RECOGNITION OF HANDWRITTEN ARABIC (INDIAN) DIGITS USING ABDUCTIVE NETWORKS

BY

Isah Abdullahi Lawal

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER ENGINEERING

JANUARY 2010

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Isah Abdullahi Lawal** under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfilment of the requirements for the degree of MASTER OF SCIENCE IN COMPUTER ENGINEERING.

Thesis Committee

R E AbdelAal

Dr. Radwan Abdel-Aal (Advisor)

Mahmoud

Dr. Sabri A. Mahmoud (Member)

M. Elsh

Dr. Moustafa Elshafei (Member)

Dr. Basem Al-Madani

Department Chairman

Dr. Salam A. Zummo

Dean of Graduate Studies

6/2/10

Date

DEDICATION

DEDICATED

TO MY BELOVED FAMILY AND TO ALL MY TEACHERS

ACKNOWLEDGEMENT

All praise and glory are due to Allah, to Him belongs the sovereignty of the heavens and the earth. May His blessings and mercies be upon the noblest of mankind, Muhammad (S.A.W.), his household, his companions and the generality of the true believers to the last day. I am grateful to Allah for all His favours on me since my birth, these blessings are indeed innumerable, and the greatest of His bounties on me is being a Muslim.

I wish to express my gratitude to my parents, siblings, uncles and aunties for their support and kind words. I am really indebted to the entire Lawals, I appreciate you all.

My warmest appreciation goes to my thesis advisor, Professor Radwan Abdel-Aal for his intellectual guidance, support and readiness to assist during this research and other several occasions. Also, I appreciate the helpful suggestions and contributions of other thesis committee members: Professor Sabri A. Mahmoud and Professor Moustafa Elshafei.

In the same vein, I appreciate all my friends at KFUPM and in the Kingdom at large. You have all made my stay in this Kingdom a memorable one. Lastly, my special appreciation goes to all my friends in Nigeria, I cannot but mention specifically Engr. Abubakar Hussaini of NLGN and Engr Muhd Kabir Hassan for their brotherly advice and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THESIS ABSTRACT

NAME:	Isah Abdullahi Lawal
TITLE OF STUDY:	Recognition of Handwritten Arabic (Indian) Digits Using Abductive Network
MAJOR FIELD:	Computer Engineering
DATE OF DEGREE:	January, 2010

Accurate automatic recognition of handwritten Arabic digits has several important applications, e.g. in banking transactions, automation of postal services, and other data entry related applications. A number of modelling and machine learning techniques have been used for handwritten Arabic numerals recognition, including Neural Network, Support Vector Machine, and Hidden Markov Models. This thesis proposes the application of abductive networks to the problem. We studied the performance of various abductive network architectures on a dataset of 21120 samples of handwritten 0-9 digits produced by 44 writers. We developed a new feature set using histogram of contour points chain codes. Recognition rates as high as 99.22% were achieved, which surpass the performance reported in the literature for other recognition techniques on the same data set. Moreover, the technique achieves a significant reduction in the number of features required.

خلاصة الرسالة

إن للتعرف الآلي على الكتابة اليدوية استخدامات شتى، منها ما يتعلق بالعمليات المصرفية ومنها ما يتعلق بالخدمات البريدية ومنها ما هو من قبيل إدخال البيانات. وقد استخدمت تقنيات نمذجة وتعلم آلي عديدة للتعرف على الأعداد العربية، بما في ذلك الخلايا العصبية، وآلات الدعم الإشعاعي، ونماذج ماركوف المخبئة.

تقترح هذه الأطروحة استخدام معمارية الشبكات المسماة بالـ(abductive)، وتعمل على 21120 عينة من الأعداد (0-9) تمت كتابتها بواسطة 44 كاتباً. وقد طورنا مجموعة جديدة من الملامح المعتمدة على المدرج الإحصائي لنقاط الكنتور من سلاسل الترميز. وبلغت نسب التعرف الناجح 99.22%، والذي يتفوق على نسب النجاح الأخرى المرصودة في الأدبيات المستخدمة للمدخلات نفسها. علاوة على ذلك، فإن التقنية قد قلصت عدد الملامح المطلوبة بشكل ملفت.

CHAPTER 1

INTRODUCTION

1.1 Overview

Recent advances in the field of pattern recognition and analysis have encouraged researchers to develop techniques for the automatic recognition of language characters. However, when the input stream of characters is handwriting, then a special advanced recognition system is required as an interface with the computer machine to be able to read and interpret the written text [1]. Handwritten digit recognition systems have contributed considerably to the progress of the automation process and have improved the interaction between human and machine in many applications, including cheque verification and a large variety of banking transactions, business and data entry applications.

Over the past few decades, many approaches have been proposed for pre processing, feature extraction, classification, and some standard image databases are widely used to evaluate the performance of such system [1]. However, most of these research activities have been done on Latin, Japanese and Chinese text and little work has been done on Arabic text including Arabic (Indian) numerals. This may be partially attributed to the fact that no generally accepted databases existed for Arabic text/numeral recognition that were freely available to researchers [2]. Therefore, various research groups working in the field had to develop their own datasets, which is tedious work. Moreover, performance of various techniques on different datasets may not be directly comparable.

Like Latin, Arabic handwriting, has many handwriting styles, so the recognition of handwritten text (characters and digits) is a difficult task due to the different handwriting styles and the associated inter-, and intra-writer variation [2]. Moreover, the Arabic language is characterized by the cursive nature of its writing rules and by having several shapes of the same character based on the position of the character in a word, which adds to the difficulties of recognition. Although Arabic text is cursive and is written from right to left, Arabic (Indian) digits are isolated and they are written such that the most significant digit is at the leftmost side as shown in Figure 1.1. Therefore, techniques used in Latin-based applications need to be augmented with advanced feature extraction and pattern recognition approaches to achieve high recognition rates for Arabic text/numerals.

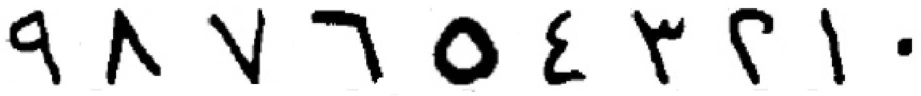


Figure 1.1: Handwritten Arabic (Indian) digits from 0 to 9

1.2 Handwritten digit recognition systems

A typical handwritten recognition system for Arabic (Indian) digits is pictorially presented in Figure 1.2. At the input end, digits typed or written in a document are scanned and digitized by an optical scanner to produce a digitized image. The system will locate the region in which digits have been entered, typed, or written in the input document and then segment the region for further processing.

The first stage of the processing is to remove any noise from the segmented region due to optical scanning. This is then followed by normalizing the image size to facilitate feature extraction. The feature extraction is performed next, and this involves extracting the value of a set of pre-defined features of each digit to be used for recognition. The recognition stage employs classifiers that were previously develop through training on a set of solved examples using the same features. The trained classifier is used to recognize the handwritten Arabic (Indian) digits presented to the system. The handwritten digit recognition processes comprises of the following fundamental steps, i.e. pre-processing, feature extraction, classification and below is a brief description of the steps involved;

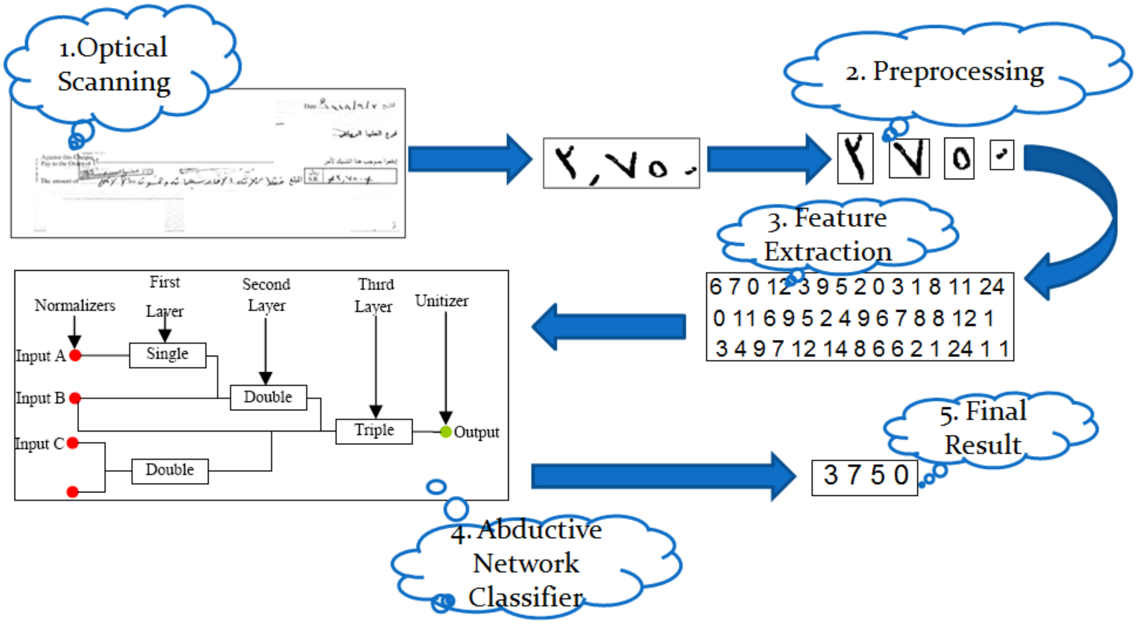


Figure 1.2: Handwritten Arabic (Indian) digits recognition system

1. PRE-PROCESSING

When a text is scanned and digitized, the raw data may carry a certain amount of noise, for example, a scanner with low resolution will produce touching line segments and smeared images. Unwanted noise in the scanned image may lead to ambiguous features and therefore poor recognition rates. The segmented region goes into a pre-processing phase to eliminate random noise, voids and other spurious components. More over, the dimensions of the handwritten digit, in terms of height and width, may differ from digit to digit. In order to make the system size-invariant, the images undergo normalisation in size and orientations; see Figure 1.3. Collectively, these operations combined facilitate the extraction of distinctive features in the next stage.

