer 4, and after union operation is performed, it shows the features of every node in Class ‘0’. Using the ‘one to many’ method, all reference patterns are extracted by all given tested patterns, while using the ‘many to one’ method, the process is vice versa. As features are represented numerically (see Chapter 4), the features in Table 6.1 are an example of seven reference patterns extracted by only one tested pattern (using the ‘one to many’ method). If there are three given tested patterns, there are 21 possible extracted patterns in each class of reference patterns.

Table 6.1: Pattern ‘0’ and Class ‘0’ features in Layer 4

Node	Feature of each image ‘0’ and Class ‘0’ by union in Layer 4							
								Class ‘0’
1	0 2 3 4 6 8	0 2 3 4 5 6 7 8 12	0 2 3 4 8	0 2 3 4 7 8	0 2 3 4 6 8	0 2 3 4 6 8	0 2 4 6 8	0 2 3 4 5 6 7 8 12
2	0 2 3 4 6 8	0 2 3 4 5 6 7 8 12	0 2 3 4 8	0 2 3 4 7 8	0 2 3 4 6 8	0 2 3 4 6 8	0 2 4 6 8	0 2 3 4 5 6 7 8 12
3	0 2 3 4 6	0 1 2 3 4 6 7	0 2 4 6	0 1 2 3 4 6	0 2 4 6	0 1 2 3 4 5 6	0 2 3 4 6	0 1 2 3 4 5 6 7
4	0 2 3 4 6	0 1 2 3 4 6 7	0 2 4 6	0 1 2 3 4 6	0 2 4 6	0 1 2 3 4 5 6	0 2 3 4 6	0 1 2 3 4 5 6 7
5	0 2 3 4	0 2 3 4 6 7 14	0 2 4	0 2 3 4 6 7 8	0 2 4 6	0 2 4 6 8	0 2 3 4	0 2 3 4 6 7 8 14
6	0 2 3 4 6 8	0 2 3 4 5 6 7 8 2	0 2 3 4 8	0 2 3 4 7 8	0 2 3 4 6 8	0 2 3 4 6 8	0 2 4 6 8	0 2 3 4 5 6 7 8 12
7	0 2 3 4	0 2 3 4 6 7 14	0 2 4	0 2 3 4 6 7 8	0 2 4 6	0 2 4 6 8	0 2 3 4	0 2 3 4 6 7 8 14
8	0 2 3 4 6	0 1 2 3 4 6 7	0 2 4 6	0 1 2 3 4 6	0 2 4 6	0 1 2 3 4 5 6	0 2 3 4 6	0 1 2 3 4 5 6 7
9	0 2 3 4	0 2 3 4 6 7 14	0 2 4	0 2 3 4 6 7 8	0 2 4 6	0 2 4 6 8	0 2 3 4	0 2 3 4 6 7 8 14

6.2.2. Similarity Measure

After the union operation, each node of Class ‘0’ contains representative features from each of its members. The similarity between each tested pattern and the available classes is measured by calculating the ratio of similarity as described in Algorithm 6.2.

Algorithm 6.2: Similarity measure

```
For all given tested patterns (n_TPat)
  For all classes of reference pattern (nClass)
    For all layers (Ln 1 to 4)
      For all nodes in the current layer (nNodesLn)
        Check if the number of features in the tested pattern (n_Feat_T) is smaller
        than the number of features in the current class (n_Feat_C):
            $simNLn = \left(\frac{n\_Feat\_T}{n\_Feat\_C}\right) * 100$        (6.2)
        Otherwise:
```

Algorithm 6.2: Similarity measure

$$simNLn = \left(\frac{n\_Feat\_C}{n\_Feat\_T}\right) * 100 \qquad (6.3)$$

Repeat Line 4 for all nodes in Layer n, Line 3 until the last layer, and Line 2 until the last class

6: Calculate the similarity features in the current layer Ln:

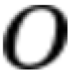
$$simLn = \frac{\sum_{n=1}^{nNodesLn} simNLn}{nNodesLn} \qquad (6.4)$$

The feature similarity rate between each node in all layers of the network *(simNLn)* is calculated by working out the ratio between the features of the given tested pattern *(n_Feat_T)* and the features of the class *(n_Feat_C)*.

The number of similar features between a tested pattern and a class of reference patterns is defined by an intersection operation. For example, if A is a vector [1 2 3 4] and B is a vector [3 4 5 6 7], the number of elements in A is 4 (denoted as *x*) and the number of elements in B is 5 (denoted as *y*). The similar elements (intersection) between A and B *(*denoted as *z)* are 3 and 4, or 2 elements. As B has more elements than A, then the similarity between A and B is calculated by finding their ratio *(z/y)* = 0.4 or 40%.

The overall similarity of features between a tested pattern and a class in layer *Ln (simLn)* is the average of the nodes' similarities *(simNLn)*. Table 6.2 shows a similarity comparison between the features of a tested pattern ('0_7.png') and the features of its class in Layer 4.

Table 6.2: Similarity rate between features of Pattern '0' and the features of its class

Nodes	Features of		Similarity Rate (%)
		Class ‘0’	
1	3 7 15	0 2 3 4 5 6 7 8 12	22.22
2	3 7 15	0 2 3 4 5 6 7 8 12	22.22
3	1 3 7	0 1 2 3 4 5 6 7	37.5
4	1 3 7	0 1 2 3 4 5 6 7	37.5
5	1 3 7 15	0 2 3 4 6 7 8 14	25
6	3 7 15	0 2 3 4 5 6 7 8 12	22.22
7	1 3 7 15	0 2 3 4 6 7 8 14	25
8	1 3 7	0 1 2 3 4 5 6 7	37.5
9	1 3 7 15	0 2 3 4 6 7 8 14	25
Average similarity rate in Layer 4 *(simNL4)*			28.24

In Node 9, Image ‘0_7.png’ has four features [1 3 7 15], while Node 9 of Class ‘0’ has eight features [0 2 3 4 6 7 8 14]. The similar features between Node 9 of Image ‘0_7.png’ and Node 9 of Class ‘0’ are ‘3’ and ‘7’; hence, the similarity rate for Node 9 between the tested image and Class ‘0’ is (2/8) x 100% = 25%. The average similarity of all nodes in Layer 4 is 28.24%.

The inclusion of the middle area of an image gives better results in terms of similarity between tested and reference patterns using HCN-I (as discussed in Chapter 5). Therefore, Layers 1, 2, 3, and 4 have an average node similarity rate of 576, 144, 36, and 9, respectively. This is compared to an average rate of 256, 64, 16, and 4 nodes, respectively, when the middle area of the image is not included. HCN-II, however, produces better results without the middle area of image. A specific class is chosen for the tested pattern by calculating the highest similarity rate between it and the available classes. Similarity rates are defined by selecting the layer (or the average of all layers) that produces the highest classification rate.

6.2.3. Classification Results with Union Operation

A classified pattern is defined by choosing the highest similarity rate between a tested pattern and the available classes. Table 6.3 shows example similarity rates between the tested pattern (‘0_7.png’) and the Class ‘0’ pattern for each layer in HCN-I using the ‘one to many’ method.

Table 6.3. Similarity rate for each layer

Layer	Similarity Rate (%)
1	74.53
2	30.79
3	29.09
4	28.24
Average Similarity	40.67

Classification in Layer 1, which has the highest similarity rate, can be seen in Table 6.4. The bold values indicate the highest similarity rate between the tested images and the predicted class.

Table 6.4 shows the classification results of 30 tested patterns against 10 reference classes. The 30 tested patterns are the last three members of each class in the digit datasets used in Chapter 5. The class features are obtained using the first seven patterns of each class in the dataset.

The values coloured red are misclassified images. In Table 6.4, 10 out of 30 tested images are correctly allocated to their own class. This means that the performance rate *(perf_rate)* for the digit datasets in Layer 1 of HCN-I using the union method is 67%. The performance rate can be calculated by dividing the number of correct classifications *($corr_{class}$)* by the total number of tested patterns *($tot_{testedPattern}$)*.

Table 6.4: ‘Classification by union’ similarity rates in Layer 1 of HCN-I using ‘one to many’ method

Tested Images	Similarity Rate to the Predicted Class (%)									
	0	1	2	3	4	5	6	7	8	9
0	74.53	72.25	59.12	61.40	61.01	62.29	63.97	69.49	61.90	63.78
0	61.72	58.45	46.40	51.60	48.79	51.15	50.85	54.23	45.47	52.95
0	61.71	52.73	44.69	50.22	47.83	48.35	48.21	48.43	43.22	48.04
1	45.33	61.11	43.34	38.62	46.02	39.28	40.44	55.36	40.41	42.35
1	46.26	52.73	34.33	35.48	41.70	39.81	36.76	45.52	34.93	38.51
1	50.50	54.58	38.94	40.00	45.20	43.39	39.88	46.88	39.37	42.58
2	60.54	69.12	71.22	52.92	61.01	50.13	53.82	71.46	58.46	57.42
2	58.05	65.82	71.35	51.10	61.76	48.15	52.32	67.45	57.03	55.74
2	56.29	61.22	69.00	49.91	57.93	47.12	50.80	62.95	52.80	55.92
3	60.42	60.86	53.68	69.10	55.47	63.86	62.34	58.09	60.14	64.19
3	50.42	53.58	51.64	67.30	49.31	56.53	58.17	50.73	61.75	57.68
3	47.23	47.83	45.91	59.74	44.69	51.40	52.33	45.60	55.98	52.55
4	64.09	75.76	59.20	50.29	68.86	59.42	54.05	73.95	52.67	54.99
4	65.38	69.72	58.95	51.67	68.07	55.72	52.68	69.01	53.43	52.24
4	63.36	61.96	59.19	53.63	64.49	53.38	51.04	62.61	54.03	50.85
5	62.17	61.79	49.04	57.53	55.36	63.05	56.73	60.79	54.74	59.09
5	49.66	52.48	38.61	50.44	44.11	55.58	49.92	47.84	47.34	49.39
5	48.81	51.12	38.07	48.31	39.89	57.46	49.63	47.59	48.14	48.59
6	69.13	66.02	53.96	64.30	56.80	68.79	63.87	61.95	59.33	64.67
6	65.71	71.03	59.21	65.92	64.39	69.80	70.74	67.76	64.84	68.80
6	53.95	59.32	46.50	58.94	50.48	63.89	62.01	54.83	56.08	58.82
7	51.00	64.23	53.77	41.05	56.07	47.10	45.46	65.70	45.38	46.09
7	51.08	57.95	46.06	37.14	50.27	39.34	41.39	58.48	41.70	41.55
7	44.40	55.10	38.33	34.25	43.74	35.64	37.77	51.24	34.91	38.94
8	64.31	69.85	62.68	70.38	64.91	67.81	71.37	64.44	74.60	71.23
8	64.38	65.22	65.81	65.28	62.71	65.60	71.15	64.63	72.29	72.39
8	51.13	46.38	53.53	50.65	46.98	47.24	52.92	45.97	56.95	53.68
9	72.11	74.05	64.89	65.25	59.40	68.11	66.87	67.45	66.33	71.61
9	67.45	72.97	64.04	63.85	61.45	68.25	65.59	70.01	64.73	70.27
9	52.90	55.66	51.35	54.23	49.79	57.86	57.70	53.78	56.81	60.46

By counting the number of correct classifications *(corr_class)*, the HCN recognition performance rate *(perf_rate)* for this digit dataset can be calculated as:

$$perf\_rate = \left(\frac{corr_{class}}{tot_{testedPatterns}}\right) * 100\% \qquad (6.5)$$

Table 6.5 shows the performance rate of classification in HCN-I using a union operation in all layers.

Table 6.5: Performance rate of ‘classification by union’ in all layers of HCN-I using ‘one to many’ and ‘many to one’ methods.

Layer	Performance Rate (%)	
	HCN-I (‘one to many’)	HCN-I (‘many to one’)
1	67	NA
2	17	0
3	47	3
4	63	NA
Average similarity all layers	83	13

In Table 6.5, the performance rate of HCN-I using the ‘one to many’ method in Layer 1 is higher (at 67%) than in the other layers. The rate decreases sharply to 17% in Layer 2, and increases slightly for each layer until the last layer, which is Layer 4 at 63%. The average of all layers shows the highest rate at 83%. This indicates that the overall performance rate of HCN-I using the ‘one to many’ method should not be based on the performance rate of the top layer. Instead, the rate should be established by basing it on the highest performance rate of all layers or the average performance rate of all layers. The reason for this is because the features might look similar in smaller pieces compared to bigger pieces, and vice versa.

Using HCN-I with the ‘many to one’ method, several performance rates are labelled with ‘NA’ meaning ‘not available’. This means that the tested pattern can be classified into more than one class. In other words, it shows a similarity to more than one class. This result indicates that HCN-I does not give a better result when using the ‘many to one’ method with ‘classification by union’.

The HCN-II feature extraction method, however, does not need to implement sequential shifting of the image to give better results in terms of similarity as discussed in Chapter 5; nor does it need to include the middle area of the image. With various threshold values for feature activation, Table 6.6 shows the union classification performance rates in HCN-II feature extraction.

Table 6.6: Performance rate of HCN-II using union classification

Activation Threshold (%)	Performance Rate (%)									
	HCN-II ('one to many')					HCN-II ('many to one')				
	L1	L2	L3	L4	Average	L1	L2	L3	L4	Average
0	73	60	13	17	27	80	60	30	NA	70
0.1	73	67	7	17	33	80	33	30	NA	67
0.2	73	67	7	23	37	80	23	20	NA	60
0.3	73	70	13	21	40	80	30	20	NA	60

A threshold value of up to 70% for feature activation in the upper layers of HCN-II (using the 'one to many' method) shows the highest performance rate in Layer 1 (at 73%). Using the 'many to one' method in the same layer, on the other hand, gives a performance rate of 80%. By averaging the similarity rate of all layers, the '0' threshold value gives the highest classification rate in HCN-II using 'many to one'.

6.2.4. Summary of Classification by Union

When performing union operation in the HCN-II feature extraction method, it does not need the threshold value for feature activation in the upper layers. In terms of the performance rate, HCN-I is better at 83% (using the 'one to many' method) compared to HCN-II at 80% (using the 'many to one' method); however, HCN-I consumes more time to extract the features (Chapter 5, Tables 5.11 and 5.20).

6.3. HCN Classification by Average

In this thesis, if 'classification by union' groups the features of patterns within the same class and then compares those features to the features of a tested pattern, 'classification by average' compares a tested pattern's features to all patterns in all classes and then measures the average similarity between a tested pattern and all classes. Classification is then based on the highest similarity rate between a tested pattern and one of the available classes.

The following example in Figure 6.6 illustrates the 'classification by average' process. Each feature of the tested pattern is compared to all features of all patterns in all classes. Using the 'one to many' method, the features of the reference patterns are extracted using each tested pattern's features. Therefore, the comparison between the tested and reference patterns is conducted only between the tested and reference patterns extracted by the tested patterns themselves. However, using the 'many to one' method, the comparison is measured between the tested patterns extracted by the reference patterns.

The comparison of each pair gives a certain similarity rate. The average similarity rate is calculated for all pairs in a class, and afterwards, each tested pattern and its class has a certain similarity rate.

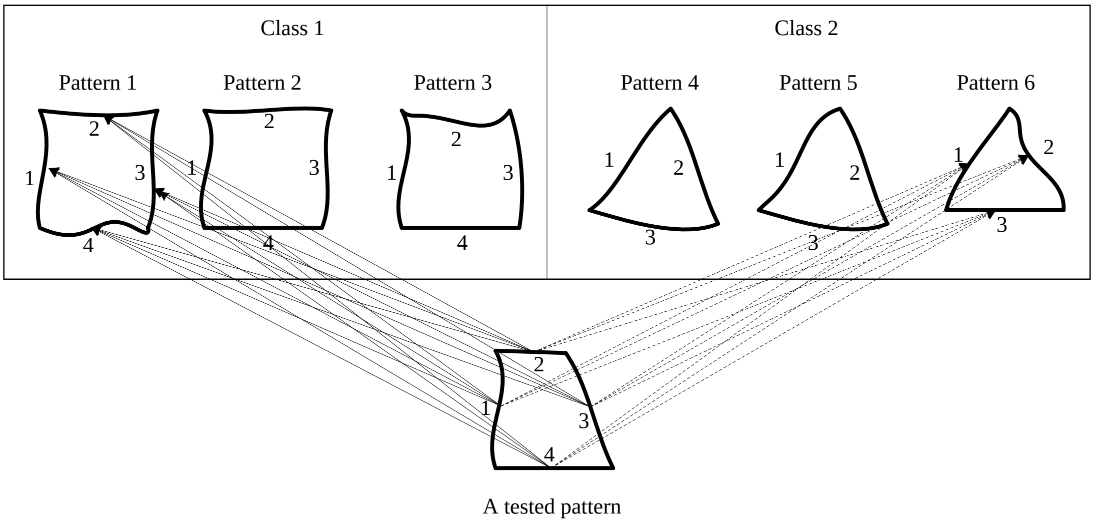


Figure 6.6: Feature similarity comparison between a tested pattern with all patterns in all classes

The tested pattern is classified into the class that shows the highest similarity rate with the tested pattern. All features of the tested pattern in Figure 6.6 are compared with all features of all patterns in each class. The comparison process, which produces the similarity rate, is similar to when the 'classification by union' method is used.

The 'classification by average' process in the HCN follows the steps below:

1. Extraction of the reference patterns' features in each class.
2. Feature similarity measure between each tested pattern and all patterns in each class.
3. Calculation of the similarity average between the tested pattern and all patterns in all available classes.

6.3.1. Similarity Measure

If the 'classification by union' process measures the similarity between the tested patterns and the group of patterns in the class, the 'classification by average' process measures the similarity between the tested patterns and all members within the class, and then calculates the average of similarities. The following algorithm (6.3) describes the process of a similarity measure between the features of the tested and reference patterns.

Algorithm 6.3: Features similarity measure

```
1:  For all given tested patterns (n_TPat), start at Class 1
2:  For all reference patterns (n_RPat) and for all nodes in layer n (nNodesLn),
    initial the temporary similarity rate of a node on layer n (tmpSimNLn) with
    '0'
3:  For all patterns in the current class
4:  Count the number of similar features between both patterns (n_simFeat)
5:  Check if the number of features in the current node's tested pattern
    (n_featTPat) is less than the number of features in the the current node's
    reference pattern (n_featRPat), and count the simNLn as:

        $simNLn = \frac{n\_simFeat}{n\_featRPat} * 100$        (6.6)

    If the n_featTPat is greater than the number of features in the reference
    pattern n_featRPat, count the simNLn as:

        $simNLn = \frac{n\_simFeat}{n\_featTPat} * 100$        (6.7)

    Add the similarity rate of the current node into the temporary similarity rate
    of layer n:

        $tmpSimNLn = tmpSimNLn + simNLn$        (6.8)

    Repeat Line 3 until the last pattern of the current class
4:  Calculate the similarity rate within the current layer (simLn) by
    accumulating the value in tmpSimNLn, dividing by the number of patterns in
    the class (n_PatClass), and then dividing by the number of nodes in the
    current layer (n_NLn):

        $simLn = \frac{\left(\sum_{the\ first\ pattern\ in\ class\ n}^{the\ last\ pattern\ in\ class\ n} tmpSimNLn\right)/n\_PatClass}{n\_NLn}$        (6.7)

5:  Increase the class number
6:  Repeat Line 2 until the last pattern in the last class
7:  Repeat Line 1 until the last tested pattern
```

Within Algorithm 6.3, the similarity of features in each node *(simNLn)* is the comparison between the total number of similarities *(n_SimFeat)* and the larger number of features in the tested or reference pattern *(n_FeatTPat)* or *(n_FeatRPat)*.

Table 6.7 shows the example similarity rates between the features of a tested pattern (image '0_7.png') and the features of Class '0' in Layer 4's nodes (using HCN-I with 'one to many' feature extraction).

Table 6.7: Similarity rates between a tested pattern '0_7.png' and all members of Class '0' in the nodes of HCN-I Layer 4 using 'one to many' method

Member of Class '0'	Similarity Rate of Features in Layer 4's Nodes								
	1	2	3	4	5	6	7	8	9
1	16.67	16.67	20	20	25	16.67	25	20	25
2	22.22	22.22	42.86	42.86	28.57	22.22	28.57	42.86	28.57
3	20	20	0	0	0	20	0	0	0
4	33.33	33.33	33.33	33.33	28.57	33.33	28.57	33.33	28.57
5	16.67	16.67	0	0	0	16.67	0	0	0
6	16.67	16.67	28.57	28.57	0	16.67	0	28.57	0
7	0	0	20	20	25	0	25	20	25
Average	17.97								

The similarity rate between image '0_7.png' and Class '0' in Layer 4 is 17.97%. This similarity rate is the average of the nodes' similarity in Layer 4 *(simL4)*. Table 6.8 shows the similarity rate between pattern '0_7.png' and Class '0' in all layers.

Table 6.8: Similarity rate in each layer

Layer	Similarity Rate (%)
1	87.01
2	32.15
3	26.91
4	17.97
Average Similarity *(simLn)*	41.01

6.3.2. Classification Results with Average Operation

The decision to classify a tested pattern into a certain class is based on the highest average similarity rate between the tested pattern and all available classes. Table 6.9 shows the percentage of similarity between the tested images and the available classes using HCN-I feature extraction with the ‘one to many’ method. Both the dataset and the proportion of the tested and reference patterns are similar to those used in the experiment in an HCN using the ‘classification with union’ method. The first seven members of each class are used as references, and the remaining three are used as tested data.

Table 6.9: Similarity rates between the tested images and all classes in all layers

Tested Images	Similarity Rate to the Actual Class (%)									
	0	1	2	3	4	5	6	7	8	9
0	41.01	35.14	32.31	36.38	31.53	35.06	35.89	34.63	33.72	32.22
0	36.33	36.19	29.79	31.19	30.61	32.33	32.27	30.91	30.19	33.06
0	47.48	32.35	35.49	41.65	31.80	38.15	38.32	31.11	37.39	38.74
1	33.45	36.65	34.25	32.51	35.77	34.03	32.28	36.28	33.10	30.80
1	45.52	46.64	39.81	40.88	43.41	41.44	41.92	43.66	38.38	41.38
1	41.92	48.79	38.39	36.29	37.94	36.35	37.58	43.99	37.24	37.53
2	29.67	26.92	40.96	29.18	31.91	26.35	27.12	32.61	30.01	33.43
2	29.59	30.71	40.31	29.76	31.67	29.13	30.54	32.82	31.89	32.81
2	30.38	27.95	43.60	30.09	29.87	26.09	27.17	31.73	30.87	34.72
3	31.76	29.00	28.53	39.21	25.00	33.98	30.97	28.00	32.70	31.07
3	32.91	31.34	33.56	40.94	26.90	34.51	33.71	30.14	38.15	35.58
3	35.50	29.69	32.28	43.76	28.19	37.33	34.78	31.79	40.64	34.98
4	31.92	35.15	32.16	27.03	39.64	31.45	29.24	35.96	28.03	29.13
4	32.02	33.27	33.17	30.15	36.76	30.34	29.94	35.54	29.29	31.52
4	29.35	30.70	32.63	28.51	33.92	26.62	29.44	30.84	27.37	28.62
5	33.77	33.22	26.57	31.15	29.65	38.89	32.76	30.53	29.46	29.45
5	30.11	35.27	28.44	33.56	29.43	39.08	32.86	31.42	30.46	29.64
5	37.21	36.83	32.85	41.04	33.45	48.33	38.55	32.96	40.58	34.44
6	34.94	30.50	26.93	34.54	27.73	34.50	37.92	29.35	31.23	28.96
6	33.66	32.41	30.94	35.42	31.44	34.94	35.75	32.28	33.69	31.81
6	30.09	27.41	24.72	36.06	26.74	32.86	37.46	26.99	31.40	27.17
7	29.50	33.25	33.06	30.48	32.36	30.89	31.30	38.95	27.31	29.65
7	35.52	38.95	35.08	31.92	35.08	32.11	34.11	35.86	33.29	33.04
7	34.19	37.08	33.61	32.49	35.03	32.06	33.40	35.86	32.17	32.40
8	32.41	26.26	29.97	35.91	27.02	31.55	33.96	25.31	39.61	30.31
8	30.64	25.64	29.85	31.12	27.19	29.47	30.86	28.11	38.78	30.52
8	28.20	21.83	29.55	34.55	27.91	30.48	33.98	25.81	41.83	32.64
9	34.32	30.87	32.14	31.05	26.00	30.12	30.41	31.07	31.24	35.88
9	33.17	31.96	31.72	32.82	29.96	36.01	33.66	33.48	34.12	36.70
9	34.31	27.17	30.94	32.85	25.61	32.29	29.55	26.41	32.38	38.18

In Table 6.9, the similarity rates in the bold font show the highest similarity rate between the tested pattern and a predicted class. The similarity rates of misclassified patterns are represented by the red-coloured font. In HCN-I, using the 'one to many' method, there are two misclassified patterns out of a total of 30. The second seven members have a similarity rate of 38.95% with Class 1 and only 35.86% with Class 7. The third seven members also have higher similarity rate to Class 1 (37.08%) than its own class (35.86%). In other words, this classification process has a performance rate of 93% (28 out of 30).

Table 6.10: Performance rate of 'classification by average' in HCN-I using 'one to many' method

Layer	Performance Rate (%) of HCN-I with 'one to many'	Performance Rate (%) of HCN-I with 'many to one'
1	70	30
2	37	80
3	67	20
4	53	13
Average similarity in all layers	93	33

In Table 6.10, the average performance rate of all layers is highest using the 'one to many' method (93%), although the 'many to one' method gives the highest classification rate in Layer 2 (80%). The 'many to one' method gives a 33% average for all layers.

Table 6.11 shows the performance rates of HCN-II using the 'classify by average' approach with different methods of feature activation.

Table 6.11: Several threshold tests on the performance rate using 'classification by average' in HCN-II with 'one to many'

Activation Threshold (%)	Performance Rate (%) of HCN-II using 'one to many'					Performance Rate (%) of HCN-II using 'many to one'				
	L1	L2	L3	L4	Average	L1	L2	L3	L4	Average
0	60	73	33	50	50	60	73	20	47	83
0.1	60	70	33	47	50	60	70	20	47	83
0.2	60	70	33	53	53	60	70	20	50	80

Activation Threshold (%)	Performance Rate (%) of HCN-II using ‘one to many’					Performance Rate (%) of HCN-II using ‘many to one’				
	L1	L2	L3	L4	Average	L1	L2	L3	L4	Average
0.3	60	70	33	53	53	60	70	27	50	80

Comparing the results between Tables 6.10 and 6.11, it can be seen, when the average similarity rate between a tested pattern and all patterns in a class is measured, performance rates are lower in HCN-II than in HCN-I. Table 6.11 shows that the majority of the highest performance rates in HCN-II are in Layer 1 (73%), while the average of all layers is 53% using the ‘one to many’ method and 83% using ‘many to one’.

6.3.3. Summary of Classification by Average

According to Tables 6.10 and 6.11, the ‘classification by average’ method gives the best performance rate in HCN-I using the ‘one to many’ method. Its best rate is the average similarity rate of all layers in the network. It is 10% higher than in HCN-II using ‘many to one’.

6.4. HCN Classification with a Distance Measure

A known similarity measure, using Euclidean distance, has also proved popular when comparing a pattern with other patterns. Using the same feature extraction method, this section investigates the use of Euclidean distance in three different measurement methods: ‘distance to mean’, ‘mean of distance’, and ‘minimum distance’.

6.4.1. The Concept of Distance Measure using HCN

As illustrated in Figure 6.7, ‘distance to mean’ compares the distance of a tested pattern with the average features value of all reference patterns within the same class

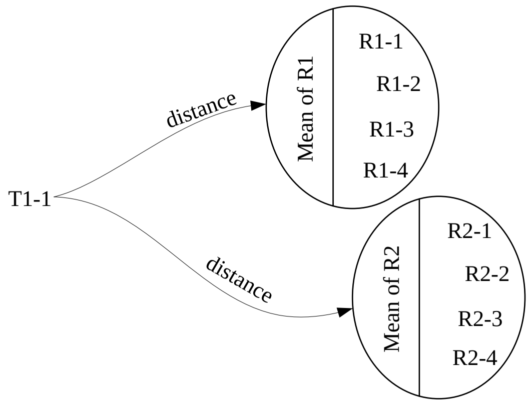


Figure 6.7: An illustration of ‘distance to mean’

Within the illustration in Figure 6.7, Class 1 of the reference patterns contains patterns R1-1 to R1-4 while Class 2 contains R2-1 to R2-4. The mean of all patterns in Class 1 is denoted as ‘mean of R1’, and the mean of all patterns in Class 2 is called the ‘mean of R2’. The distances between the tested pattern (T1-1) and the two classes are respectively:

- the distance between T1-1 and the ‘mean of R1’, and
- the distance between T1-1 and the ‘mean of R2’

Algorithm 6.4 shows the process of measuring the distance by ‘distance to mean’.