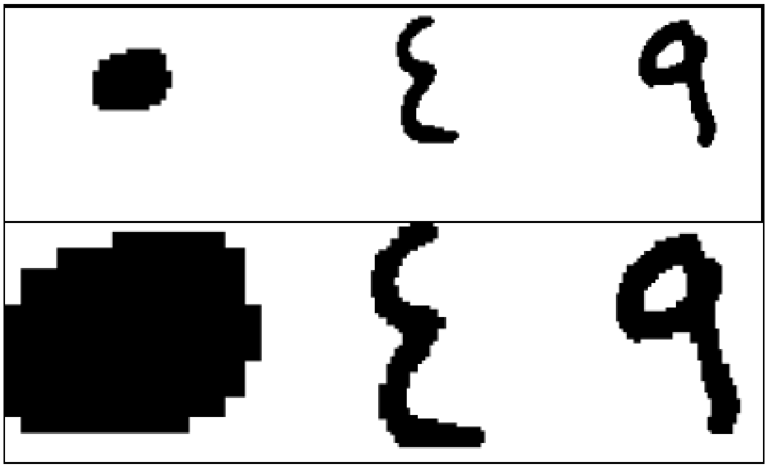


Figure 1.3: Digits ‘0’, ‘4’, and ‘9’ in original form and normalized forms, upper and lower row respectively

2. FEATURE EXTRACTION

Features are the information passed to the recognizer, such as pixels counts, shape information, or mathematical properties. If the features extracted are carefully chosen it is expected that the features set will extract the relevant information from the input data in order to perform the desired recognition task. In handwritten Arabic digit recognition system, the objective is to extract good quality features that allow optimum discrimination between the various digits. Most methods study the structural form of the image and extract a skeleton or list of contour projections from the image [3]. In some cases, quantitative features such as the number and density of black pixels and the numbers of endpoints, loops, corner points, and branch points are used [3]. More discussion on feature extraction techniques is presented in chapter 2. Once the distinctive features of the digits have been extracted, they are forwarded to a classifier network built during learning process to classify the digits.

3. CLASSIFICATION

Classification is an important stage in a handwritten Arabic (Indian) digits recognition system. In the past, many neural network architectures have been used as classifiers for digit classification. Multi-layered perceptron (MLP) is usually a common choice, see Figure 1.4. The MLP is initially trained with training set that comprises of all the features extracted from the handwritten digit and then the optimal model developed afterward is evaluated with evaluation set yet unseen by the model. The size of the extracted handwritten digit features determine the number of input neurons in the input layer of the MLP while the classes of the digits determine the number of output neuron in the output layer. Unfortunately, as the number of inputs and outputs grow, the MLP grows quickly and its training becomes computationally expensive. In addition, it is not easy to come up with a suitable network design because users of MLP must select training parameters, model architecture and all these are done in trial and error manner. Recently, support vector machines (SVM) and Hidden Markov Models (HMM) were also used for classification. In SVM, training vectors are mapped into a higher dimensional space by the function called the kernel function. Then SVM finds a linear separating hyper plane with the maximal margin in this higher dimensional space which corresponds to minimising the weight vector in a canonical framework. The solution is obtained as a set of support vectors that can be sparse. These lie on the boundary and as such summarise the information required to separate the data see Figure 1.5. Although SVMs have good generalization performance, the main drawback is how to the select the best kernel function parameters during training. Conventional HMM model one-dimensional sequences of data and contain states and probabilities for transitioning between them according to an observed sequence of data or “observations”, see Figure 1.6. For Arabic (Indian) digit recognition, the observations could be sets of pixel values (features) and states could represent parts of digits. Like ANN and SVM, one of the HMM limitations is long training time.

Whichever classifier is chosen for Arabic (Indian) digit classification, it must be developed through training on a dataset of examples that are representative of the digits to be classified. During the training process, the classifier learns the feature patterns that distinguish between the different digit classes. Learning techniques fall into two main categories: Supervised and Unsupervised. Unsupervised learning assumes no previous knowledge of the classes present in the training data. It is usually performed using clustering methods and the analysis of the classes is done *a posterior.* Supervised learning techniques assume that the classes in the data are known and can be described by a probabilistic distribution, with as many variables as dimensions of the space. Such distributions describe the chance of finding a feature belonging to any class and the discrimination between the classes is learned during training.

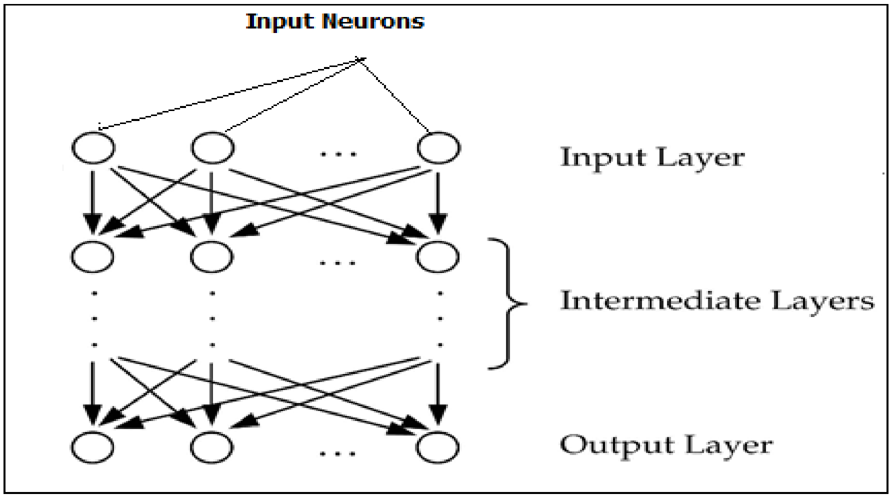


Figure 1.4: MLP neural network architecture

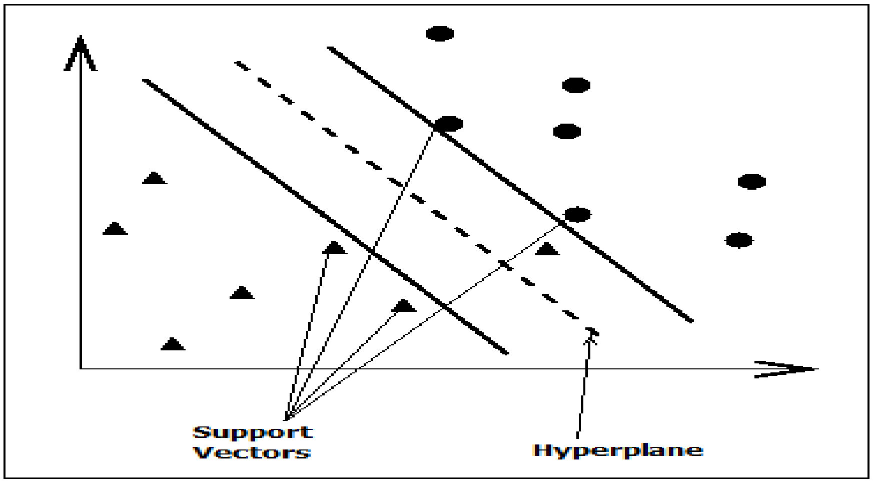


Figure 1.5: Creation of margins between two class labels in a training set by support vectors

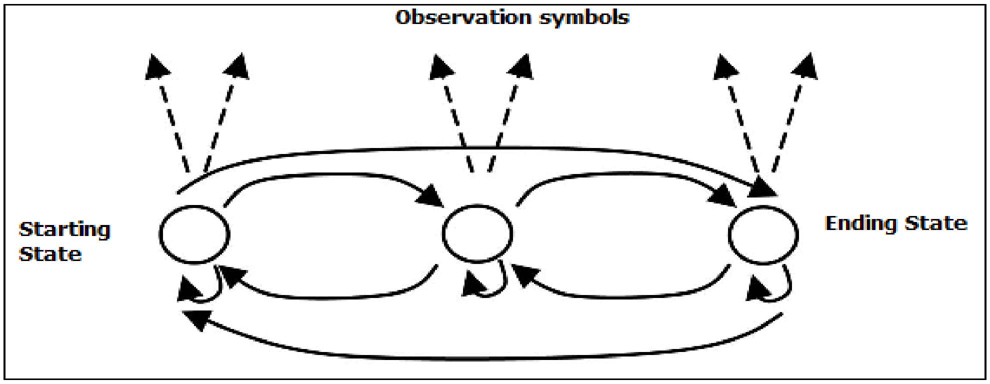


Figure 1.6: HMM with three states and two possible observation symbols at each state

1.3 Thesis Objectives

Many modern machine learning approaches have been used for handwritten Arabic (Indian) digits recognition in the recent years with high recognition rates. However, a very important factor in the process is overlooked; the speed and simplicity of the training and recognition. The objective of this research work is to explore the use of alternative technique of abductive networks based on the group method of data handling (GMDH) algorithm [4][5]. The GMDH approach to classification offers the advantage of simplified and more automated model synthesis. In addition to using the features developed earlier by writers in the field, we proposed new features based on histograms of object contour chain codes. Compared to earlier work on the same dataset, a significant improvement in recognition rate has been achieved together with a considerable reduction in the number of input features.

The work presented here aims at the following aspects.

- Review the existing literature on Arabic handwritten digit recognition systems and techniques

- Explore the new recognition approach which promises optimal performance both in terms of recognition accuracy and speed, and
- Compare performance of various existing techniques with that of the proposed approach

1.4 Thesis Organisation

The thesis is structured as follows:

- **Chapter 2:** Presents a literature review of the techniques used for Arabic text (character/digits) recognition.
- **Chapter 3:** Gives an in depth description of abductive networks, its structure and applications.
- **Chapter 4:** Presents description of the dataset used and feature extraction techniques employed in this work.
- **Chapter 5:** Discusses the experiments performed and analyzes the result obtained at various stages under different conditions in this research.
- **Chapter 6:** Gives conclusions and list suggestion for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

A good deal of effort has been made in the recent past by researchers in their attempt to come up with computational intelligence models with an acceptable level of recognition accuracy for Arabic text. An extensive survey and an overview of different approaches to the problem can be found in [2]. However, template matching, structural analysis, neural networks and statistical models based on Hidden Markov Models have been popular strategies for Arabic text recognition.

2.2 Template matching approach

One of the simplest and earliest approaches to handwritten digit recognition is based on template matching. Matching is a generic operation in pattern recognition which is used to determine the similarity between two entities (points, curves, or shapes) of the same type. In template matching, a template **(**typically a 2D shape) or a prototype of the pattern to be recognized is available. The pattern to be recognized is matched against the stored template while taking into account all allowable pose (translation and rotation) and scale changes. The similarity measure, often correlation, may be optimized based on the available training set. Often, the template itself is learned from the training set. While template matching is effective in handwritten digit recognition, it has a number of limitations. For example, it is computationally demanding and would fail if the patterns are distorted due to the imaging process, viewpoint change, or large intraclass variations among the patterns [3].

Al-Omari [6] used an average template-matching approach for recognizing Arabic (Indian) numerals. Feature vector representing significant boundary point distances from the digit centre of gravity (COG) were extracted and used to form a model for each numeric digit; see Figure 2.1 [6]. The author made the classification using the Euclidean distance between the feature vector of the test samples and the extracted template models. Similarly, Ziaratban et al. [7] proposed a template based feature extraction method. Twenty templates believed to be capable of capturing the most significant information from handwritten Farsi/Arabic numerals were selected heuristically. The features generated were used to train a neural network classifier for digit recognition.

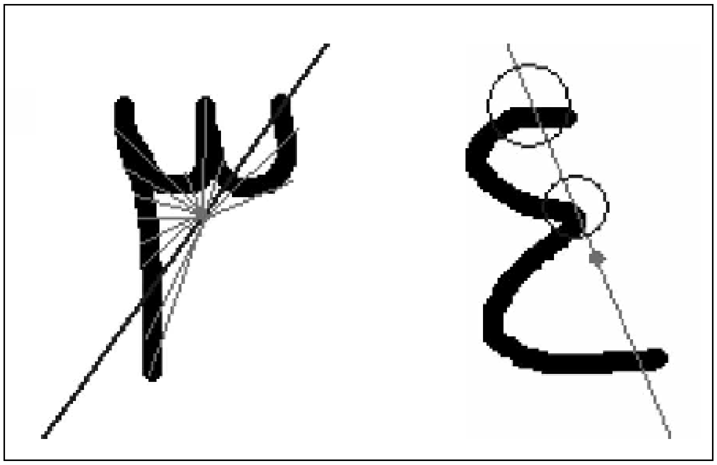
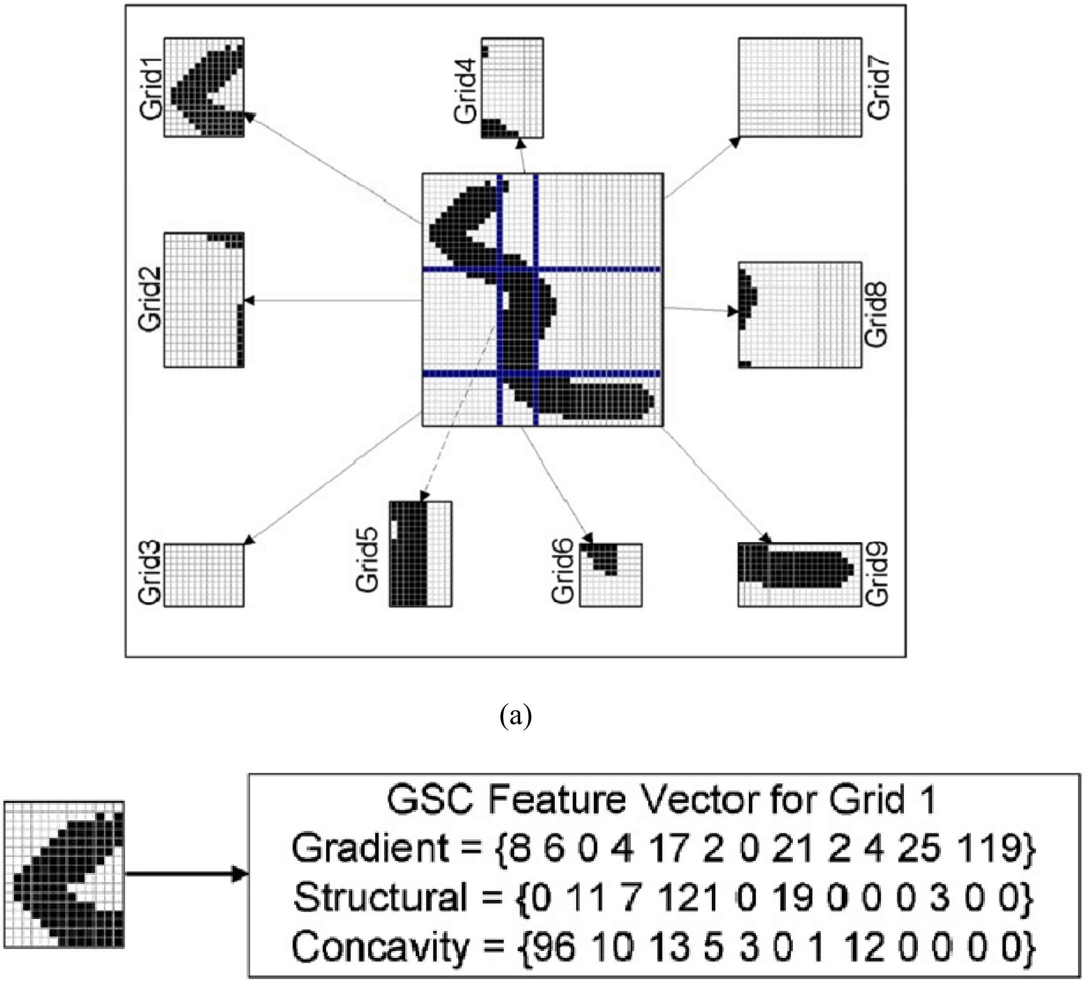


Figure 2.1: Example portraying more than one intersection along the measurement vector for digits ‘3’ and ‘4’

2.3 Structural analysis approach

The main primitives that form digits are line segments and curves. Different arrangements of these primitives form different digits. To recognize a digit, it is necessary to first determine the structural relationships between the features make up of the digit [8].

Structural features are intuitive aspects of writing, such as loops, branch-points, endpoints, and dots. They are often, but notalways computed from a skeleton of the text image. The text is thinned in order to extract information of stroke intersections [9]. Thinning text is fraught with difficulties, particularly with poor quality characters, as superfluous 'tails' are often produced and the end of a stroke can split into two. Although Arabic (Indian) digits are distinct from one another, many Arabic letters share common primary shapes, differing only in the number of dots and whether the dots are above or below the primary shape. Structural features are a natural method for capturing dot information explicitly, which is required to differentiate between such letters. This is partly the reason why structural features are more common for the recognition of Arabic text [10]. Al-Omari and Al-Jarrah [11] presented an online recognition system for handwritten Indian numerals 1 to 9. The system extracted geometrical features of the skeleton of the digits which were then fed into a probabilistic neural network (PNN) for classification. Awaidah et al. [12] used three different features of Arabic handwritten digit in their proposed recognition system. These features are known as the gradient, structural, and concavity (GSC) feature set.
The gradient features capture the low-level gradient direction frequency; structural features capture middle-level geometric characteristics of the digit while the concavity features capture high-level topological and geometrical features, as shown in Figure 2.2 [12]. Recognition is performed using a hidden markov model.



(a)

GSC Feature Vector for Grid 1
Gradient = {8 6 0 4 17 2 0 21 2 4 25 119}
Structural = {0 11 7 121 0 19 0 0 0 3 0 0}
Concavity = {96 10 13 5 3 0 1 12 0 0 0 0}

(b)

Figure 2.2: Sample extracted segment for Arabic digit 4 and feature vector for Grid 1

2.4 Statistical analysis approach

Statistical features are numerical measures computed over images or regions of images. They include, but are not limited to, pixel densities, histograms of chain code directions, moments, and Fourier descriptors. Mahmoud [13] proposed using Fourier descriptors and character contour encoding techniques for Arabic character recognition. The contour analysis approach resulted in a recognition rate of 98%. Shahrezea et al. [14] proposed a shadow coding method for the recognition of Persian handwritten digits. In this method, a digit is encoded by defining a segment mask on the digit image and the features are calculated by projecting the image pixels into the mask. Similarly, a method for recognition of isolated handwritten Arabic (Indian) numerals using Hidden Markov Models (HMM) was presented by Mahmoud [15]. In his method, four sets of features (i.e. angle, circle, horizontal and vertical) were generated based on the segmentation of digit pixel image, and for each segment the ratio of black pixels to segment size was computed. These features were used for training and evaluating the HMM models. A recognition rate of 97.99% was achieved. Sadri et al. [16] viewed each sample digit from four different directions and estimated from the derivative of the horizontal and vertical profiles of the image a feature vector of length 16, as shown in Figure 2.3 [16]. The resulting 64 features obtained were used to train SVM classifiers to separate the different digits. A recognition rate 94.14%, was achieved using the CENPARMI Indian (Arabic/Persian) handwritten digit database. Al-Yousefi and Udpa [17] described a statistical approach for the recognition of Arabic handwritten characters, using a normalized zero-order moment horizontal and vertical projections of the primary character to extract the features of each character. Classification was accomplished using quadratic discriminant functions.