Algorithm 6.4: Distance to Mean

```
1: For all tested patterns (n_TPat), start a class from 1
2: For all reference patterns (n_RPat) within the current class, initial the mean
   of the pattern in each layer (meanPatLn) with the first pattern ($patLn_1$) in the
   class
3: From the second to the last pattern (m) in the current class:

       $meanPatLn = \sum_{2}^{m} meanPatLn + patLn_m$        (6.8)

4: Calculate the mean of the patterns within the current class:

       $meanPatLn = \frac{meanPatLn}{number\ Pattern\ in\ a\ class}$        (6.9)

5: Increase the number of classes, and repeat from Line 1
6: Measure the distance within each layer (Ln):

       $distLn = \sqrt{(patLn\_T - patLn)^2}$        (6.10)
```

Using the same dataset in HCN-I with the 'one to many' method gives the best performance rate. This approach classifies the tested patterns by calculating the average distance to the reference patterns from all layers (Table 6.13). The detailed similarity rates are shown in Table 6.12.

Table 6.12: Classification in HCN-I using the 'one to many' method and 'distance to mean'

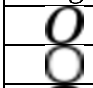
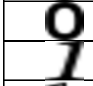
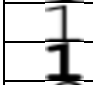
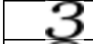
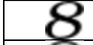
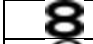
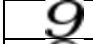
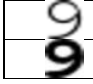
Tested Images	Similarity Rate to the Predicted Class (%)									
	0	1	2	3	4	5	6	7	8	9
	24.42	44.83	42.01	36.35	46.84	35.30	35.48	49.07	36.80	36.07
	37.60	49.07	45.58	42.33	53.64	45.53	44.07	51.23	44.57	43.81
	20.68	46.52	38.95	34.56	46.12	35.86	34.08	50.20	33.69	33.32
	44.66	31.30	34.74	36.62	34.43	41.23	43.76	34.58	41.57	38.34
	47.72	32.75	39.38	43.63	42.72	43.11	45.12	38.77	46.83	43.72
	49.86	32.63	38.92	44.46	45.02	43.51	46.14	40.41	46.75	44.67
	40.92	39.21	25.26	39.14	46.99	42.35	45.18	41.46	41.75	37.53
	46.14	38.49	30.86	42.26	45.83	46.08	48.51	38.07	46.39	39.93
	40.95	40.08	22.06	38.88	47.03	43.33	46.39	42.96	41.83	35.77
	39.71	42.87	41.66	29.65	46.46	34.30	38.27	48.34	35.81	44.38
	42.11	36.75	38.72	32.47	44.24	37.86	40.53	41.25	37.88	41.63
	37.11	37.31	35.12	23.26	43.32	33.13	36.71	42.17	30.34	38.35
	46.19	37.62	43.86	40.92	24.53	40.65	39.75	44.63	42.26	43.73
	44.20	41.58	46.25	45.36	33.80	42.89	40.84	48.79	44.22	45.03
	48.20	41.62	47.49	43.58	26.73	41.98	39.25	51.81	42.54	47.00
	37.08	41.25	42.52	35.43	44.40	30.41	35.69	45.55	38.09	41.56
	42.70	40.01	42.30	39.88	45.32	33.65	35.85	44.93	40.58	44.06
	37.47	39.17	39.53	30.64	42.85	23.38	29.10	45.59	30.88	41.56
	34.01	46.42	46.79	37.37	44.86	30.80	30.24	51.31	34.01	42.62
	39.18	43.07	45.99	38.67	42.72	35.03	34.55	48.68	37.06	42.66
	41.63	43.96	46.66	39.63	40.39	32.64	30.28	51.24	34.77	44.49
	49.34	34.10	40.50	44.01	45.12	45.51	47.65	27.37	47.88	44.21
	46.86	34.60	40.64	41.80	39.31	43.53	45.90	33.59	45.72	40.80
	43.89	37.09	37.22	39.14	41.11	44.13	48.20	35.58	43.51	36.35
	40.57	47.69	45.53	35.92	46.29	36.34	36.86	51.26	28.20	43.55
	39.15	44.54	43.58	37.01	43.48	38.22	37.74	48.02	32.13	38.45
	33.58	47.99	41.86	35.37	47.29	37.26	36.08	51.70	26.44	36.47
	35.41	46.01	41.12	41.30	49.15	40.15	43.74	50.26	40.95	32.62
	38.73	40.31	40.78	39.60	44.89	39.97	42.53	43.15	40.29	32.96
	37.93	45.93	40.56	40.22	49.31	40.60	43.83	47.33	38.00	28.82

Table 6.12 shows the average similarity rates for all layers, and it can be seen that there are no misclassified patterns. This means that the performance rate of HCN-I using 'one to many' is 100%.

The classification rates for other layers using both the 'one to many' and 'many to one' methods are shown in Table 6.13.

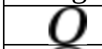
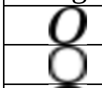
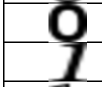
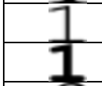
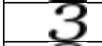
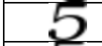
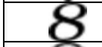
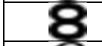
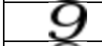
Table 6.13: Performance rate of HCN-I using a 'distance to mean' similarity measure

Layer	Performance Rate (%) of HCN-I using 'one to many'	Performance Rate (%) of HCN-I using 'many to one'
1	97	97
2	43	13
3	67	13
4	50	13
Average distance of all layers	100	90

Even when implementing the 'many to one' method, Layer 1 has the highest performance rate when compared to other layers. Its performance rate is 10% lower than when the 'one to many' method is used for the feature extraction in HCN-I.

In HCN-II, using both the 'one to many' method and the 'many to one' method, the best performance rate is on Layer 2. Table 6.14 shows the similarity rates between the tested patterns and all available classes. The similarity rates are based on Layer 2's results when implementing the 'many to one' method in HCN-II.

Table 6.14: Classification in HCN-II using a 'distance to mean' similarity measure

Tested Images	Similarity Rate to the Predicted Class (%)									
	0	1	2	3	4	5	6	7	8	9
	97.736	244.59	230.95	192.55	251.95	183.75	182.43	279.15	194.81	185.33
	173.07	234.21	229.83	211.79	273.15	228.34	220.22	260.58	237.62	217.35
	89.015	277.61	235.58	207.52	272.23	206.57	197.23	307.84	203.27	190.19
	263.7	166.95	193.57	200.71	181.82	236.33	256.83	195.75	247.48	219.03
	270.62	141.29	211.41	227.64	217.33	225.62	240.72	199.98	264.62	239.65
	315.16	176.42	233.66	265.72	265.74	262.47	278.87	238.76	290.75	277.69
	211.75	192.37	103.53	192.09	248.44	215.9	239.17	218.12	221.3	193.63
	234.02	170.07	137.6	199.71	228.95	226.62	243.81	177.62	241.79	200.6
	226.68	207.65	92.906	202.82	261.86	238.17	263.34	233.51	232.47	195.57
	190.26	197.56	193.7	121.2	229.33	156.48	183.89	235.73	171	216.12
	220.67	160.58	188.11	152.2	219.38	183.99	201.33	196.17	206.25	210.74
	202.12	176.88	169.67	87.093	219.8	154.15	189.78	218.38	162.4	196.65
	252.15	186.81	241.19	210.64	99.183	216.21	209.23	241.9	228.29	231.48
	232.57	207.86	242.86	226.01	155.22	221.82	210.64	258.68	233.3	235.68
	272.31	221.71	263.01	233.05	118.42	225.75	206.32	295.87	226.12	261.56
	187.7	213.57	228.68	181.64	241.14	143.6	182.11	243.91	201.5	217.9
	220.31	185.4	214.94	189.01	232.86	156.65	183.32	218.43	210.14	220.53
	206.92	206.12	217.62	153.98	230.84	100.59	150.18	253.63	163.56	224.57
	150.02	239.03	243.31	179.21	227.24	140.1	128.4	267.94	155.75	208.72
	182.72	196.53	224.99	177.55	201.56	150.31	146.38	237.13	178.9	201.36
	215.47	236.43	258.42	207.73	206.58	148.9	129.68	291.37	164.92	235.93
	284.61	181.36	230.99	241.42	259.21	256.95	274.04	127.83	271.85	250.12
	272.84	179.83	225.4	229.7	214.45	248.37	267.07	175.04	270.56	229.21
	267.27	205.51	216.3	226.68	239.62	257.12	289.06	189.52	266.12	213.68
	181.79	227.53	221.41	152.42	214.87	152.69	153.84	258.94	103.74	196.74
	173.89	187.96	191.75	150.7	190.78	153.79	155.78	216.7	139.83	153.76
	168.17	262.35	221.77	179.47	248.13	183.38	177.64	292.29	115.55	181.49
	153.61	226.31	203.88	194.97	242.26	189.4	208.73	259.97	194.23	129.45
	188.63	187.65	201.28	190.46	219.39	191.15	208.76	212.47	207.03	139.6
	195.04	246.15	210.1	217.28	267.67	215.43	236.58	265.82	202.93	120.19

To get the results in Table 6.14, the threshold value for feature activation in the upper layers is set at 0.3. This indicates that HCN-II can recognise the current digit dataset with a 100% performance rate using a 'distance to mean' distance measure in Layer 2.

Performance rates within all layers using different activation thresholds are shown in Table 6.15.

Table 6.15: Performance rate in all layers using a 'distance to mean' distance measure

Activation Threshold (%)	Performance Rate (%)									
	HCN-II using 'one to many'					HCN-II using 'many to one'				
	L1	L2	L3	L4	Average	L1	L2	L3	L4	Average
0	97	100	77	57	90	97	100	43	40	77
0.1	97	100	80	57	90	97	100	47	40	77
0.2	97	100	83	63	93	97	100	40	40	73
0.3	97	100	90	70	93	97	100	40	40	80

A threshold value of up to 70% (0.3) for feature activation, using both the 'many to one' and 'one to many' method, gives a 100% performance rate in Layer 2 when classifying the tested patterns into the correct class.

'Mean of distance' is the average measurement of distance between a tested pattern and all patterns in a class. It is illustrated in Figure 6.8.

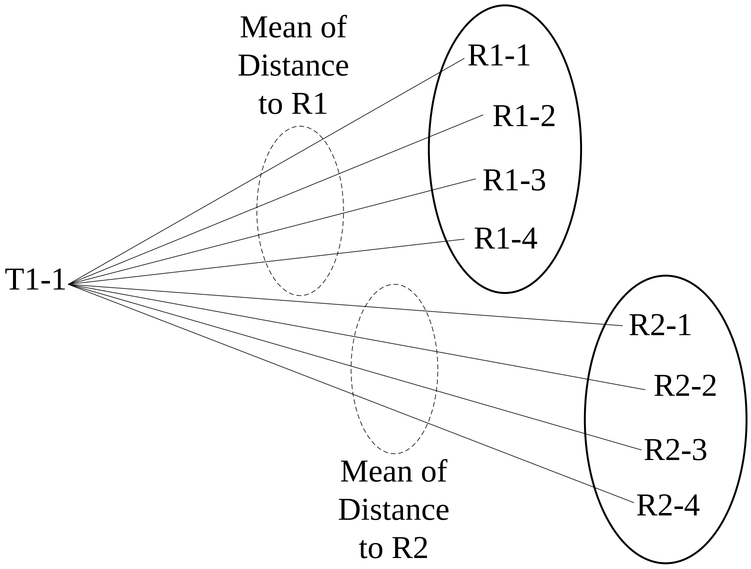


Figure 6.8: An illustration of 'mean of distance'

Based on the illustration in Figure 6.8, there are two steps to calculate the mean of distance. Firstly, the distance is measured between a tested pattern and all reference patterns in Classes 1 and 2. Secondly, the distance between a tested pattern and the class is calculated by averaging the distance between the tested and reference patterns within the same class.

The concept of a distance measure using ‘mean of distance’ is described in Algorithm 6.5.

Algorithm 6.5: ‘Mean of Distance’

```
For all tested patterns (n_TPat), start a class from 1
For all reference patterns (n_RPat) within the current class, calculate the distance in all layers (dLn) between the tested pattern (TPat) and each reference pattern (RPat) within the current class:
    $dLn = \sqrt{(TPat - RPat)^2}$    (6.11)
Calculate the average distance between the tested pattern and all patterns (from 1 to m) within the current class (distLn):
    $distLn = \frac{\sum_1^m dLn_m}{m}$    (6.12)
```

Table 6.16 shows the similarity rates between a tested pattern and all classes. This table refers to the classification rate in HCN-I using the ‘one to many’ method in Layer 1, because this layer shows a 97% performance rate. Within this table, there is only one image (‘1-png’) misclassified into Class ‘7’.

Table 6.16: Classification in HCN-I using ‘one to many’ method with a ‘mean of distance’ distance measure

Tested	Similarity Rate to the Predicted Class (%)									
Images	0	1	2	3	4	5	6	7	8	9
0	96.74	164.70	151.46	133.30	167.13	133.60	134.63	167.17	135.82	136.34
0	120.82	158.76	148.47	137.05	171.68	148.67	144.92	155.24	148.48	144.21
0	86.68	173.64	147.31	131.80	170.03	137.71	132.72	173.85	131.36	130.01
1	161.92	129.94	132.09	137.63	131.13	155.11	164.30	124.73	157.10	145.68
1	174.66	131.43	149.90	160.03	159.43	162.22	167.16	137.76	174.62	165.32
1	178.07	123.96	144.23	158.48	162.59	160.10	166.90	140.47	169.69	165.83
2	139.17	143.04	98.65	133.70	161.34	146.83	155.88	138.13	145.65	137.30
2	153.32	140.66	114.30	141.60	158.18	157.11	164.65	127.27	158.74	143.03
2	143.60	149.34	90.54	136.23	168.73	155.06	165.55	141.95	149.74	134.72
3	132.94	149.42	139.42	105.38	159.18	126.05	137.53	150.73	126.24	151.09
3	142.80	136.36	133.81	114.62	153.76	136.03	143.23	131.98	136.06	144.69
3	133.47	141.52	126.46	93.32	155.54	125.59	137.04	139.26	117.40	138.88
4	160.55	143.65	155.13	144.53	104.95	148.92	146.46	150.12	152.08	158.21
4	148.73	147.73	152.77	147.85	122.75	147.95	143.39	154.48	150.14	154.50
4	164.61	151.52	161.99	148.95	104.15	149.03	140.15	170.63	147.84	164.13
5	131.77	153.98	152.44	128.63	159.77	118.64	134.40	152.01	138.76	150.20
5	148.54	146.93	149.03	137.46	159.57	126.34	134.03	144.47	145.20	155.65
5	137.99	152.41	146.08	119.15	159.30	103.07	119.61	155.76	121.81	151.83
6	115.57	163.83	157.10	128.47	155.44	116.66	114.42	163.59	121.77	146.10
6	133.26	150.93	153.87	131.80	148.81	126.72	126.27	152.84	132.51	149.01
6	142.04	161.73	162.59	137.91	146.27	119.91	113.65	171.60	123.26	157.94
7	169.68	134.52	146.90	151.88	163.27	161.61	169.63	95.85	167.20	159.09
7	168.73	139.48	150.32	151.91	148.16	161.40	169.61	119.72	168.85	154.19
7	160.33	144.46	139.82	144.92	155.00	161.66	175.46	120.91	161.45	140.23
8	130.94	161.65	151.67	119.82	153.13	126.46	128.38	161.56	101.48	146.16
8	128.16	150.86	141.81	119.39	145.97	130.68	131.00	145.76	113.13	131.82
8	117.76	170.60	143.24	122.78	163.10	132.87	129.52	169.45	100.44	129.80
9	117.86	159.59	141.70	138.84	163.56	138.87	148.27	160.57	140.92	117.74
9	133.55	146.36	141.87	138.01	155.22	140.88	149.12	140.97	144.32	122.53
9	130.35	164.12	141.31	141.95	170.83	146.55	155.85	156.58	137.94	108.07

Table 6.17 shows both feature extraction methods (‘one to many’ and ‘many to one’) in HCN-I.

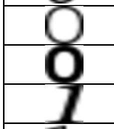
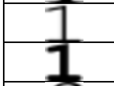
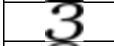
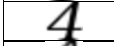
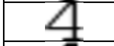
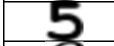
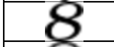
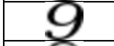
Table 6.17: Performance rate of HCN-I using a ‘mean of distance’ similarity measure

Layer	Performance Rate (%) of HCN-I using ‘one to many’	Performance Rate (%) of HCN-I using ‘many to one’
1	97	97
2	37	10
3	53	17
4	50	13
Average distance of all layers	97	93

Both methods give the best performance rate in Layer 1 at 97%. When calculating the average similarity rate in all layers, the ‘one to many’ method gives a better rate (97%) than the ‘many to one’ method (93%).

A ‘mean of distance’ distance measure is implemented in HCN-II using both the ‘one to many’ and ‘many to one’ feature extraction method. The similarity rates for the tested patterns and all available classes in HCN-II using ‘one to many’ are shown in Table 6.18.

Table 6.18: The results of classification in HCN-II using a 'mean of distance' distance measure

Tested	Similarity Rate to the Predicted Class (%)									
Images	0	1	2	3	4	5	6	7	8	9
	96.739	164.7	151.46	133.3	167.13	133.6	134.63	167.17	135.82	136.34
	120.82	158.76	148.47	137.05	171.68	148.67	144.92	155.24	148.48	144.21
	86.678	173.64	147.31	131.8	170.03	137.71	132.72	173.85	131.36	130.01
	161.92	129.94	132.09	137.63	131.13	155.11	164.3	124.73	157.1	145.68
	174.66	131.43	149.9	160.03	159.43	162.22	167.16	137.76	174.62	165.32
	178.07	123.96	144.23	158.48	162.59	160.1	166.9	140.47	169.69	165.83
	139.17	143.04	98.654	133.7	161.34	146.83	155.88	138.13	145.65	137.3
	153.32	140.66	114.3	141.6	158.18	157.11	164.65	127.27	158.74	143.03
	143.6	149.34	90.543	136.23	168.73	155.06	165.55	141.95	149.74	134.72
	132.94	149.42	139.42	105.38	159.18	126.05	137.53	150.73	126.24	151.09
	142.8	136.36	133.81	114.62	153.76	136.03	143.23	131.98	136.06	144.69
	133.47	141.52	126.46	93.321	155.54	125.59	137.04	139.26	117.4	138.88
	160.55	143.65	155.13	144.53	104.95	148.92	146.46	150.12	152.08	158.21
	148.73	147.73	152.77	147.85	122.75	147.95	143.39	154.48	150.14	154.5
	164.61	151.52	161.99	148.95	104.15	149.03	140.15	170.63	147.84	164.13
	131.77	153.98	152.44	128.63	159.77	118.64	134.4	152.01	138.76	150.2
	148.54	146.93	149.03	137.46	159.57	126.34	134.03	144.47	145.2	155.65
	137.99	152.41	146.08	119.15	159.3	103.07	119.61	155.76	121.81	151.83
	115.57	163.83	157.1	128.47	155.44	116.66	114.42	163.59	121.77	146.1
	133.26	150.93	153.87	131.8	148.81	126.72	126.27	152.84	132.51	149.01
	142.04	161.73	162.59	137.91	146.27	119.91	113.65	171.6	123.26	157.94
	169.68	134.52	146.9	151.88	163.27	161.61	169.63	95.849	167.2	159.09
	168.73	139.48	150.32	151.91	148.16	161.4	169.61	119.72	168.85	154.19
	160.33	144.46	139.82	144.92	155	161.66	175.46	120.91	161.45	140.23
	130.94	161.65	151.67	119.82	153.13	126.46	128.38	161.56	101.48	146.16
	128.16	150.86	141.81	119.39	145.97	130.68	131	145.76	113.13	131.82
	117.76	170.6	143.24	122.78	163.1	132.87	129.52	169.45	100.44	129.8
	117.86	159.59	141.7	138.84	163.56	138.87	148.27	160.57	140.92	117.74
	133.55	146.36	141.87	138.01	155.22	140.88	149.12	140.97	144.32	122.53
	130.35	164.12	141.31	141.95	170.83	146.55	155.85	156.58	137.94	108.07

The distance values in Table 6.18 show the distances between the tested patterns and the available classes in Layer 1. The 'one to many' method in HCN-II gives the highest performance rate (97%) in Layer 1 compared to the other layers (see Table 6.19).

Table 6.19: Performance rate of HCN-II in all layers using a ‘mean of distance’ distance measure

Activation Threshold (%)	Performance Rate (%)									
	HCN-II (‘one to many’)					HCN-II (‘many to one’)				
	L1	L2	L3	L4	Average	L1	L2	L3	L4	Average
0	97	93	67	53	77	97	93	40	33	67
0.1	97	93	67	53	77	97	93	37	33	67
0.2	97	93	70	60	83	97	93	33	33	67
0.3	97	93	73	60	83	97	93	40	33	70

In Table 6.19, both methods give the same performance rate in Layers 1 and 2 at 97% and 93%, respectively; however, the ‘one to many’ method gives a better performance rate on the average of all layers.

As illustrated in Figure 6.8, ‘minimum distance’ chooses the closest distance between a tested pattern and all available reference patterns in a class

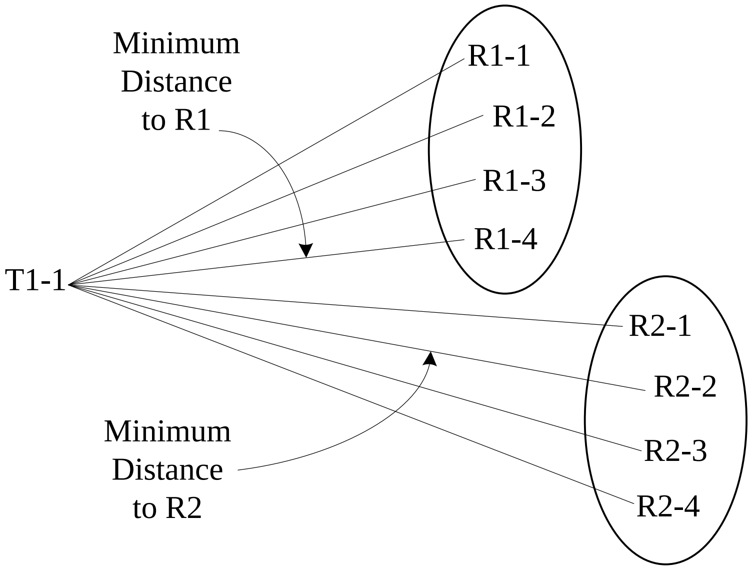


Figure 6.9: An illustration of ‘minimum distance’

Similar to the ‘mean of distance’ concept, ‘minimum distance’ measures the distance between a tested pattern and all the patterns in a class. If ‘mean of distance’ calculates the average distance, ‘minimum distance’ chooses the minimum value of distance between the tested pattern and the class. For example, in Figure 6.9, the shortest distance between T1-1 and Class R1 is the distance between T1-1 and R1-4, while the minimum distance between T-1 and Class R2 is the distance between T1-1 and R2-4.

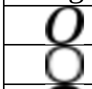
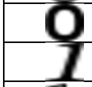
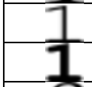
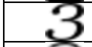
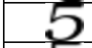
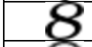
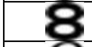
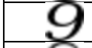
The distance measure algorithm using ‘minimum distance’ is presented in Algorithm 6.6.

Algorithm 6.6: ‘Minimum Distance’

```
1: For all tested patterns (n_TPat), start a class from 1
2: For all reference patterns (n_RPat) within the current class, calculate the
   distance in all layers (dLn) between the tested pattern (TPat) and each
   reference pattern (RPat) within the current class:

       $dLn = \sqrt{(TPat - RPat)^2}$        (6.13)

3: Initial the distance at the current layer (distLn) with the distance of the
   tested pattern and the first pattern in the current class (dLn)
4: From the second pattern to the last pattern in the current class, replace the
   current distLn with the minimum value of the distance
5: Increase the class value, and repeat from Line 2
```

The similarity rates of classification in Table 6.20 refer to the performance rate in Layer 1 of HCN-I using the ‘one to many’ method. This layer has the highest classification rate when compared to the other layers (see Table 6.21).

Table 6.20: The results of classification in HCN-I using a 'minimum distance' distance measure

Tested Images	Similarity Rate to the Predicted Class (%)									
	0	1	2	3	4	5	6	7	8	9
	19.87	42.45	40.64	34.71	47.55	36.66	33.34	48.24	36.19	35.23
	31.03	45.87	41.29	37.87	53.30	43.04	37.50	49.78	40.25	36.42
	14.43	45.03	38.93	34.13	47.68	35.92	29.35	49.73	31.42	29.58
	45.59	28.47	35.67	37.25	37.56	42.46	46.11	35.04	42.38	39.45
	43.97	29.40	36.24	40.96	43.30	41.56	42.27	38.05	44.92	41.73
	48.28	22.63	41.35	44.26	45.64	45.38	46.46	40.22	47.25	44.59
	40.16	43.69	24.30	37.93	48.92	42.31	44.49	42.59	39.27	38.84
	41.00	42.09	27.49	39.11	46.83	44.83	47.37	37.13	44.18	38.58
	42.60	44.42	16.21	40.43	49.63	45.41	46.13	42.74	43.77	37.93
	36.17	41.25	40.19	26.78	46.11	35.36	38.21	48.37	31.23	42.49
	37.40	38.54	39.06	27.32	44.71	37.70	38.60	40.82	33.56	39.04
	37.95	41.14	36.25	23.62	46.11	35.03	37.63	42.85	33.02	37.90
	46.59	39.65	44.97	42.17	30.09	43.08	39.06	44.68	43.73	44.15
	43.87	44.27	45.95	45.34	35.11	43.18	39.08	47.90	44.07	45.38
	48.77	43.36	48.52	45.45	30.47	44.21	40.22	53.28	43.01	48.97
	34.40	41.97	43.55	36.00	46.21	29.49	37.26	44.74	38.28	37.26
	39.76	42.14	41.40	37.13	46.93	29.23	33.99	43.27	36.91	41.88
	39.35	44.77	39.03	32.26	45.83	20.30	31.54	47.15	33.16	38.38
	31.31	47.16	44.04	34.40	47.02	31.05	29.75	51.17	32.06	42.13
	35.30	45.45	44.21	34.81	44.16	33.24	34.87	47.46	35.03	43.00
	42.59	48.58	44.93	37.97	42.76	30.37	29.16	51.84	35.81	41.66
	47.22	31.82	39.53	43.67	47.57	45.60	49.48	24.04	47.32	42.85
	44.73	38.94	38.95	40.33	40.90	42.78	45.51	32.64	44.29	37.67
	45.25	38.81	39.90	41.02	43.96	46.99	48.80	33.16	45.38	35.63
	37.16	49.94	45.66	34.49	46.82	37.48	37.69	51.49	26.30	40.35
	32.75	48.55	43.30	34.58	44.81	37.57	37.44	48.13	27.80	38.04
	30.87	51.19	40.15	33.04	49.53	37.92	32.93	51.50	25.33	33.30
	31.18	44.87	41.24	39.22	50.04	41.53	42.77	49.75	40.32	32.26
	35.63	43.27	39.77	39.09	45.91	39.87	43.26	43.09	39.00	33.36
	38.80	49.05	39.41	38.91	51.20	42.22	42.62	46.87	38.52	27.10

The average distance in all layers of HCN-I using the 'one to many' method gives a classification rate of 93%. Two digits (the second '6', and the first '9') are placed into the incorrect class. Other layers have their own classification rate as shown in Table 6.21.

Table 6.21: Performance rate of HCN-I using a ‘minimum distance’ similarity measure

Layer	Performance Rate (%) of HCN-I using ‘one to many’	Performance Rate (%) of HCN-I using ‘many to one’
1	90	90
2	57	10
3	NA	NA
4	NA	NA
Average distance of all layers	93	90

The performance rates in Layers 3 and 4, which are labelled as ‘NA’ (not available), are not able to be recorded. This is because some tested patterns are classified into more than one predicted class. In other words, these patterns share the minimum distance with more than one class.

According to Table 6.21, the average distance of all layers in HCN-I using the ‘many to one’ method is 3% lower than when the ‘one to many’ method is used in HCN-I.