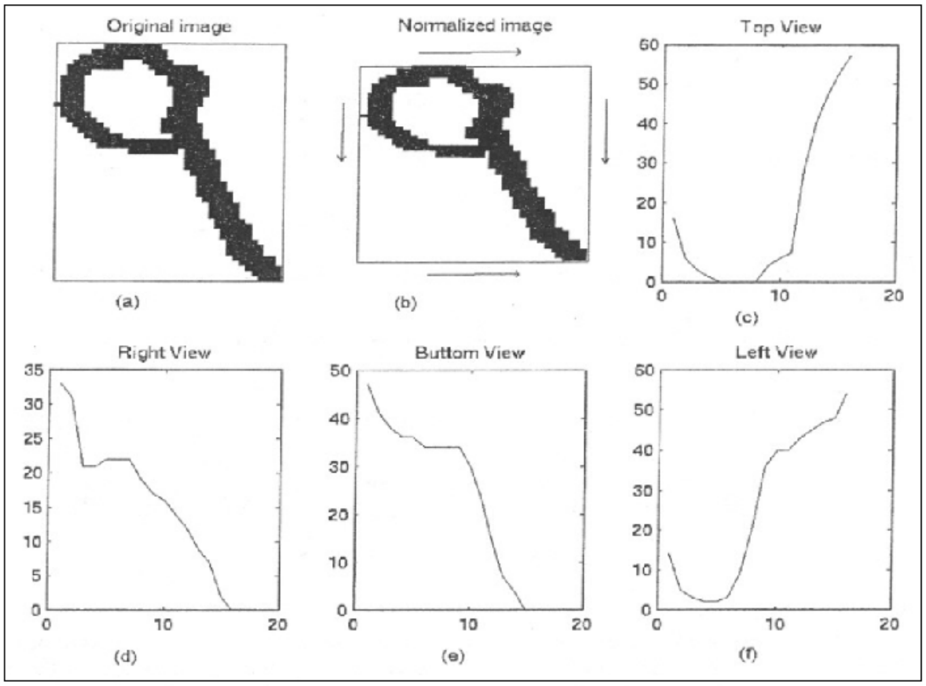


Figure 2.3: Sample of digit 9, normalized image and the resultant side views

2.5 Other approaches

Other approaches to Arabic text recognition have been reported by researchers. Soltanzadeh et al. [18] used the normalized image profile which is calculated at multiple orientations, as the main feature for the recognition of Persian/Arabic handwritten digits. The crossing counts and projection histogram calculated at multiple orientations were used as complementary features. The authors indicated that most of the system errors occurred in discriminating the digits ‘2’,‘3’, ‘4’, ‘0’,and ‘5’. Hence, discriminating these digits required the use of additional features and possibly the use of additional classifiers. Said et al. [19] used the pixels of the normalized digit image as features into a neural network classifier, where the number of the hidden units was determined dynamically. In a different approach, Altuwaijr et al. [20] proposed a hybrid Arabic character recognition system based on moment invariant features.The feature extraction stage uses a set of moment invariant descriptors which are invariant under shift, scaling, and rotation. The actual classification was performed using a multilayer perceptron network with a back-propagation learning algorithm.

Despite the wide use of the modelling tools described above, their black box nature and the complexity in determining best network topology and training parameters represent a major drawback in their application. This lack of clear picture for selecting important features and training parameters for the above mentioned models, most often compel researchers to resort to trial and error approach which can be tiresome and time consuming. Thus, the need to exploit another way of getting improved results while eliminating potential disadvantage of previous recognition approaches.

Recently, abductive networks have emerged as a powerful tool in pattern recognition, decision support, classification, and forecasting in many areas [21-22]. Inspired by promising results obtained in other fields, we explore the use of this approach for the recognition of handwritten Arabic (Indian) digits.

CHAPTER 3

GMDH AND AIM ABDUCTIVE NETWORKS

3.1 Overview

Abductory inductive mechanism (AIM) is a powerful supervised inductive learning tool for automatically synthesizing network models from a database of input and output values [23]. The model emerging from the AIM synthesis process is a robust and compact transformation implemented as a layered abductive network of feed-forward functional elements as shown in Figure 3.1. The potential for GMDH in Arabic text recognition has not been explored before in the literature. However, compared to neural networks and other learning tools, the method offers the advantages of faster model development requiring little user intervention, faster convergence during model synthesis without the problems of getting stuck in local minima, automatic selection of relevant input variables, and automatic configuration of the model structure.

3.2 Abductive machine learning

The abductive machine learning approach is based on the self organizing group method of data handling (GMDH) [24]. The GMDH approach is a proven concept for iterated polynomial regression that can generate polynomial models in effective predictors. The iterative process involves using initially simple regression relationships to derive more accurate representations in the next iteration in an evolutionary manner.

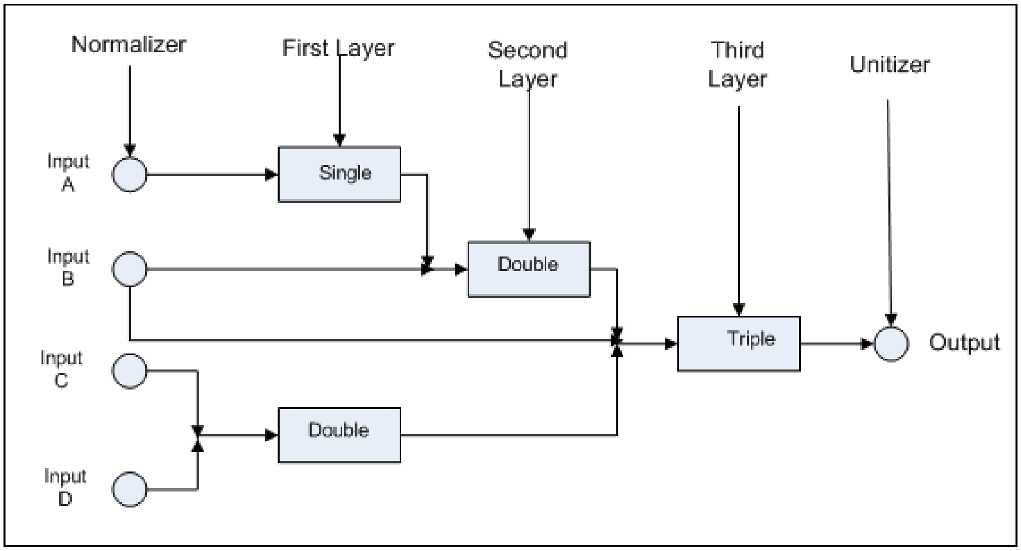


Figure 3.1: Abductive network showing various types of functional elements

The algorithm selects the polynomial relationships and the input combinations that minimize the prediction error in each phase. This prevents exponential growth in the polynomial model generated. Iteration is stopped automatically at a point in time that strikes a balance between model complexity for accurate fitting of the training data and model simplicity that enables it to generalise well with new data.

In the classical GMDH-based approach abductive network models are constructed by the following 6 steps [24]:

1. **Separating the original data into training data and testing data**.

   The available dataset are split into training dataset and testing dataset. The training dataset is used for estimating the optimum network model and the testing dataset is used for evaluating the network model obtained on the new data. Usually a 70-30 splitting rule is employed on the original data, but in this work a pre-determined split used by earlier published work was adopted to allow direct comparison of results.

2. **Generating the combinations of the input variables in each layer**.

Many combinations of r input variables are generated in each layer. The number of combinations is $p!/((p\text{-}r)!\ r!)$. Here, p is the number of input variables and the value of r is usually taking as 2.

3. **Calculating the partial descriptors**

For each input combination, a partial descriptor which describes the partial characteristics of the model is calculated by applying regression analysis on the training data. The following second order polynomial regression relationship is usually used

$$y_k = b_0 + b_1 x_i + b_2 x_j + b_3 x_i x_j + b_4 x_i^2 \qquad (1)$$

The output variables y_k in Eq. (l) are called intermediate variables.

4. **Selecting optimum descriptors**.

The classical GMDH algorithm employs an additional and independent selection data for selection purposes. To prevent exponential growth and limit model complexity, the algorithm selects only relationships having good predicting powers within each phase. The selection criterion is based on root mean squared (RMS) error over the selection data. The intermediate variables which give the smallest root mean squared errors among the generated intermediate variables (y_k) are selected.

5. **Iteration**

Steps 3 and 4 are iterated where optimum predictors from a model layer are used as inputs to the next layer. At each iteration, the root mean squared error obtained is compared with that of the previous value and the process is continued until the error starts to increase or a prescribed complexity is achieved. An increasing root mean squared error is an indication of the model becoming overly complex, thus over-fitting the training data and will more likely perform poorly in predicting the selection data.

6. **Stopping the multilayered iterative computation**

Iteration is stopped when the new generation regression equations start to have poorer prediction performance than those of the previous generation, at which point the model starts to become overspecialized and, therefore, unlikely to perform well with new data.

Computationally, the resulting GMDH model can be seen as a layered network of partial descriptor polynomials, each layer representing the results of iteration. Therefore, the algorithm has three main elements: representation, selection, and stopping. Figure 3.2 shows the flow chart of the classical GMDH-based training.

An abductive network model numerical input output relationships through abductive reasoning. As a result, the abductive network can be used effectively as a predictor for estimating the outputs of complex systems [25], as a classifier for handling difficult pattern recognition problems [26],[27] or as a system identifier for determining which inputs are important to the modelling system. With the model represented as a hierarchy of polynomial expressions, resulting analytical model relationships can provide insight into the modelled phenomena, highlight contributions of various inputs, and allow comparison with previously used empirical or statistical models.

AIM is a later development of the classical GMDH that uses a better stopping criterion that discourages model complexity without requiring a separate subset of selection data. AIM adopts a well-defined automatic stopping criterion that minimizes the predicted square error (PSE) and penalises model complexity to keep the model as simple as possible for best generalization. Thus, the most accurate model that does not overfit the training data is selected and hence a balance is reached between accuracy of the model in representing the training data and its generality which allows it to fit yet unseen new evaluation data.

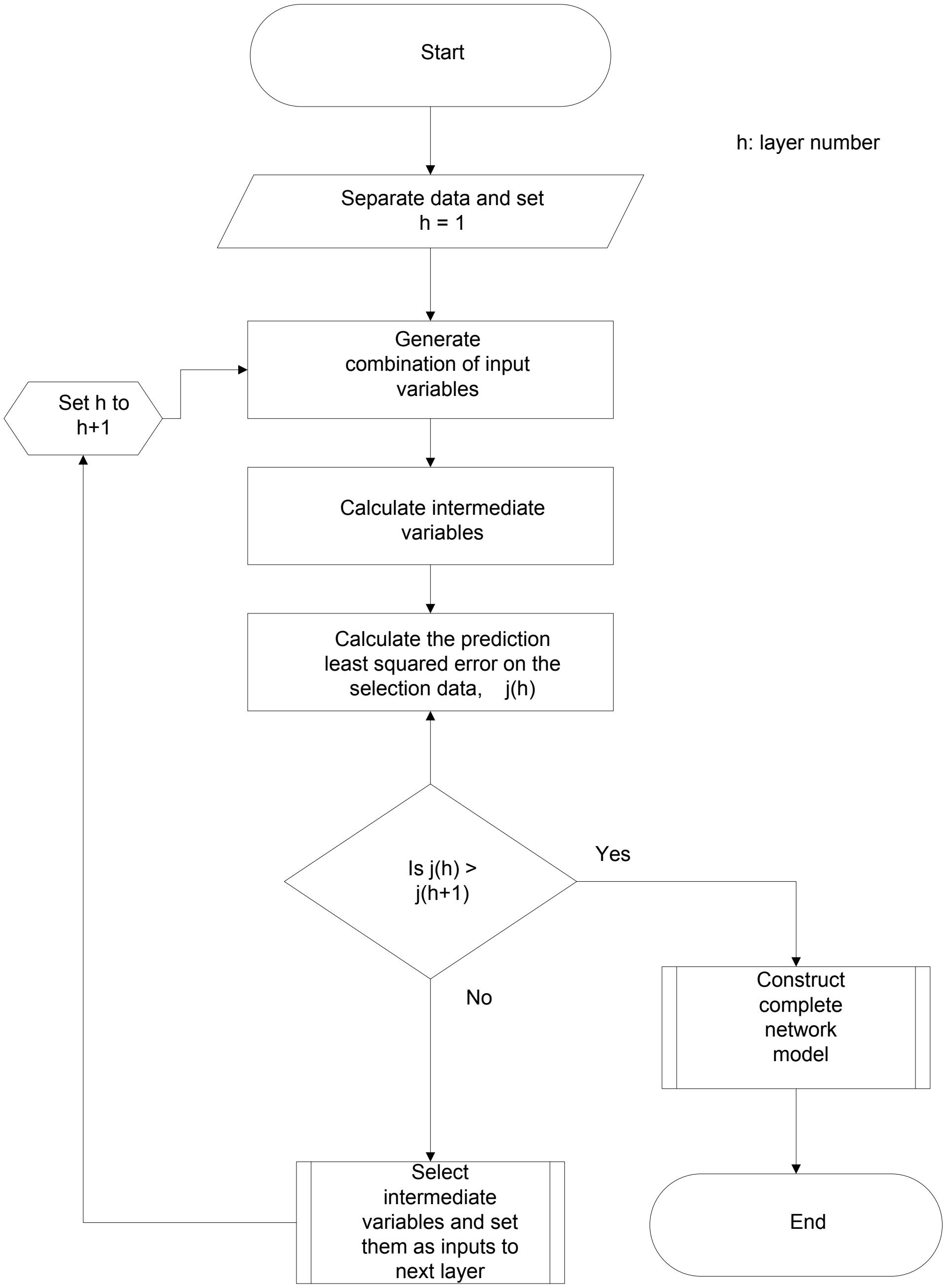


Figure 3.2: Flow chart of the classical GMDH-based abductive network

The PSE consists of two terms [23]:

$$\text{PSE} = \text{FSE} + \text{KP} \qquad (2)$$

Where FSE is the average fitting squared error of the network for fitting the training data and KP is the complexity penalty for the network, expressed as [23].

$$\text{KP} = \text{CPM}\ (2\sigma^2/\text{N})\ \text{K} \qquad (3)$$

Where CPM is the complexity penalty multiplier, K is the number of coefficients in the network, and σ^2 is a prior estimate of the model error variance. Usually, a complex network has a high fitting accuracy but may not generalize well on new evaluation data unseen previously during training. Training is automatically stopped to ensure a minimum value of the PSE for the CPM parameter used, which has a default value of 1. The user can also control the trade-off between accuracy and generality using the CPM parameter. CPM values greater than 1 will result in less complex models that are more likely to generalise well with unseen data while values less than the default value will result in a more complex models that are likely to overfit training data and produce poor prediction performance. Figure 3.3 shows the relationship between PSE, FSE and KP, [23].

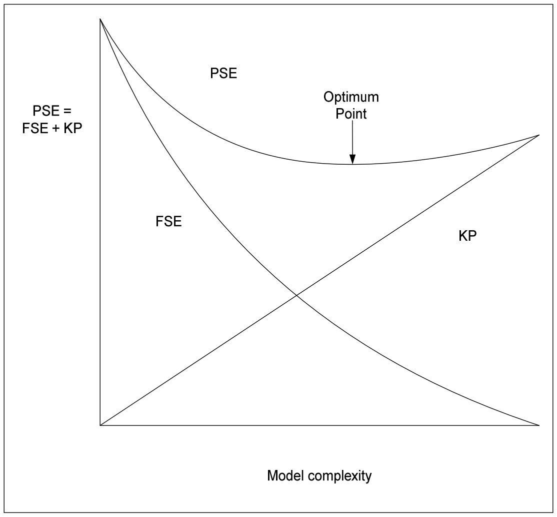


Figure 3.3: The predicted square error

3.3 AIM Functional Elements

The used version of AIM supports several functional elements [23], see Figure 3.1, including:

Normaliser*:* Transforms the original input into a normalized variable having a mean of zero and a variance of unity.

$$y = z_0 + z_1 x \qquad (4)$$

Where x is the original input, y is the normalized input, z_0 and z_1 are the coefficients of the normaliser.

Unitizer: Converts the range of the network outputs to a range with the mean and variance of the output values used to train the network.

Single Node*:* The single node only has one input and the polynomial equation is limited to the third degree, i.e.

$$y = z_0 + z_1 x + z_2 x^2 + z_3 x^3 \qquad (5)$$

Where x is the input to the node, y is the output of the node and z_0, z_1, z_2 and z_3 are the node coefficients.

Double Node*:* The double node takes two inputs and the third-degree polynomial equation includes cross term so as to consider the interaction between the two inputs, i.e.

$$y = z_0 + z_1 x_i + z_2 x_j + z_3 x_i^2 + z_4 x_j^2 + z_5 x_i x_j + z_6 x_i^3 + z_7 x_j^3 \qquad (6)$$

Where x_i, x_j are the inputs to the node, y is the output of the node and z_0, z_1, z_2 …and z_7 are the node coefficients

Triple Node: Similar to the single and double nodes, the triple node with three inputs has a more complicated polynomial equation allowing the interaction among these inputs.

It is noted that not all terms of an element's equation will necessarily appear in a node since AIM will throw away or carve terms that do not contribute significantly to the solution. The eligible inputs for each layer and the network synthesis strategy are defined as a set of rules and heuristics that form an integral part of the model synthesis algorithm as described earlier.

On a final note, any abductive network model is only as good as the training data used to construct it. To build a good model it is important that the training database be representative of the problem space. Figure 3.4 [23], illustrates a scenario where the training database used to create the AIM model does not cover an important portion of the problem. Training AIM using only the data to the left of the dotted line will result in a model that generalizes well within the training data range but will be inaccurate in the other region.

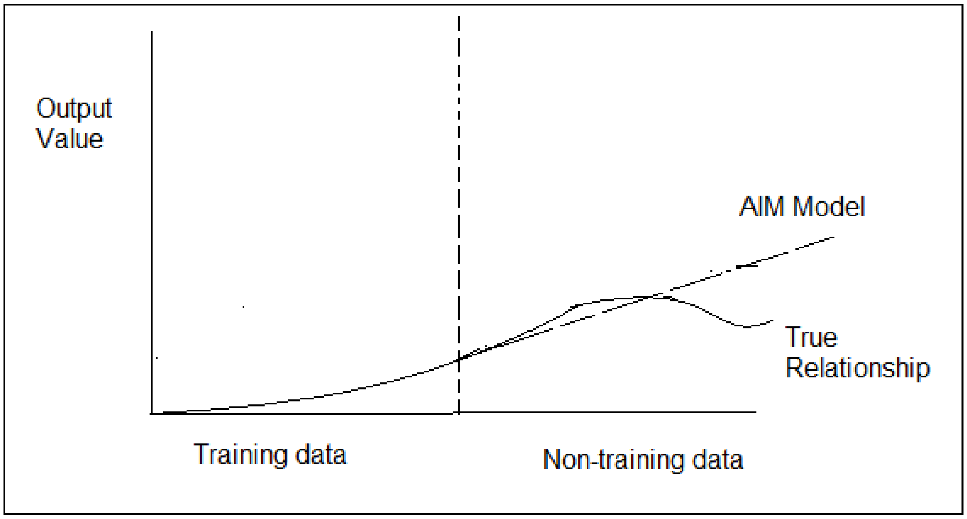


Figure 3.4: Non-representative training data

3.4 Abductive Network Committees

In this thesis work, the concept of abductive network models committees as an ensemble method for improving recognition rates in handwritten Arabic (Indian) digits recognition system was explored, see Figure 3.5

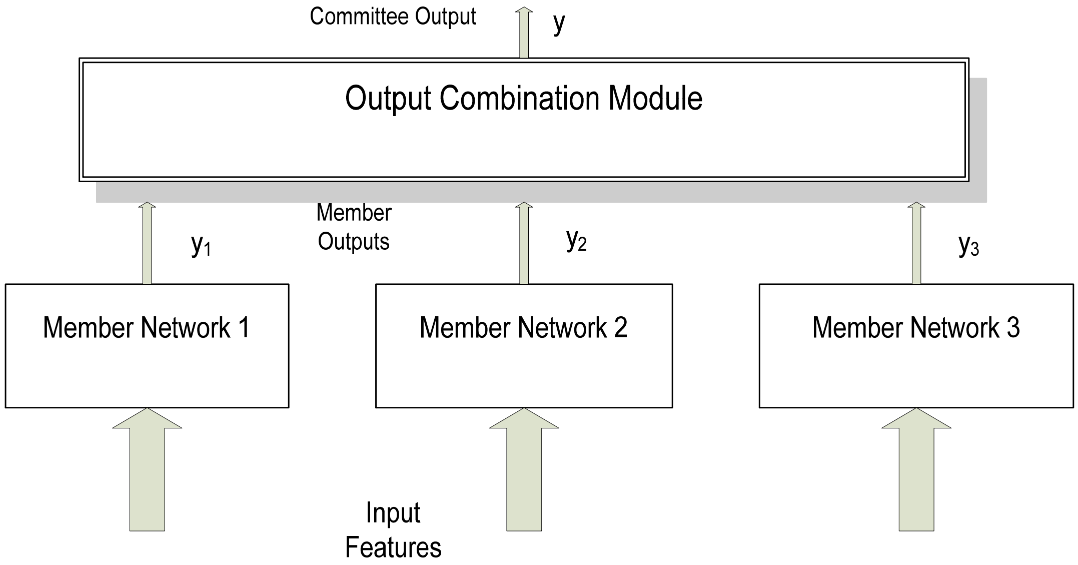


Figure 3.5: Schematic of the evaluation of a 3-member abdcutive network committee

3.4.1 Modelling individual committee members

As described in the previous sections, abductive networks implement a different learning approach from that used in other modelling tools like neural networks. With neural networks, the freedom to choose modelling parameters such as size and number of hidden layers, number of neurons and the type of transfer function for the various layers and other training parameters provides room for diversity among the resulting models (though all at the detriment of model simplicity). However, while the self-organizing and self-stopping features of abductive networks simplify model development and reduce user intervention, they are not very conducive in the way of increasing diversity and independence of models generated from a given training dataset. Only a few parameters can be controlled by the user when developing AIM abductive networks as stated earlier.