Classification results for HCN-II using a ‘minimum distance’ distance measure are shown in Table 6.22.

Table 6.22: The results of classification in HCN-II using a ‘minimum distance’ distance measure

Tested Images	Similarity Rate to the Predicted Class (%)									
	0	1	2	3	4	5	6	7	8	9
0	74.518	229.73	217.51	181.31	251.03	187.72	168.16	268.93	184.17	167.93
0	132.48	217.57	214.9	191.59	275.31	214.85	175.27	251.53	217.97	173.07
0	58.06	278.13	244.61	209.12	284.71	215.48	162.34	307.3	194.69	170.02
1	262.64	142.7	200.75	213.36	203.37	240.37	272.52	196.29	247.25	216.97
1	235.93	122.36	192.28	211.62	199.41	213.04	214	200.14	244.87	221.22
1	296.07	89.213	252.28	264.44	257.46	273.82	273.18	236.61	289.56	278.54
2	199.44	241.75	93.124	191.05	258.54	216.46	241.11	231.43	201.81	208.06
2	192.06	221.97	113.39	187.11	235.61	210.66	241	165.07	222.29	187.83
2	238.08	238.54	65.169	209.62	288.54	253.64	269.69	226.96	253.68	215
3	167.79	193.88	183.08	100.15	224.18	160.29	184.17	229.14	132.16	206.2
3	173.2	185.79	184.38	114.67	226.13	164.5	188.74	183.73	174.13	202.57
3	199.36	215.76	180.4	88.888	243.71	175.87	205.96	222.38	181.07	206.16
4	246.7	204.01	242.55	221.38	133.49	232.13	205.79	239.19	234.9	225.2
4	218.98	236.18	235.54	220.47	174.69	219.61	200.03	247.47	224.12	233.05
4	272.85	228.49	267.61	242.35	129.17	240.2	205.41	302.05	223.28	275.68
5	168.58	222.26	230.52	177.38	249.54	131.13	195.89	231.66	194.43	196.1
5	178.8	219.62	207.59	155.75	232.61	118.08	171.37	209.17	182.49	213.6
5	217.58	256.96	215.82	165.87	254.03	76.335	173.19	261.17	180.65	208.53
6	134.29	255.53	229.27	161.78	241.42	143.44	122.88	263.02	142.01	217.5
6	150.36	230	205.89	154.05	211.4	126.74	140.31	226.76	166.51	208.47
6	220.23	275.9	248.04	204.59	226.26	133.06	131.81	297.96	170.16	212.87
7	250.59	162.93	219.53	227.95	266.15	240.9	279.75	96.016	256.17	229.7
7	240.17	226.13	206.99	215.04	216.87	233.6	259.48	168.78	251.55	189.17
7	269.88	226.21	230.03	244.48	255.74	271.23	297.99	169.44	269.05	200.92
8	172.77	261.47	222.31	157.48	224.98	160.5	165.44	259.48	103.97	186.05
8	127.97	234.3	197	135.67	200.91	130.08	155.35	215.78	110.94	155.19
8	155.55	292.32	205.59	150.81	266.08	188.07	149.75	286.79	100.57	156.07
9	130.45	226.7	197.64	188.77	237.48	197.81	210.73	254.87	186	128.96
9	157.89	224.63	193.04	185.66	227.17	174.73	210.78	205.95	186.36	140.41
9	198.15	271.44	208.1	209.11	282.72	225.04	223.33	256.65	201.94	104.07

When using a ‘minimum distance’ distance measure with a threshold value of 0.3, HCN-II (using ‘one to many’) can recognise the tested pattern at a 97% performance rate in Layer 3. The ‘many to one’ method gives the highest performance rate (90%) when using the average distance with the threshold value of 0 to 0.3.

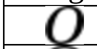
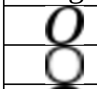
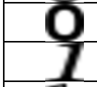
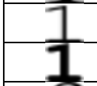
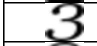
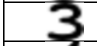
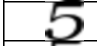
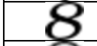
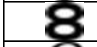
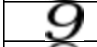
Table 6.23 shows that certain tested patterns share the minimum distance with more than one class. These patterns are labelled ‘NA’.

Table 6.23: Performance in all layers using a 'minimum distance' distance measure

Activation Threshold (%)	Performance Rate (%)									
	HCN-II ('one to many')					HCN-II ('many to one')				
	L1	L2	L3	L4	Average	L1	L2	L3	L4	Average
0	0	90	93	NA	NA	90	93	NA	NA	90
0.1	0.1	90	93	67	73	90	93	NA	NA	90
0.2	0.2	90	93	77	80	90	93	NA	NA	90
0.3	0.3	90	97	80	87	90	93	53	57	87

The example of a tested pattern belongs to more than one class is shown in Table 6.24.

Table 6.24: Multiple minimum distances when using a 'minimum distance' distance measure

Tested Images	Similarity Rate to the Predicted Class (%)									
	0	1	2	3	4	5	6	7	8	9
	54.791	214.11	171.55	161.24	192.82	197.36	190.83	225.81	231.74	219.98
	70.718	264.27	214.82	205.27	209.85	192.24	223.17	246.43	258.03	231.74
	0	141.61	143.14	137.16	141.73	107.88	105.58	194.54	171.46	108.66
	404.61	264.03	222.3	258.43	225.44	349.78	358.11	190.07	441.64	372.7
	447	242.45	384.12	371	384.27	349.93	347.05	321.38	460.91	422.19
	350.41	0	218.01	227.38	207.46	211.01	245.5	180	303.59	312.13
	244.92	205.04	20.149	203.85	193.3	218.3	258.6	157.65	270.72	232.4
	326.87	299.14	167.96	285.28	277.35	337.13	340.94	215.37	368.46	298.12
	196.76	117.2	25.239	146.28	151.83	205.08	244.5	95	231.52	113.71
	208.75	189.43	182.61	88.165	192.7	158.46	224.39	145.94	233.94	304.73
	328.04	312.85	265.11	189.26	247.57	263.59	308.87	193.79	395.46	344.52
	232.78	139.45	95.026	48.218	189.95	174.74	216.18	116.85	275.57	242.29
	420.45	344.31	315.85	293.86	134.63	323.23	291.45	208.09	360.77	386.19
	365.11	272.92	305.29	292.79	143.14	285.12	237.5	242.45	330.13	341.8
	306.04	214.13	211.31	161.71	8	137.18	88.363	129.29	187.22	331.27
	163.81	226.1	192.63	58.771	192.62	95.514	196.8	142.1	223.6	241.34
	313.13	313.93	336.32	228.99	318.86	185.65	253.57	266.71	368.86	341.36
	183.56	132.55	155.48	31.828	171.82	33.734	141.51	111.03	200.29	236.69
	129.03	271.33	194.11	129.17	171.03	104.88	107.9	123.05	171.39	249.26
	269.03	306.36	330.63	245.95	268.3	201.98	186.99	246.35	287.65	320.81
	143.77	77.827	137.61	91.897	75	61.205	36	35.228	125.1	177.89
	415.3	251.92	263.78	365.8	370.35	383.76	419.23	16.155	459.07	310.34
	483.04	424.96	346.16	397.19	365.32	437.93	494.55	264.58	561	352.62
	375.05	302.32	264.68	265.81	240.09	322.93	415.95	194.81	442.99	226.13
	112.89	224.69	118.14	80.474	85.294	62.418	66.858	109.64	90.615	193.4
	175.57	241.46	185.33	167.24	152.96	153.87	189.9	176.16	219.49	180.96
	0	3.1623	2.2361	22.361	0	1	29.614	42	32	43.186
	144.4	226.64	108.8	163.79	167.29	193.57	263.62	195.14	248.2	121.38
	267.57	293.24	226.47	251.51	254.65	250.17	321.15	263.86	349.94	172.35
	127.01	71.694	40.719	53.749	107.35	80.075	161.48	79	142.67	0

The green cells in Table 6.24 indicate when the distance minimum value can be belong to more than one class. In Table 6.24, the third ‘8’ gives a minimum distance of ‘0’ with its own class and also Class 4. A decision cannot be made using these ‘NA’ results.

6.5. Discussion and Validation

This chapter presents a summary of the classification rates in both HCN-I and II, and compares the results with existing methods. The research in this thesis explores the many different concepts of feature extraction and recognition, combining or concatenating the lower level’s features as the features of the higher level. The ability of hierarchical concatenation is investigated using classification performance results.

The hierarchical concatenation network (HCN) seeks to mimic human behaviour when recognising patterns or shapes. It examines the smaller features first of all; next, it investigates the smaller features’ concatenation; then finally, it makes a decision based on the resulting higher level features.

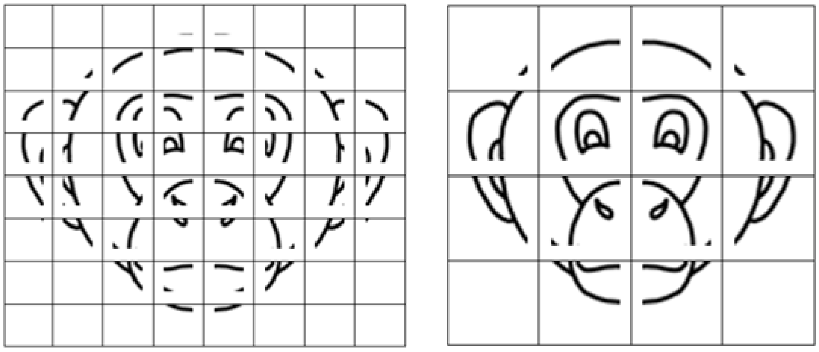


Figure 6.10: Features in a hierarchical examination

The left picture in Figure 6.10 illustrates the smaller features, while the right picture shows the concatenation of the features on the left as bigger features (one block rectangle in the right picture is the concatenation of four small rectangles in the left). In the left picture, each single block might have a high similarity rate with other blocks of picture of a fruit; whereas each block in the right picture more clearly represents a feature of a monkey’s head. This means, to recognise the pattern more efficiently, the decision should be made at the end of the concatenation process (at the top layer).

This research investigates the performance rate of classification by exploring different activation methods (i.e. HCN-I and HCN-II) and using various ways to extract the features (i.e. ‘one to many’ and ‘many to one’ methods). The results show that the best classification rate is not based on the output of the top layer as supposed. Table 6.25 shows the summary of classification results using the digits dataset.

Table 6.25: Performance rate of classification summary in the HCN

Feature Extraction Method	Feature Extraction Position	Similarity Measure	Activation Threshold (%)	Performance Rate (%)				
				L1	L2	L3	L4	Ave
HCN-I	‘One to any’	Ratio by Union	NA	67	17	47	63	83
		Ratio by Average	NA	70	37	67	53	93
		Distance to Mean	NA	97	43	67	50	100
		Mean of Distance	NA	97	37	53	50	97
		Minimum Distance	NA	90	57	NA	NA	NA
	‘Many to One’	Ratio by Union	NA	NA	0	3	NA	13
		Ratio by Average	NA	87.01	32.15	26.91	17.97	41.01
		Distance to Mean	NA	97	13	13	13	90
		Mean of Distance	NA	97	10	17	13	93
		Minimum Distance	NA	90	10	NA	NA	90
HCN-II	‘One to Many’	Ratio by Union	0	73	60	13	17	27
			0.1	73	67	7	17	33
			0.2	73	67	7	23	37
			0.3	73	70	13	21	40
		Ratio by Average	0	60	73	33	50	50
			0.1	60	70	33	47	50
			0.2	60	70	33	53	53
			0.3	60	70	33	53	53
		Distance to Mean	0	97	100	77	57	90
			0.1	97	100	80	57	90
			0.2	97	100	83	63	93
			0.3	97	100	90	70	93
		Mean of Distance	0	97	93	67	53	77
			0.1	97	93	67	53	77
			0.2	97	93	70	60	83
			0.3	97	93	73	60	83
		Min Distance	0	90	93	NA	NA	NA
			0.1	90	93	67	73	87
			0.2	90	93	77	80	87
			0.3	90	97	80	87	87
	‘Many to One’	Ratio by Union	0	80	60	30	NA	70
			0.1	80	33	30	NA	67
			0.2	80	23	20	NA	60
			0.3	80	30	20	NA	60
		Ratio by Average	0	60	73	20	47	83
			0.1	60	70	20	47	83
			0.2	60	70	20	50	80
			0.3	60	70	27	50	80
		Distance to Mean	0	97	100	43	40	77
			0.1	97	100	47	40	77
			0.2	97	100	40	40	73
			0.3	97	100	40	40	80
		Mean of Distance	0	97	93	40	33	67
			0.1	97	93	37	33	67
			0.2	97	93	33	33	67

Feature Extraction Method	Feature Extraction Position	Similarity Measure	Activation Threshold (%)	Performance Rate (%)				
				L1	L2	L3	L4	Ave
			0.3	97	93	40	33	70
		Min Distance	0	90	93	NA	NA	90
			0.1	90	93	NA	NA	90
			0.2	90	93	NA	NA	90
			0.3	90	93	53	57	87

Table 6.25 indicates that both HCN activation methods give their best classification results when using the ‘distance to mean’ similarity measure. HCN-I gives the best results when calculating the average similarity rate in all layers, while HCN-II recognises all the tested images in Layer 2. Both methods are presented in Chapter 5, where it can also be seen, however, that HCN-I takes more time to extract the features compared to HCN-II. Based on time consumption and the results in Table 6.25, HCN-II (which only contains two layers) is best able to represent the hierarchical concatenation network.

Table 6.26 presents some other comparisons with two other datasets (USPS and MNIST). Due to the input of Hierarchical Concatenation Network is the image with the size of 32 x 32 pixels, the images from these datasets need to suit the network input. As the original images from USPS dataset are drawn on 16 x 16 pixels, they can be fed into the second layer of HCN. This means that the first layer of HCN has 16 x 16 pixels input, hence the network will have only three layers. MNIST dataset contains images which are drawn on 28 x 28 pixels. The images from this dataset cannot be fed directly into HCN at any layer. For this reason, they must be normalized by changing the size of the image to be 16 x 16 pixels or 32 x 32 pixels. In this thesis, the size of images in this dataset were resized to 32 x 32 pixels. The size of 16 x 16 pixels is to much reduction from 28 x 28 pixels.