One option to introduce diversity in individual committee member is to use different feature subsets present in the training set to train the individual members. Training committee members on different feature subsets of approximately equal predictive power can enhance diversity and improve committee performance. The other option to create diversity among committee members is by splitting the full training set available into *n* mutually exclusive subsets to train an n-member committee each using the full set of the available features. This ensures model independence, but may reduce the quality of individual members, particularly when the total data available for training is limited.

3.4.2 Combining the outputs of individual committee members

There are many output combination methods reported in literature but the two methods used in this work are briefly explained below:

1. Simple majority vote of committee members outputs:

Here the final committee output from the outputs combination module is obtained directly using a simple majority vote among the outputs of individual members. In this case the number of members in the committee should preferably be an odd number, with a minimum value of 3.

2. Simple averaging of committee members outputs:

In this method, the final committee output is determined by first averaging linearly the output of the committee members using the relationship shown in equation 8 [28] and then applying the winner-take-all approach to determine the committee output.

$$y_o = \frac{1}{n}\sum_{j=1}^{n} y_j \qquad (8)$$

Where n is number of members, y_j is the member linear output and y_o is the final linear output

CHAPTER 4

THE DATASET AND FEATURE EXTRACTION

4.1 Introduction

In any handwritten text recognition system the formation of suitable features that are representative of the text is very important. This is so because to achieve acceptable recognition accuracy, the recognition model should be trained on training examples that are representative of the text in question. In this work, two types of feature sets were used. Both were developed from the dataset of pixel images of handwritten Arabic (Indian) digits developed by Mahmoud [29].

4.2 Data preparation

The data were collected from writers using semitransparent paper over a tabular grid as described in [29]. The data were collected from 44 writers; where each writer wrote 48 samples of each digit (0–9), see Figure 4.1, i.e. a total of 480 digits per writer; for a total of 21,120 digits. The handwritten pages were then scanned using a scanner with a resolution of 300 pixels/in. The scanned document images were transformed into binary images (viz. black and white). The black pixels represent the text lines and are given a value of one, while the white pixels represent the background and are given a value of zero. For each scanned page the horizontal histogram was computed, and used to extract numeral lines. Next, for each line the vertical histogram was computed which is used to specify the location of each digit in the line. Each extracted digit was saved in a separate file.

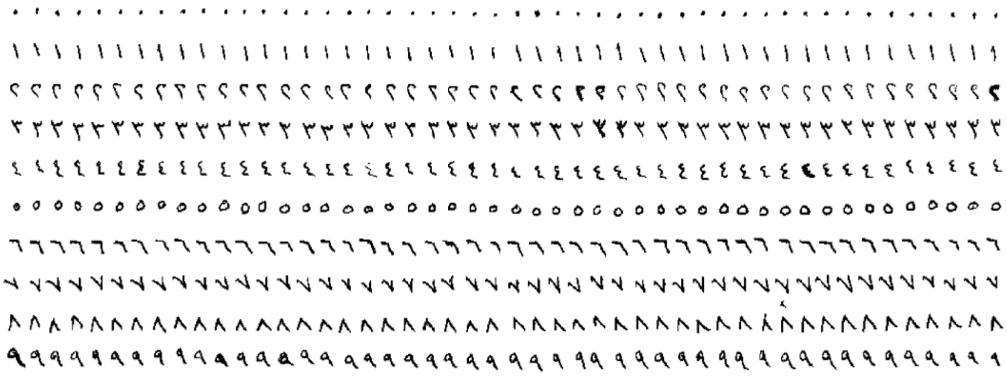


Figure 4.1: Data collected from one writer. Each digit is written 48 times.

4.3 Angle, Circle, Horizontal and Vertical features

The feature set developed by Mahmoud [15] consists of four different categories, namely: 72 angle features, 8 circle features, 20 horizontal features and 20 vertical features. Therefore, the overall dataset consists of a total of 120 features. The centre of gravity (COG) of the pixel image of each digit was determined and the image was segmented depending on the feature category. For example, for the angle features, the numeral image was sliced using angular lines at an angle from the COG and the ratio of the number of black pixels in each slice to the segment size in pixels was computed as shown in Figure 4.2. Similarly, for circle feature, concentric circles of different radii from the centre of the numeral image were used. The circle feature values were calculated by summing up the black pixels between two consecutive concentric circles divided by the total number of black pixels in the digit pixel image; see Figure 4.3 [15].

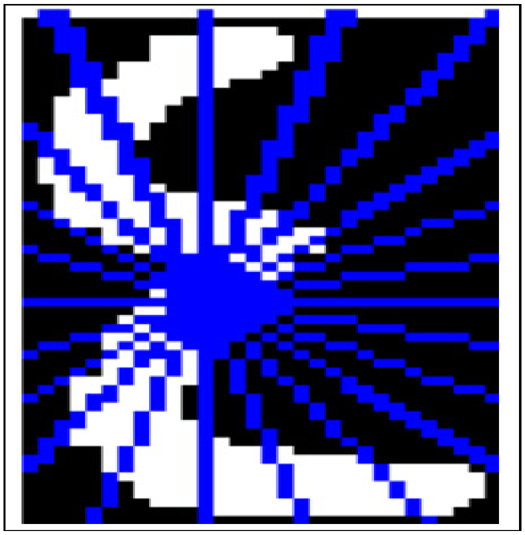


Figure 4.2: Angle slicing of Arabic (Indian) digit 4

Similarly, the horizontal and vertical feature values were computed by dividing the whole image into a number of equal horizontal and vertical bars; see Figure 4.4. The number of bars in each dimension is taken, in this case, as 20 in the horizontal and 20 in the vertical directions. For each digit the numbers of black pixels in the horizontal and vertical bars were calculated. Again, the features extracted are ratios of absolute pixel numbers.

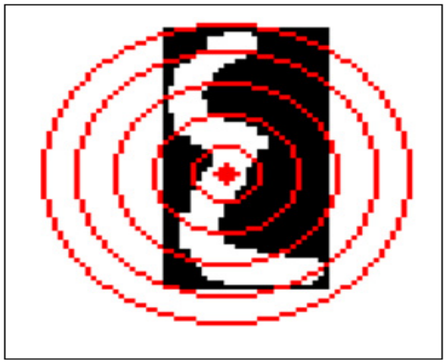


Figure 4.3: Concentric circles used for calculating the circle features of Arabic (Indian) digit 4

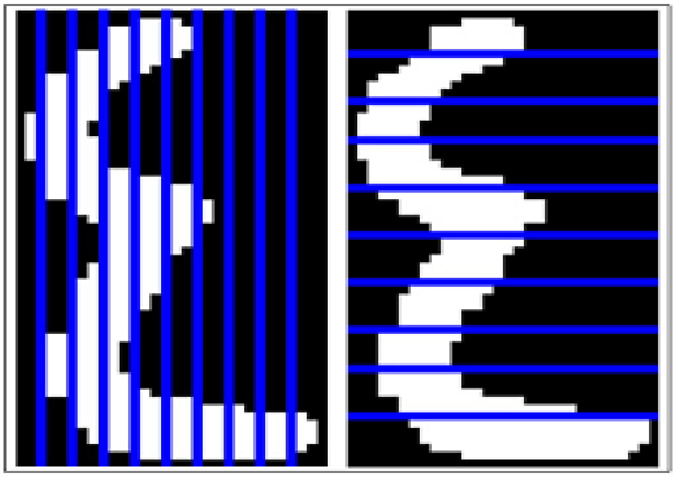


Figure 4.4: Horizontal and Vertical slices of Arabic (Indian) digits 4

4.4 Contour points chain codes features

Histograms of directional chain codes of the contour points of handwritten Arabic (Indian) digits image were used as the second feature set for recognition in this work. This new set of feature was developed by the author of this work. The feature set proved to be simple and effective. Before discussing the whole feature extraction techniques, it is important to highlight some important concepts of contour points and chain codes.

4.4.1 Determination of object contour

A prerequisite for extracting the chain codes is finding the object contour. Given a two-tone image (i.e. 1 being object value while 0 is the background value) the contour points of the image can be determined by the following algorithm. For every object point (P) in the image, consider a 3 x 3 window surrounding the point as shown in Figure 4.5. If any of the four neighbouring point (X) is a background point then this object point (P) is considered as contour point, otherwise it is a non- contour point.

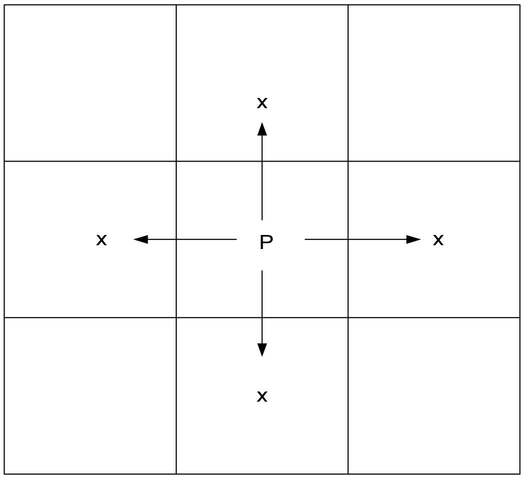


Figure 4.5: Object point P and its four neighbours Xs

The object contour algorithm was applied to the Arabic (Indian) digit pixel image in this thesis to determine the image contour, see Figure 4.6. The image contour consists of the set of contour points.

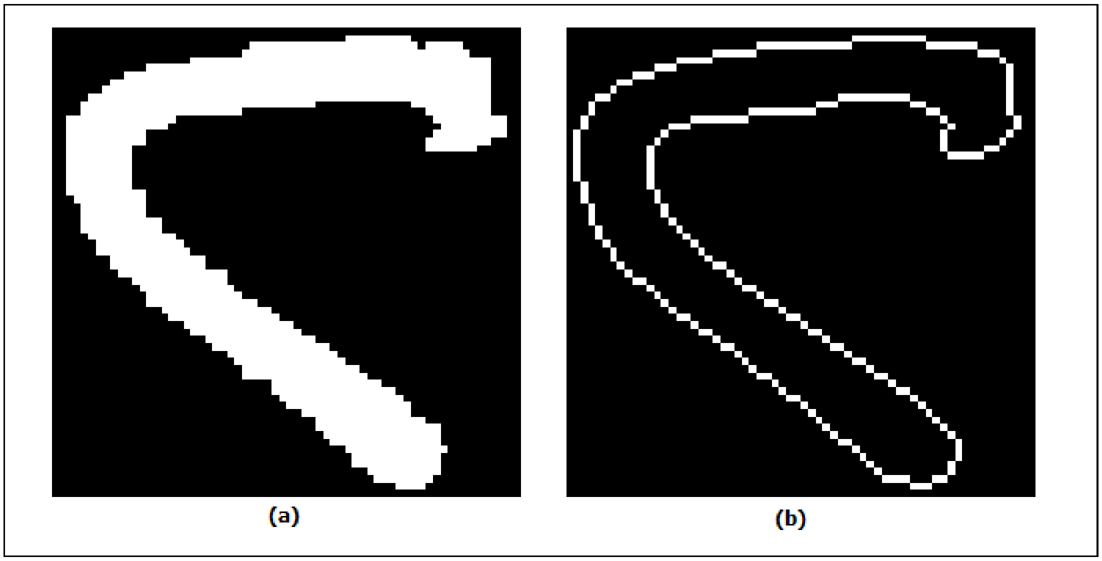
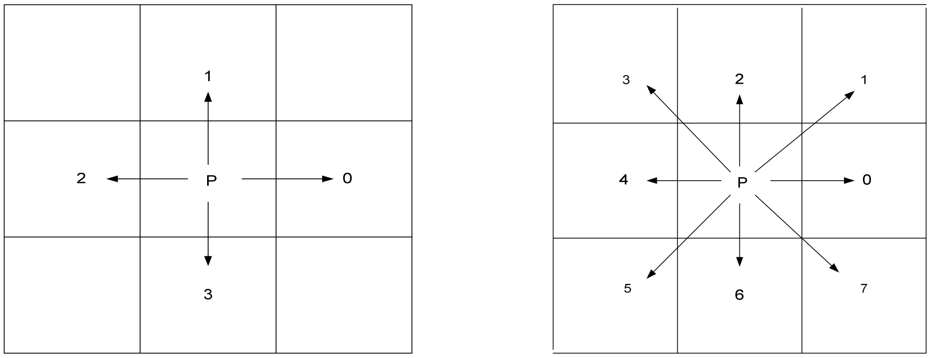


Figure 4.6: (a) Normalized Hadwritten Arabic (Indian) digit 2 pixel image. And (b) is the image contour

4.4.2 Chain codes

Chain code is a type of image representation that has been commonly used in image processing fields such as feature extraction and image compression. Since introduced at the first time, the evolution and improvement of chain code representation scheme has been an active topic for research. This is because it preserves information and allows considerable data reduction. Chain codes provide a very compact region representation, suitable for detecting such feature of a region as sharp corners, area, perimeter, moments, centres, eccentricity, projection and straight-line segments [30]. Thus it is more often used in shape based pattern recognition and image analysis. The first chain code scheme introduced by Freeman in 1961, known as Freeman Chain Code (FCC) [31], is used here. This code follows the boundary in a counter clockwise manner and keeps track of the direction as it goes from one contour pixel to the next.

The Freeman chain code is a sequence of directions of the steps taken when following the contour of an image. The codes involve 4-connected (Figure 4.7a) and 8-connected (Figure 4.7b) paths.



(a)

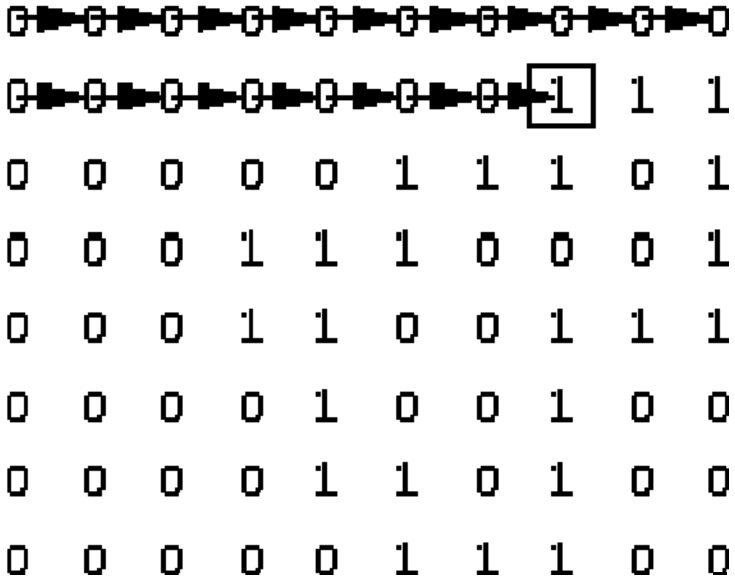
(b)

Figure 4.7: Four and Eight possible directions of a contour point and the associated freeman's chain code

If the codes involve 4-connected, it needs four values (two bits) per pixel to encode the direction to the next pixel, i.e. 0, 1, 2, and 3. While, if the contour is 8-connected, it needs eight numbers (three bits), i.e. 0, 1, 2, 3, 4, 5, 6, and 7.

The image contour tracing algorithm for 4-connected is defined as [32]

1. Search the image from top left until a pixel P_0 belonging to the region is found. See Figure 4.8.



0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1	0	1
0	0	0	1	1	1	0	0	0	1
0	0	0	1	1	0	0	1	1	1
0	0	0	0	1	0	0	1	0	0
0	0	0	0	1	1	0	1	0	0
0	0	0	0	0	1	1	1	0	0

Figure 4.8: Search for initial point

For 4-connected or 8-connected code, initially assign *d* = [(n/2 +1) mod n], where n is the number of directions (i.e. 4 or 8) as the case may be and *d* is the code value.

2. Search the neighbourhood of the current pixel for another pixel P_i of the boundary in an anti-clockwise direction beginning with direction $(d + 3)\ mod\ 4$. Update the value of *d*.

3. If the current boundary element is equal to P_1 and the previous was $\square_0$, then stop. Otherwise, goto step 2.

4. The detected inner border is represented by pixels $p_0 \ldots . p_{i-2}$.

Step 2-4 determine the boundary of the region as shown below

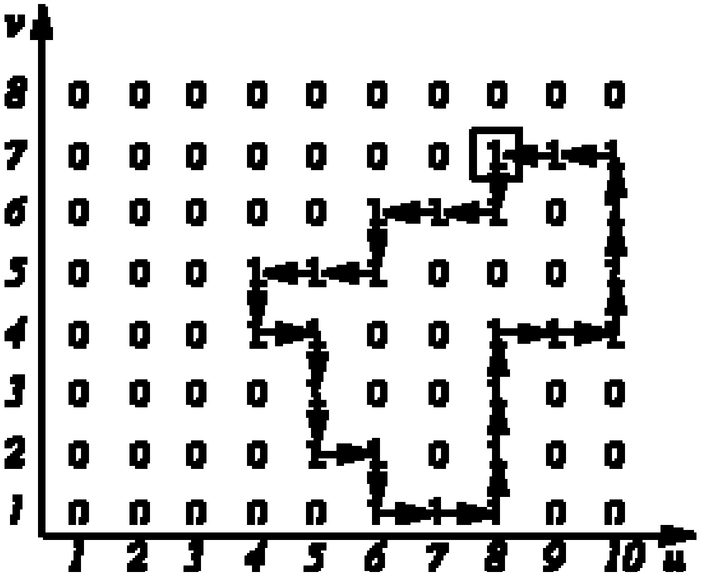


Figure 4.9: Inner boundary tracing

While the algorithm is running, the changing values of *d* define the Freeman chain code 3223223033303001110011122.