Table 6.26: Performance Rate Comparison

Datasets	Feature Extraction + Classification	Performance Rate (%)
Digits (100 data)	HCN	100
	Binary vector	90
	LDA	96.67
	PCA	96.67
USPS (100 data)	HCN	70
	Binary vector	33
	LDA	50
	PCA	43.33
MNIST (100 data)	HCN	67
	Binary vector	23
	LDA	50
	PCA	56.67
USPS (1000 data)	HCN	80
	Binary vector	55
	LDA	81.33
	PCA	74.67
MNIST (1000 data)	HCN	82
	Binary vector	53
	LDA	89.33
	PCA	53.67

HCN algorithm in Table 6.26 refers to HCN-II with the ‘many to one’ method due to it gives a better performance rate and consumes less time than HCN-I (see Table 6.25).

The comparison in Table 6.26 uses binary vectors, LDA and PCA. Binary feature (vector of binary) extraction is the simplest feature extraction method. Each feature is extracted to be a vector of binary numbers. The classification process uses Euclidean distance to measure the distance between the tested and reference patterns. With it simple comparison process, binary vector extraction gives the lowest classification rates when compared to others.

LDA and PCA are two other methods that can be used to compare the classification rates of HCNs. Both LDA and PCA are feature extractions methods, also known as dimension reduction methods (Zheng, Lai, & Li, 2008). Table 6.26 indicates that LDA has a higher performance rate than the HCN for handwritten datasets with 1000 samples. In contrast, if small amount of datasets (100 samples) are applied to LDA, it gives lower performance rate compared with HCN.

By using simple Digits dataset, HCN has shown its performance rate by 100%, while LDA and PCA has given the same rate at 96.67%. By using USPS dataset (100 samples), HCN's classification rate is 20% higher than LDA, and 26.67% higher than PCA. For MNIST dataset (100 samples), the HCN's performance is still higher than LDA and PCA, 17% and 10.33% respectively. Once the dataset becomes bigger (1000 samples), LDA is better than HCN for both MNIST and USPS dataset (1.33% and 7.33% respectively), but HCN still outperforms PCA by 5.33% for USPS data set and by 28.33% for MNIST dataset. The performance rate of the HCN is raised when the number of datasets increases. By increasing the number of samples from 100 to 1000, the performance rate is increased by more than 10 times. This indicates that HCN could produce a good result with small samples of dataset in terms of classification. Even though the classification rates could be increased by making the dataset bigger, HCN consumes much time to finish the process compared with others. It is due to the feature extraction process and similarity measure conducted in all layers.

To conclude this chapter, HCN performance can be increased by increasing the number of datasets. HCN-II is representative of the hierarchical concatenation network for feature extraction, and the 'distance to mean' similarity measure is best suited for the classification process. The study of HCN algorithm has achieved its ability to classify two types of dataset (artificial handwritten digits datasets) to represent feature extraction and concatenation from the computers term with the consideration of human behaviour in seeing and combining features of objects.

Chapter 7

Conclusions and Future Work

7.1. General Conclusion

The focus in this thesis was to design, implement, and evaluate the proposed algorithm called HCN (Hierarchical Concatenation Network) that tried to follow the humans' behaviour in recognizing simple patterns. This was achieved by describing humans' perspective when they see the differences between two images. The process is to compare each feature that constructs the whole image. It was then followed by proposing the same process that suits with the computers term, considering position exchange between the tested and reference images for feature extraction process, and finding the similarity measure that suitable for classification. The algorithm for activating the upper layers' features was proposed by assigning value '1' for active features. This feature activation method is then called as HCN-I. This algorithm was then modified by involving the weight of features in upper layers' features activation, called HCN-II. Shifting the image and involving the middle area of image during scanning process were considered to influence the result of similarity measure. A series of experiments were performed based on the position of the tested and reference patterns ("many to one" and "one to many"), the feature activation process (HCN-I and HCN-II), the implementation of sequential and non-sequential image shifting, and overlapping and non-overlapping the middle area of image to measure the similarity between the tested and reference patterns. Finally, several classification methods were evaluated to perform suitable classification process within HCN. A summary of contribution made in this thesis are outlined, followed by suggestion of future work in the field.

7.2. Research Contributions

The literature review in Chapter 2 reveals the significant gaps in feature concatenation and activation studies. In this thesis, the contribution were made in the following area areas:

1. Outside of this thesis, there is no investigation into feature extraction using other features as the reference and no support on how to represent upper layers' features. However, Chapter 4 (Sections 4.1 and 4.2) of this study explores this. It can be seen that before conducting feature extraction, the input image can be processed. When inputs are binarised, the reference patterns can extract their own features, in all layers, while the tested patterns use the reference patterns' features to extract features in Layers 2 to 4. Feature extraction from both the reference and tested pattern is carried out with the following steps:
    a. The binary image is extracted using a coincidence array in each layer.
    b. The activation groups of the reference pattern are constructed within the coincidence array of the upper layer.
    c. In HCN-I, the upper layer features of the reference pattern are represented by decimal numbers, converted from the number of vector bits in the activation group. The features of the tested pattern, in the upper layers, are the product of feature activation, and the tested pattern's features are activated based on the number of similar bits within the vector of the activation group. If a tested pattern has a feature that is similar to a feature within the activation group, the corresponding bit in the activation group is activated.
    d. In HCN-II, the activation group involves the occurrence of features within the upper layers' coincidence array. The activation group therefore contains unique features that are multiplied by their occurrences.

2. As the proposed feature extraction uses other pattern's features to extract features of a pattern, there is no supporting information that the exchanging position between the tested and reference pattern could lead different result.

The position exchange increases the similarity rate between the tested and reference patterns, and in this research, two methods of pattern exchange are investigated: the ‘one to many’ method and the ‘many to one’ method.

a. The ‘one to many’ method extracts the features of the reference pattern from the tested pattern’s side. In other words, the tested pattern is used to extract the features of the reference pattern.
b. The ‘many to one’ method uses the reference pattern as the reference for extraction of the tested pattern’s features.
c. The preferred HCN feature extraction method is selected as HCN-II using the ‘many to one’ extraction method. Results show that this method gives better results than HCN-I when using either the ‘many to one’ or ‘one to many’ method.

3. There is depth investigation on the number of layer of the network.

Majority of previous works use fixed number of layer in the network. In this thesis, the network was designed to have four layers, as the output will be the concatenation of four nodes on layer 4. It is important to know whether HCN should have four layers or not. Before the classification method was constructed, the experiment on similarity measured was investigated on each layer (see section 4.3). A tested pattern will have 100% similar to itself on all layers, but its similarity rate will decrease from layers 2 to 4 when the reference patterns are other patterns. Even though other patterns looks similar from human point of view. During the construction of classification method, the classification rate on each layer was measured. Table 6.25 on section 6.5 shows that HCN can achieve its best performance with only two layers.

4. This study investigates classification rates using a ‘union’ operation that follows the way humans group the features of patterns into classes.
   a. In this thesis, ‘classification by union’ is conducted by grouping the features of the reference patterns in the same class so that each reference class ends up having a unique representative feature. To measure the similarity of a tested pattern against each of the classes, the similarity rate between the features of the tested pattern and each available class is calculated.

Classification into a class is then decided by assigning the highest rate of similarity.

b. ‘Classification by average’, on the other hand, calculates the mean of similarity between a tested pattern and all members of a class. This method gives better results in terms of similarity using HCN-I.
c. A standard Euclidean distance is used in the experiments. The ‘distance to mean’ measure, which measures the distance between a tested pattern and the mean value of the reference pattern’s features within a class, gives better results than both the ‘mean of distance’ measure and the ‘minimum distance’ measure. The best results are implemented in HCN-II using the ‘many to one’ method. ‘Mean of distance’ is the average distance between a tested pattern and all patterns in a class. ‘Minimum distance’ is the minimum distance between a tested pattern and a member of one class.

7.2.1. Summary of Major Contributions

The major research contributions in this thesis are listed below:

a. Design of a hierarchical concatenation network
b. Implementation of scanning algorithms on a network’s layers
c. Two new approaches to extraction of a pattern with another pattern’s features, namely the ‘one to many’ method and the ‘many to one’ method
d. Implementation of two new feature activation systems called HCN-I and HCN-II
e. Investigation into different similarity measures for classification purposes, using union, average, and distance measures

7.3. Suggestion for Future Work

The results in this thesis give some direction for future researchers to replicate the way humans recognise patterns or objects. The most important idea to come out of this research is the need for significant development into replicating the way humans concatenate the smaller parts of the object being recognised. Also, because the hierarchical concatenation network in this thesis represents features as numbers, future development of the algorithm may instead look at representing these features as standard geometry shapes such as lines, curvatures, circles, squares, etc.

A work conducted by (Y. Zhang, Cao, & Wang, 2015) shows that a face recognition algorithm based on local binaries and implemented on a Virtex-7 FPGA has a recognition speed 74 times faster than a software version implemented on a PC platform. Tapiador, Rios-Navarro et al. (2016) investigate the time consumption difference between a Virtex-7 FPGA and a Titan X GPU to run a binary convolutional neural network algorithm. The results show that the FPGA is 8.3 times faster than the GPU. For the algorithm proposed in this thesis, future extraction is computationally expensive; future works could therefore investigate hardware implementation of the HCN algorithm.

As we have seen, replicating human intelligence is not an easy task. Researchers from many different areas, such as psychology, mathematics, computer science, electronics, etc., should be involved in development of the field. The interaction of researchers from different disciplines could reveal new opportunities for progress in this challenging but fascinating area.

Bibliography

Afzal, M. Z., Capobianco, S., Malik, M. I., Marinai, S., Breuel, T. M., Dengel, A., & Liwicki, M. (2015, 23-26 Aug. 2015). *Deepdocclassifier: Document classification with deep Convolutional Neural Network.* Paper presented at the 2015 13th International Conference on Document Analysis and Recognition (ICDAR).

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Hasan, M., Esesn, B. C. V., . . . Asari, V. K. (2018). The History Began from AlexNet: {A} Comprehensive Survey on Deep Learning Approache. *CoRR, abs/1803.01164*.

Amin, H., & Fujii, R. (2004, 25-29 July 2004). *Spike train decoding scheme for a spiking neural network.* Paper presented at the 2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541).

Amin, H. H., & Fujii, R. H. (2005, 2005//). *Learning Algorithm for Spiking Neural Networks.* Paper presented at the Advances in Natural Computation, Berlin, Heidelberg.

and, K. H., and, X. Z., and, S. R., & Sun, J. (2016). Identity Mappings in Deep Residual Networks. *CoRR, abs/1603.05027*.

and, K. S., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR, abs/1409.1556*.

Baev, K. V. (1998). Learning in Artificial and Biological Neural Networks. In *Biological Neural Networks: Hierarchical Concept of Brain Function* (pp. 110-125). Boston, MA: Birkhäuser Boston.

Balaban, P. M., & Gulyaeva, N. V. (2016). Neurophilosophy. *Neuroscience and Behavioral Physiology, 46*(9), 1078-1081. doi:10.1007/s11055-016-0354-2

Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19*(7), 711-720. doi:10.1109/34.598228

Benchenane, K., Peyrache, A., Khamassi, M., Tierney, P. L., Gioanni, Y., Battaglia, F. P., & Wiener, S. I. (2010). Coherent Theta Oscillations and Reorganization of Spike Timing in the Hippocampal- Prefrontal Network upon Learning. *Neuron, 66*(6), 921-936. doi:https://doi.org/10.1016/j.neuron.2010.05.013

Bischl, B., Schiffner, J., & Weihs, C. (2013). Benchmarking local classification methods. *Computational Statistics, 28*(6), 2599-2619. doi:10.1007/s00180-013-0420-y

Bolotova, Y., & Spitsyn, V. (2012, 18-21 Sept. 2012). *Analysis of hierarchically-temporal dependencies for handwritten symbols and gestures recognition.* Paper presented at the Strategic Technology (IFOST), 2012 7th International Forum on.

Boone, A. R. W., Karnowski, T. P., Chaum, E., Giancardo, L., Li, Y., & Tobin, K. W., Jr. (2010, 25-26 May 2010). *Image processing and hierarchical temporal memories for automated retina analysis.* Paper presented at the Biomedical Sciences and Engineering Conference (BSEC), 2010.

Cazzanti, L., & Gupta, M. R. (2006, 9-14 July 2006). *Information-theoretic and Set-theoretic Similarity.* Paper presented at the 2006 IEEE International Symposium on Information Theory.

Cha, S.-H. (2000). Efficient Algorithms for Image Template and Dictionary Matching. *Journal of Mathematical Imaging and Vision, 12*(1), 81-90. doi:10.1023/a:1008309026555

Charalampous, K., Kostavelis, I., Amanatiadis, A., & Gasteratos, A. (2012). Sparse deep-learning algorithm for recognition and categorisation. *Electronics Letters, 48*(20), 1265-1266. doi:10.1049/el.2012.1033

Chen, X., Wang, W., & Li, W. (2012, 24-26 June 2012). *An overview of Hierarchical Temporal Memory: A new neocortex algorithm.* Paper presented at the Modelling, Identification & Control (ICMIC), 2012 Proceedings of International Conference on.

Chernyak, D. A., & Stark, L. W. (2001). Top-down guided eye movements. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 31*(4), 514-522. doi:10.1109/3477.938257

Choi, Y., Kim, S., & Lee, J. H. (2016, 25-28 Aug. 2016). *Recurrent Neural Network for Storytelling.* Paper presented at the 2016 Joint 8th International Conference on Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems (ISIS).

Chou, C. H., & Hsu, Y. H. (2011, 3-4 Dec. 2011). *Image Quality Assessment Based on Binary Structure Information.* Paper presented at the 2011 Seventh International Conference on Computational Intelligence and Security.

Clarke, R. J. (1983). Application of sine transform in image processing. *Electronics Letters, 19*(13), 490-491. doi:10.1049/el:19830332

Costa, L. d. F., & Cesar, R. M. (2000). Shape Recognition and Classification. In *Shape Analysis and Classification*: CRC Press.

Crnkovic, I. (2012). Software Engineering Research.

Curtin, R. R. a. C., James R. and Slagle, Neil P. and March, William B. and Ram, P. and Mehta, Nishant A. and Gray, Alexander G. (2013). MLPACK: A Scalable {C++} Machine Learning Library. *Journal of Machine Learning Research, 14*, 801-805.

Daynac, M., Cortes-Cabrera, A., & Prieto, J. M. (2015). Application of Artificial Intelligence to the Prediction of the Antimicrobial Activity of Essential Oils. *Evidence-Based Complementary and Alternative Medicine, 2015*, 9. doi:10.1155/2015/561024

Dougherty, G. (2013). Introduction. In *Pattern Recognition and Classification: An Introduction* (pp. 1-7). New York, NY: Springer New York.

Dragulescu, D., & Albu, A. (2007, Yearly 17 2007-May 18 2007). *Expert System for Medical Predictions.* Paper presented at the 2007 4th International Symposium on Applied Computational Intelligence and Informatics.

Du, X., Cai, Y., Wang, S., & Zhang, L. (2016, 11-13 Nov. 2016). *Overview of deep learning.* Paper presented at the 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC).

Eden, M. (1962). Handwriting and pattern recognition. *IRE Transactions on Information Theory, 8*(2), 160-166. doi:10.1109/TIT.1962.1057695

Ersoy, O. K., & Shi-Wee, D. (1995). Parallel, self-organizing, hierarchical neural networks with continuous inputs and outputs. *IEEE Transactions on Neural Networks, 6*(5), 1037-1044. doi:10.1109/72.410348

Ezoji, M., & Faez, K. (2011). Use of matrix polar decomposition for illumination-tolerant face recognition in discrete cosine transform domain. *IET Image Processing, 5*(1), 25-35. doi:10.1049/iet-ipr.2009.0340

Foster, P., Dixon, S., & Klapuri, A. (2015). Identifying Cover Songs Using Information-Theoretic Measures of Similarity. *IEEE/ACM Transactions on Audio, Speech, and Language Processing, 23*(6), 993-1005. doi:10.1109/TASLP.2015.2416655

Fukushima, K., Miyake, S., & Ito, T. (1983). Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-13*(5), 826-834. doi:10.1109/TSMC.1983.6313076

Gabrielsson, P., Konig, R., & Johansson, U. (2012, 29-30 March 2012). *Hierarchical Temporal Memory-based algorithmic trading of financial markets.* Paper presented at the Computational Intelligence for Financial Engineering & Economics (CIFEr), 2012 IEEE Conference on.

George, D. (2008a). How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition.

George, D. (2008b). How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition. *Ph.D Thesis, Stanford University*.

George, D., & Hawkins, J. (2005a, 31 July-4 Aug. 2005). *A hierarchical Bayesian model of invariant pattern recognition in the visual cortex.* Paper presented at the Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.

George, D., & Hawkins, J. (2005b, 31 July-4 Aug. 2005). *A hierarchical Bayesian model of invariant pattern recognition in the visual cortex.* Paper presented at the Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on.

George, D., & Hawkins, J. (2009). Towards a Mathematical Theory of Cortical Micro-Circuits. *PLoS Computational Biology 5(10): e100032*.

Gold, B. (1959). Machine recognition of hand-sent Morse code. *IRE Transactions on Information Theory, 5*(1), 17-24. doi:10.1109/TIT.1959.1057478

Gotze, J., & Sauer, M. (1993, 19-21 May 1993). *Unitary image transforms and their implementation.* Paper presented at the Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing.

Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., & Schmidhuber, J. (2009). A Novel Connectionist System for Unconstrained Handwriting Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 31*(5), 855-868. doi:10.1109/TPAMI.2008.137

Grimsdale, R. L., Sumner, F. H., Tunis, C. J., & Kilburn, T. (1959a). Pattern recognition by digital computer. *Electrical Engineers, Journal of the Institution of, 5*(51), 160-164. doi:10.1049/jiee-3.1959.0066

Grimsdale, R. L., Sumner, F. H., Tunis, C. J., & Kilburn, T. (1959b). A system for the automatic recognition of patterns. *Proceedings of the IEE - Part B: Electronic and Communication Engineering, 106*(29), 445-446. doi:10.1049/pi-b-2.1959.0287

Gupta, N., & Panchal, V. K. (2011, 24-29 July 2011). *Artificial intelligence for mixed pixel resolution.* Paper presented at the 2011 IEEE International Geoscience and Remote Sensing Symposium.

Gurevich, I. B., & Koryabkina, I. V. (2006). Comparative analysis and classification of features for image models. *Pattern Recognition and Image Analysis, 16*(3), 265-297. doi:10.1134/s1054661806030023

Hasanbelliu, E., Giraldo, L. S., & Príncipe, J. C. (2014). Information Theoretic Shape Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 36*(12), 2436-2451. doi:10.1109/TPAMI.2014.2324585

Hawkins, J., & Blakeslee, S. (2005). On Intelligence.

Hawkins, J., George, D., & Niemasik, J. (2009). Sequence Memory for Prediction, Inference and Behaviour *The Royal Society Biological Science*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016, 27-30 June 2016). *Deep Residual Learning for Image Recognition.* Paper presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Hebb, D. O. (2005). *The organization of behavior: A neuropsychological theory*: Psychology Press.

Hen Hu, Y., & Hwang, J.-N. (2001). Introduction to Neural Networks for Signal Processing. In *Handbook of Neural Network Signal Processing*: CRC Press.

Herman, I. P. (2016). Light, Eyes and Vision. In *Physics of the Human Body* (pp. 731-817). Cham: Springer International Publishing.

Holland, C. D., & Komogortsev, O. V. (2013, 4-7 June 2013). *Complex eye movement pattern biometrics: Analyzing fixations and saccades.* Paper presented at the 2013 International Conference on Biometrics (ICB).

Horwitz, L. P., & Shelton, G. L. (1961). Pattern Recognition Using Autocorrelation. *Proceedings of the IRE, 49*(1), 175-185. doi:10.1109/JRPROC.1961.287787

Huang, G., Liu, Z., & Weinberger, K. Q. (2016). Densely Connected Convolutional Networks. *CoRR, abs/1608.06993*.

Jain, A. K. (1979). A Sinusoidal Family of Unitary Transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-1*(4), 356-365. doi:10.1109/TPAMI.1979.4766944

Jain, R., Murthy, S. N. J., Chen, P. L. J., & Chatterjee, S. (1995, 20-24 Mar 1995). *Similarity measures for image databases.* Paper presented at the Proceedings of 1995 IEEE International Conference on Fuzzy Systems.

Jalil, N. A., Basari, A. S. H., Salam, S., Ibrahim, N. K., & Norasikin, M. A. (2015). The Utilization of Template Matching Method for License Plate Recognition: A Case Study in Malaysia. In H. A. Sulaiman, M. A. Othman, M. F. I. Othman, Y. A. Rahim, & N. C. Pee (Eds.), *Advanced Computer and Communication Engineering Technology: Proceedings of the 1st International Conference on Communication and Computer Engineering* (pp. 1081-1090). Cham: Springer International Publishing.

Jha, N. K. (2008). *Research Methodology*: Abhishek Publications.

Kalmar, A., & Vida, R. (2013, 18-20 Sept. 2013). *Towards context-aware mobile services through the use of Hierarchical Temporal Memory.* Paper presented at the Software, Telecommunications and Computer Networks (SoftCOM), 2013 21st International Conference on.

Katic, D., & Vukobratovic, M. (2003). Neural Network Approach in Robotics. In *Intelligent Control of Robotic Systems* (pp. 21-70). Dordrecht: Springer Netherlands.

Kekre, H. B., & Solanki, J. K. (1978). Comparative performance of various trigonometric unitary transforms for transform image coding. *International Journal of Electronics, 44*(3), 305-315. doi:10.1080/00207217808900821

Khamehchi, E., Rahimzadeh Kivi, I., & Akbari, M. (2014). A novel approach to sand production prediction using artificial intelligence. *Journal of Petroleum Science and Engineering, 123*, 147-154. doi:http://dx.doi.org/10.1016/j.petrol.2014.07.033

King-Sun, F., & Rosenfeld, A. (1976). Pattern Recognition and Image Processing. *IEEE Transactions on Computers, C-25*(12), 1336-1346. doi:10.1109/TC.1976.1674602

Kobayashi, T. (2016, 27-30 June 2016). *Structured Feature Similarity with Explicit Feature Map.* Paper presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Kostavelis, I., Nalpantidis, L., & Gasteratos, A. (2012, 16-17 July 2012). *Object recognition using saliency maps and HTM learning.* Paper presented at the Imaging Systems and Techniques (IST), 2012 IEEE International Conference on.

Kresevic, J. L., Marinovic, W., Johnston, A., & Arnold, D. H. (2016). Time order reversals and saccades. *Vision Research, 125*, 23-29. doi:http://dx.doi.org/10.1016/j.visres.2016.04.005

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Commun. ACM, 60*(6), 84-90. doi:10.1145/3065386

Kuang, P., Ma, T., Chen, Z., & Li, F. (2018). Image super-resolution with densely connected convolutional networks. *Applied Intelligence*. doi:10.1007/s10489-018-1234-y

Kumar, G., & Bhatia, P. K. (2014, 8-9 Feb. 2014). *A Detailed Review of Feature Extraction in Image Processing Systems.* Paper presented at the 2014 Fourth International Conference on Advanced Computing & Communication Technologies.

Lee, C. H., & Yang, H. C. (2007, 2-4 April 2007). *Implementation of Unsupervised and Supervised Learning Systems for Multilingual Text Categorization.* Paper presented at the Information Technology, 2007. ITNG '07. Fourth International Conference on.

Lee, C. S. G., & Lin, C. T. (1992, 18-21 Oct 1992). *Supervised and unsupervised learning with fuzzy similarity for neural-network-based fuzzy logic control systems.* Paper presented at the [Proceedings] 1992 IEEE International Conference on Systems, Man, and Cybernetics.

Lee, H., Kwon, H., Robinson, R. M., & Nothwang, W. D. (2016, 20-25 March 2016). *DTM: Deformable template matching.* Paper presented at the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

Lei, W., Xianbin, W., Xu, J., & Jianguang, Z. (2009, 17-19 Oct. 2009). *Object Recognition Using a Bayesian Network Imitating Human Neocortex.* Paper presented at the Image and Signal Processing, 2009. CISP '09. 2nd International Congress on.

Levialdi, S. (1970). Parallel counting of binary patterns. *Electronics Letters, 6*(25), 798-800. doi:10.1049/el:19700550

Li, H., Peng, S., Zhizhen, C., & Jingjing, W. (2016, 5-8 Aug. 2016). *Image retrieval and classification on deep convolutional SparkNet.* Paper presented at the 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC).

Lisitsa, D., & Zhilenkov, A. A. (2017, 1-3 Feb. 2017). *Prospects for the development and application of spiking neural networks.* Paper presented at the 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus).

Liwei, W., Yan, Z., & Jufu, F. (2005). On the Euclidean distance of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 27*(8), 1334-1339. doi:10.1109/TPAMI.2005.165

Macukow, B. (2016). Neural Networks – State of Art, Brief History, Basic Models and Architecture. In K. Saeed & W. Homenda (Eds.), *Computer Information Systems and Industrial Management: 15th IFIP TC8 International Conference, CISIM 2016, Vilnius, Lithuania, September 14-16, 2016, Proceedings* (pp. 3-14). Cham: Springer International Publishing.

Mahalanobis, A., Shilling, R., Muise, R., Hines, K., & Neifeld, M. (2016, 11-15 July 2016). *Pixel resolution improvement using a sliding mask.* Paper presented at the 2016 15th Workshop on Information Optics (WIO).

Maltoni, D. (2011). Pattern Recognition by Hierarchical Temporal Memory *DEIS Technical Report*.

Manohar, N., Kumar, Y. H. S., & Kumar, G. H. (2016, 21-24 Sept. 2016). *Supervised and unsupervised learning in animal classification.* Paper presented at the 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI).

Marr, D., & Nishihara, H. K. (1978). Representation and Recognition of the Spatial Organization of Three-Dimensional Shapes. *Proceedings of the Royal Society of London. Series B. Biological Sciences, 200*(1140), 269-294. doi:10.1098/rspb.1978.0020

Mata, G., Morales, M., Romero, A., & Rubio, J. (2015). Zigzag persistent homology for processing neuronal images. *Pattern Recognition Letters, 62*, 55-60. doi:http://dx.doi.org/10.1016/j.patrec.2015.05.010

Maxwell, J. B., Pasquier, P., & Eigenfeldt, A. (2009). Hierarchical Sequential Memory for Music: A Cognitive Model. *International Society for Music Information Retrieval Conference*, 429-434.

Mayer, R. E. (1998). Does the Brain Have a Place in Educational Psychology? *Educational Psychology Review, 10*(4), 389-396. doi:10.1023/a:1022837300988

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics, 5*(4), 115-133. doi:10.1007/bf02478259

McCulloch, W. S., & Pitts, W. (1990). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology, 52*(1), 99-115. doi:10.1007/bf02459570

Mehta, M. R., Lee, A. K., & Wilson, M. A. (2002). Role of experience and oscillations in transforming a rate code into a temporal code. *Nature, 417*, 741. doi:10.1038/nature00807
https://www.nature.com/articles/nature00807#supplementary-information

Melis, W. J. C., Chizuwa, S., & Kameyama, M. (2009). *Evaluation of the Hierarchical Temporal Memory for a Real World Application.* Paper presented at the 2009. 4th International Symposium on Innovative Computing, Information and Control.

Nezhinsky, A. E., & Verbeek, F. J. (2012). Efficient and Robust Shape Retrieval from Deformable Templates. In T. Margaria & B. Steffen (Eds.), *Leveraging Applications of Formal Methods, Verification and Validation. Applications and Case Studies: 5th International Symposium, ISoLA 2012, Heraklion, Crete, Greece, October 15-18, 2012, Proceedings, Part II* (pp. 42-55). Berlin, Heidelberg: Springer Berlin Heidelberg.

Nishino, K., & Nayar, S. K. (2006). Corneal Imaging System: Environment from Eyes. *International Journal of Computer Vision, 70*(1), 23-40. doi:10.1007/s11263-006-6274-9

Nixon, M. S., & Aguado, A. S. (2012a). Chapter 4 - Low-level feature extraction (including edge detection). In *Feature Extraction & Image Processing for Computer Vision (Third edition)* (pp. 137-216). Oxford: Academic Press.

Nixon, M. S., & Aguado, A. S. (2012b). Chapter 5 - High-level feature extraction: fixed shape matching. In *Feature Extraction & Image Processing for Computer Vision (Third edition)* (pp. 217-291). Oxford: Academic Press.

Nixon, M. S., & Aguado, A. S. (2012c). Chapter 7 - Object description. In *Feature Extraction & Image Processing for Computer Vision (Third edition)* (pp. 343-397). Oxford: Academic Press.

Pal, K. K., & Sudeep, K. S. (2016, 20-21 May 2016). *Preprocessing for image classification by convolutional neural networks.* Paper presented at the 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT).

Patil, S. S., & Patil, H. S. (2013). Study and Review of Various Image Texture Classification Methods. *International Journal of Computer Applications, 75*(16). doi:http://dx.doi.org/10.5120/13197-0897

Pau-Choo, C., Chen, E. L., & Ching-Tsorng, T. (1994, 27 Jun-2 Jul 1994). *Pattern recognition using a hierarchical neural network.* Paper presented at the Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference on.

Pompe, U. (2013). The Value of Computer Science for Brain Research. In H. Andersen, D. Dieks, W. J. Gonzalez, T. Uebel, & G. Wheeler (Eds.), *New Challenges to Philosophy of Science* (pp. 87-97). Dordrecht: Springer Netherlands.

Ponce, J., Berg, T. L., Everingham, M., Forsyth, D. A., Hebert, M., Lazebnik, S., . . . Zisserman, A. (2006). Dataset Issues in Object Recognition. *LNCS 4170*, pp. 29-48.

Ramli, I., & Ortega-Sanchez, C. (2015, 5-7 Oct. 2015). *Pattern recognition using hierarchical concatenation.* Paper presented at the Computer, Control, Informatics and its Applications (IC3INA), 2015 International Conference on.

Ramli, I., & Ortega-Sanchez, C. (2016, 22-25 Nov. 2016). *Pattern recognition using fragmentation and concatenation.* Paper presented at the 2016 IEEE Region 10 Conference (TENCON).

Ray, A. K. (2009). Fuzzy Similarity Measure, Measure of Fuzziness and Entropy. In *Fuzzy Image Processing and Applications with MATLAB* (pp. 31-44): CRC Press.

Ronacher, B. (1998). How do bees learn and recognize visual patterns? *Biological Cybernetics, 79*(6), 477-485. doi:10.1007/s004220050497

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386-408. doi:10.1037/h0042519

Rosenblatt, F. (1960). Perceptron Simulation Experiments. *Proceedings of the IRE, 48*(3), 301-309. doi:10.1109/JRPROC.1960.287598

Rubin, J., Parvaneh, S., Rahman, A., Conroy, B., & Babaeizadeh, S. (2017). Densely connected convolutional networks and signal quality analysis to detect atrial fibrillation using short single-lead ECG recordings. In (Vol. 44, pp. 1-4).

Sapkal, S. D., Kakarwal, S. N., & Revankar, P. S. (2007, 13-15 Dec. 2007). *Analysis of Classification by Supervised and Unsupervised Learning.* Paper presented at the International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007).

Savchenko, A. V. (2016). Intelligent Classification Systems. In *Search Techniques in Intelligent Classification Systems* (pp. 1-13). Cham: Springer International Publishing.

Schey, N. C. (2008). Song Identification Using the Numenta Platform for Intelligent Computing.

Sebestyen, G. (1961). Recognition of membership in classes. *IRE Transactions on Information Theory, 7*(1), 44-50. doi:10.1109/TIT.1961.1057617

Shavlik, J. W., Mooney, R. J., & Towell, G. G. (1991). Symbolic and neural learning algorithms: An experimental comparison. *Machine Learning, 6*(2), 111-143. doi:10.1007/bf00114160

Sherwin, J., & Mavris, D. (2009, 7-14 March 2009). *Hierarchical Temporal Memory algorithms for understanding asymmetric warfare.* Paper presented at the Aerospace conference, 2009 IEEE.

Slot, K., & Gozdzik, M. (2008, 19-21 Aug. 2008). *Fingerprint Alignment with Deformable Templates.* Paper presented at the 2008 19th International Conference on Systems Engineering.

Song, J., Chen, B., Chi, Z., Qiu, X., & Wang, W. (2007). Face Recognition Based on Binary Template Matching. In D.-S. Huang, L. Heutte, & M. Loog (Eds.), *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues: Third International Conference on Intelligent Computing, ICIC 2007 Qingdao, China, August 21-24, 2007 Proceedings* (pp. 1131-1139). Berlin, Heidelberg: Springer Berlin Heidelberg.

Sreela, S. R., & Idicula, S. M. (2017, 14-16 July 2017). *Modified densely connected convolutional network for content generation in automatic image description generation system.* Paper presented at the 2017 IEEE Region 10 Symposium (TENSYMP).

Stanislaw, H., & Todorov, N. (1999). Calculation of signal detection theory measures. *Behavior Research Methods, Instruments, & Computers, 31*(1), 137-149. doi:10.3758/bf03207704

Starzyk, J. A., & Haibo, H. (2007). Anticipation-Based Temporal Sequences Learning in Hierarchical Structure. *Neural Networks, IEEE Transactions on, 18*(2), 344-358. doi:10.1109/TNN.2006.884681

Starzyk, J. A., & Haibo, H. (2009). Spatio-Temporal Memories for Machine Learning: A Long-Term Memory Organization. *Neural Networks, IEEE Transactions on, 20*(5), 768-780. doi:10.1109/TNN.2009.2012854

Stearns, S. D. (1960). A Method for the Design of Pattern Recognition Logic. *IRE Transactions on Electronic Computers, EC-9*(1), 48-53. doi:10.1109/TEC.1960.5221604

Stefanucci, J. K. (2011). Object and Scene Recognition. In *Visual Perception from a Computer Graphics Perspective* (pp. 369-391): A K Peters/CRC Press.

Steinbach, M., & Tan, P.-N. (2009). kNN. In *The Top Ten Algorithms in Data Mining* (pp. 151-161): Chapman and Hall/CRC.

Sze, V., Chen, Y., Yang, T., & Emer, J. S. (2017). Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *Proceedings of the IEEE, 105*(12), 2295-2329. doi:10.1109/JPROC.2017.2761740

Takahashi, Y., Tanaka, H., Suzuki, A., Shio, A., & Ohtsuka, S. (2007). License plate recognition using gray image template matching with noise reduction filters and character alignment. *Systems and Computers in Japan, 38*(3), 49-61. doi:10.1002/scj.20342

Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T. e. e., & Maida, A. S. (2018). Deep Learning in Spiking Neural Networks. *CoRR, abs/1804.08150*.

Taylor, W. K. (1959). Pattern-recognition system. *Electrical Engineers, Journal of the Institution of, 5*(51), 157-160. doi:10.1049/jiee-3.1959.0065

Tirunelveli, G., Gordon, R., & Pistorius, S. (2002, 2002). *Comparison of square-pixel and hexagonal-pixel resolution in image processing.* Paper presented at the IEEE CCECE2002. Canadian Conference on Electrical and Computer Engineering. Conference Proceedings (Cat. No.02CH37373).

Tou, J. T. (1968). Feature extraction in pattern recognition. *Pattern Recognition, 1*(1), 3-11. doi:http://dx.doi.org/10.1016/0031-3203(68)90011-3

Turing, A. M. (1950). I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind, LIX*(236), 433-460. doi:10.1093/mind/LIX.236.433

Turk, M. A., & Pentland, A. P. (1991, 3-6 Jun 1991). *Face recognition using eigenfaces.* Paper presented at the Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

Tversky, A. (1977). Features of Similarity. *Psychological Review*, 327-352.

Tversky, A., & Gati, I. (1978). Studies of Similarity. *Cognition and Categorization, E. Rosch and B. Lloyd, Eds. Hillsdale, N.J.: Earlbaum*.

Ukil, A. (2007). Neural Network. In *Intelligent Systems and Signal Processing in Power Engineering* (pp. 75-159). Berlin, Heidelberg: Springer Berlin Heidelberg.

Unger, S. H. (1959). Pattern Detection and Recognition. *Proceedings of the IRE, 47*(10), 1737-1752. doi:10.1109/JRPROC.1959.287109

VanRullen, R., Guyonneau, R., & Thorpe, S. J. (2005). Spike times make sense. *Trends in Neurosciences, 28*(1), 1-4. doi:https://doi.org/10.1016/j.tins.2004.10.010

Vari, A., & Vecsenyi, J. (1988). Concepts and tools of artificial intelligence for human decision making. *Acta Psychologica, 68*(1–3), 217-236. doi:https://doi.org/10.1016/0001-6918(88)90057-1

Vidal, R., Ma, Y., & Sastry, S. S. (2016). Principal Component Analysis. In *Generalized Principal Component Analysis* (pp. 25-62). New York, NY: Springer New York.

Vu-Anh, N., Starzyk, J. A., & Wooi-Boon, G. (2012, 10-15 June 2012). *Sequence recognition with spatio-temporal long-term memory organization.* Paper presented at the Neural Networks (IJCNN), The 2012 International Joint Conference on.

Wang, H., Maldonado, D., & Silwal, S. (2011). A nonparametric-test-based structural similarity measure for digital images. *Computational Statistics & Data Analysis, 55*(11), 2925-2936. doi:http://dx.doi.org/10.1016/j.csda.2011.04.021

Wen, R., Fu, K., Sun, H., Sun, X., & Wang, L. (2018). Image Superresolution Using Densely Connected Residual Networks. *IEEE Signal Processing Letters, 25*(10), 1565-1569. doi:10.1109/LSP.2018.2861989

Wettschereck, D., & Dietterich, T. G. (1995). An Experimental Comparison of the Nearest-Neighbor and Nearest-Hyperrectangle Algorithms. *Machine Learning, 19*(1), 5-27. doi:10.1023/a:1022603022740

Williams, M. R. (1991). The first computer. In H. Maurer (Ed.), *New Results and New Trends in Computer Science: Graz, Austria, June 20–21, 1991 Proceedings* (pp. 371-387). Berlin, Heidelberg: Springer Berlin Heidelberg.

Wong, A. M. F. (2014). Eye Movements; Saccades. In *Encyclopedia of the Neurological Sciences (Second Edition)* (pp. 249-251). Oxford: Academic Press.

Wosiak, A., Zamecznik, A., & Niewiadomska-Jarosik, K. (2016, 11-14 Sept. 2016). *Supervised and unsupervised machine learning for improved identification of intrauterine growth restriction types.* Paper presented at the 2016 Federated Conference on Computer Science and Information Systems (FedCSIS).

Wu, Y., Zhang, Y., Wei, L., & Ozcan, A. (2016, 5-10 June 2016). *Multiplexed color imaging using demosaiced pixel super-resolution.* Paper presented at the 2016 Conference on Lasers and Electro-Optics (CLEO).

Xiaohang, M., & Dianhui, W. (2004, 2004). *Learning similarity for image retrieval with locally spatial information feedback.* Paper presented at the International Conference on Intelligent Sensing and Information Processing, 2004. Proceedings of.

Xie, X., Qu, H., Liu, G., Zhang, M., & Kurths, J. (2016). An Efficient Supervised Training Algorithm for Multilayer Spiking Neural Networks. *PLoS One, 11*(4). doi:http://dx.doi.org/10.1371/journal.pone.0150329

Xu, L., Krzyzak, A., & Suen, C. Y. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics, 22*(3), 418-435. doi:10.1109/21.155943

Xue, H. (2009). Svm. In *The Top Ten Algorithms in Data Mining* (pp. 37-59): Chapman and Hall/CRC.

Yadav, N., Yadav, A., & Kumar, M. (2015). History of Neural Networks. In *An Introduction to Neural Network Methods for Differential Equations* (pp. 13-15). Dordrecht: Springer Netherlands.

Yagawa, G., & Okuda, H. (1996). Neural networks in computational mechanics. *Archives of Computational Methods in Engineering, 3*(4), 435. doi:10.1007/bf02818935

Yeap, T. H., Zaky, S. G., Tsotsos, J. K., & Kwan, H. C. (1990, 1-3 May 1990). *A hierarchical neural network for temporal pattern recognition.* Paper presented at the Circuits and Systems, 1990., IEEE International Symposium on.

Yegnanarayana, B. (1994). Artificial neural networks for pattern recognition. *Sadhana, 19*(2), 189-238. doi:10.1007/bf02811896

Younes, L. (2010). Deformable Objects and Matching Functionals. In *Shapes and Diffeomorphisms* (pp. 203-247). Berlin, Heidelberg: Springer Berlin Heidelberg.

Zagoruyko, S., & Komodakis, N. (2016). Wide Residual Networks. *CoRR, abs/1605.07146*.

Zhang, K., Sun, M., Han, T. X., Yuan, X., Guo, L., & Liu, T. (2018). Residual Networks of Residual Networks: Multilevel Residual Networks. *IEEE Transactions on Circuits and Systems for Video Technology, 28*(6), 1303-1314. doi:10.1109/TCSVT.2017.2654543

Zhang, Y., Cao, W., & Wang, L. (2015, 3-6 Nov. 2015). *Implementation of high performance hardware architecture of face recognition algorithm based on local binary pattern on FPGA.* Paper presented at the 2015 IEEE 11th International Conference on ASIC (ASICON).

Zhao, F., & Ma, Z. M. (2006, 8-11 Oct. 2006). *Similarity Measures of Vague Sets Based on the Set-theoretic Approach.* Paper presented at the 2006 IEEE International Conference on Systems, Man and Cybernetics.

Zhao, M., Gersch, T. M., Schnitzer, B. S., Dosher, B. A., & Kowler, E. (2012). Eye movements and attention: The role of pre-saccadic shifts of attention in perception, memory and the control of saccades. *Vision Research, 74*, 40-60. doi:http://dx.doi.org/10.1016/j.visres.2012.06.017

Zheng, W.-S., Lai, J. H., & Li, S. Z. (2008). 1D-LDA vs. 2D-LDA: When is vector-based linear discriminant analysis better than matrix-based? *Pattern Recognition, 41*(7), 2156-2172. doi:http://dx.doi.org/10.1016/j.patcog.2007.11.025

Zhu, L., Chen, Y., & Yuille, A. (2010). Learning a Hierarchical Deformable Template for Rapid Deformable Object Parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 32*(6), 1029-1043. doi:10.1109/TPAMI.2009.65

"Every reasonable effort has been made to acknowledge the owners of copyright material. I would be pleased to hear from any copyright owner who has been omitted or incorrectly acknowledged.

Appendix A

Statement for the availability of research docs and codes, contact A/Prof. Cesar Ortega-Sanchez at C.Ortega@exchange.curtin.edu.au to access them. The documents are located in a folder named ‘16306215.’ To use the files in this folder, read the ReadMe.txt file which describe the contains of the folder and the instruction of how to use the code.