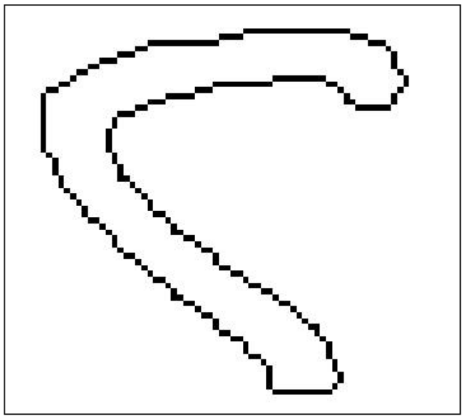
4.5 Application of Freeman chain codes techniques to Arabic (Indian) digits

Figure 4.11 shows the outline of the feature extraction of the contour points chain codes applied on a handwritten Arabic (Indian) digit 2. In this thesis, the author used a chain code of eight directions. In the 8-connected FCC, each code can be considered as the angular direction, in multiples of 45^0 that we must move to go from one contour pixel to the next [direction 0 (horizontal right), 1 (45 degree slanted), 2 (90 degree), 3 (135 degree), 4 (180 degree), 5 (225 degree), 6 (270 degree) and lastly 7

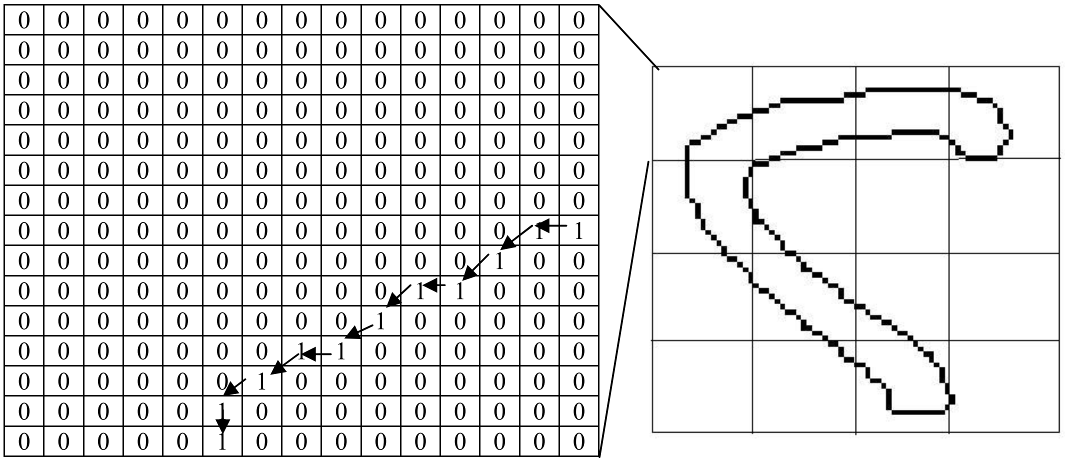
(315 degree)] so as to cover all possible directions of the digit image contour points. See Figure 4.7b for illustration of the eight chain code directions. Also, the Freeman chain code algorithm described in section 4.4.2 was designed for a closed boundary; however the author here modified the algorithm to suit the disjoint nature of the contour points in this work. Similarly, the author ensured that no contour point is visited twice by eliminating the point (i.e. turn pixel value 1 to 0) after it has been visited once. See APPENDIX 3 for the matlab code.

After the pre-processing phase which involves normalization of position and size of the original digit image to 60 x 60 pixels as shown in Figure 4.11a, the bounding box (minimum rectangle containing the digit) is then divided into 4 x 4 (in other experiments, 2x2 and 3x3) blocks as shown in Figure 4.11b. For each of the blocks the directional chain codes for each contour point were determined and the frequency of the directional codes was computed as illustrated in Figure 4.11c.

Thus, in each block an array of eight integer values representing the frequencies of the codes are recorded and these frequencies are horizontally concatenated for all blocks to represent the new feature set. Hence, for a 4x4 blocks a total of 4x4x8 = 128 features is generated. In a similar development, the author computed the percentage of black pixels in each of the 4x4 image blocks and a total of 16 separate features were generated. Figure 4.12 shows a normalized, contoured and 3x3 segmented image of handwritten Arabic (Indian) digit 9 used in this work.



(a)



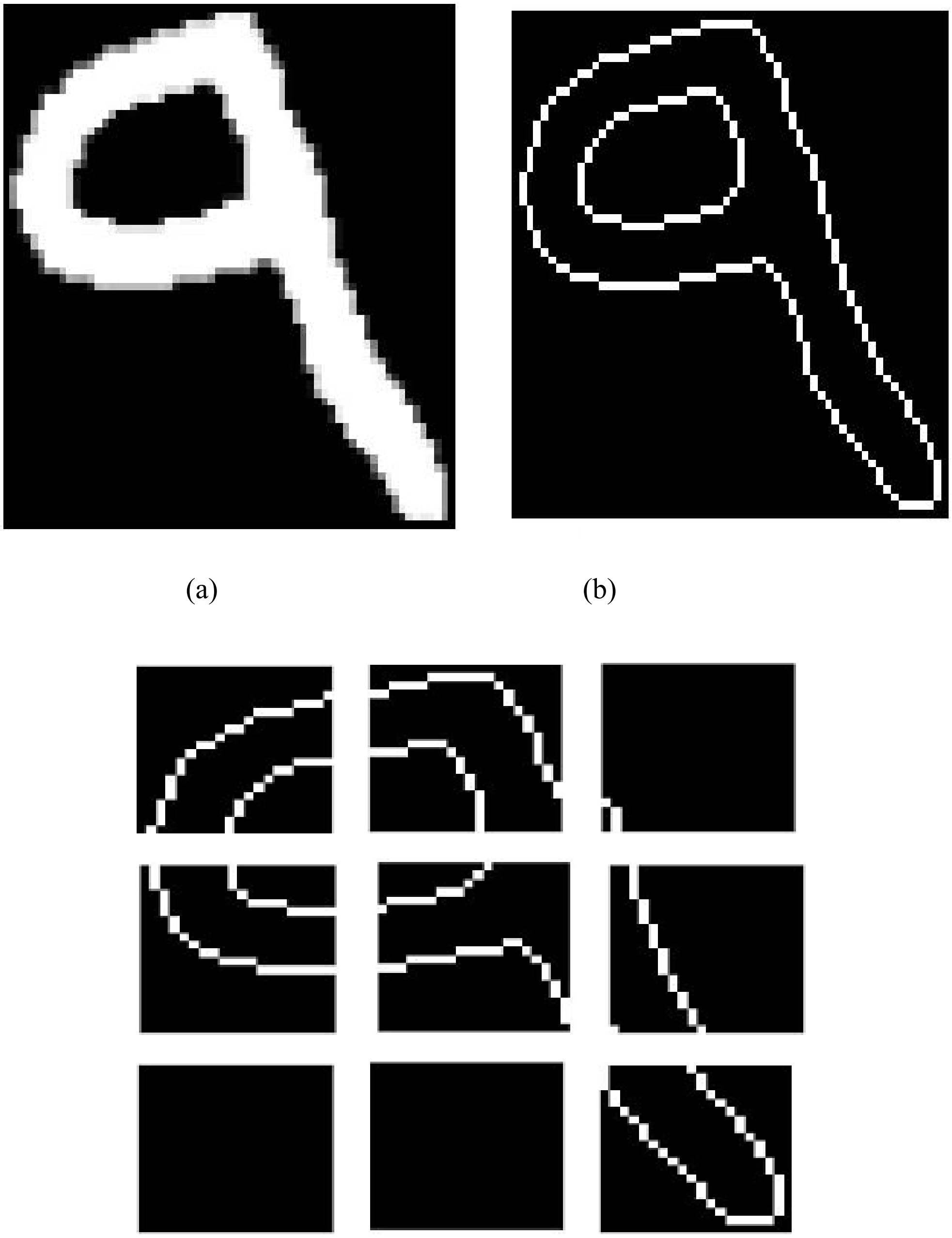
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

(b)

Chain code	0	1	2	3	4	5	6	7
Count	0	0	0	0	3	6	1	0

(c)

Figure 4.11: Pictorial representation of the contour point chain code feature extraction process for a sample of handwritten Arabic (Indian) digit



(a) (b)

(c)

Figure 4.12: Normalized, contoured and 3x3 blocks of Arabic (Indian) digit 9

CHAPTER 5

EXPERIMENTAL RESULTS AND DISCUSSION

5.1 Introduction

In this study, various abductive network architectures have been developed for the recognition of handwritten Arabic digits, this includes monolithic and committee models. In what follows, we presented the development and evaluation of our proposed techniques for handwritten Arabic digit recognition. We also presented a comparison of results obtained in this work with earlier published work on the same dataset. We adopted the same splitting used by earlier published work using the dataset to allow direct comparison of results, i.e. 15840 cases of handwritten Arabic digits for training set and 5280 cases for evaluation set. The criteria for performance evaluation are:

1. Recognition rate obtained with each classifier

2. Time complexity in training and evaluation of each classifier type

3. Number of features needed for recognition of the handwritten Arabic digit

5.2 Monolithic abductive models using the angle, circle, horizontal and vertical feature subsets

In the first experiment using the feature set developed by Mahmoud [15], the full training set was used to develop 10 abductive network models, one for each of the 10 digits using each of the four feature subset (angle, circle, horizontal and vertical), with all their corresponding features enabled as model inputs. For example Fig. 5.1 shows the abductive network models synthesized using the 20 vertical feature subset using CPM = 3.5. The number (e.g. Var_1) indicated at the model input represents the feature selected as input to the model during training, while Var_21, Var_22 . . . Var_30 represent the 10 network outputs corresponding to digits 0,1, …, 9 respectively. During model evaluation on the evaluation set, the winner-takes-all approach was used to determine the output of the combined model as the digit whose network has the largest output, see APPENDIX 1 for figures showing recognition rate obtained for abductive models synthesized using different CPM values for each feature subset used. Table 5.1 gives the model complexity parameter, the number of features automatically selected for the optimum model during training, and the recognition rate obtained using each of the four feature subsets. The table indicates that automatic feature selection by AIM achieves a 50% overall data reduction through reducing the number of required features from 120 to 60.

However, the experiment shows that the circle and vertical feature subset representation for most of the digits especially digit ٧ and ٨ is inadequate due to high correlation between the feature vectors for these digits. As seen in Table 5.2 and Table 5.3, the optimum abductive model could not differentiate properly between two digits resulting to high error rate. The recognition rate for digit ٨ as shown in Table 5.2 is 44.13% and this poor performance is partly due to 25% of the digit been mistaking as digit ٧.

Table 5.1: Optimum model complexity, recognition rate, number of available features and number of selected features for the optimum models synthesized using the four feature subsets

Features subset	CPM for optimum model	Recognition rate %	Number of available features	Number of selected features
Angle	3.5	96.53	72	27
Circle	1.0	78.48	8	8
Horizontal	0.1	90.74	20	13
Vertical	3.5	72.31	20	12
Total number of Features			120	60

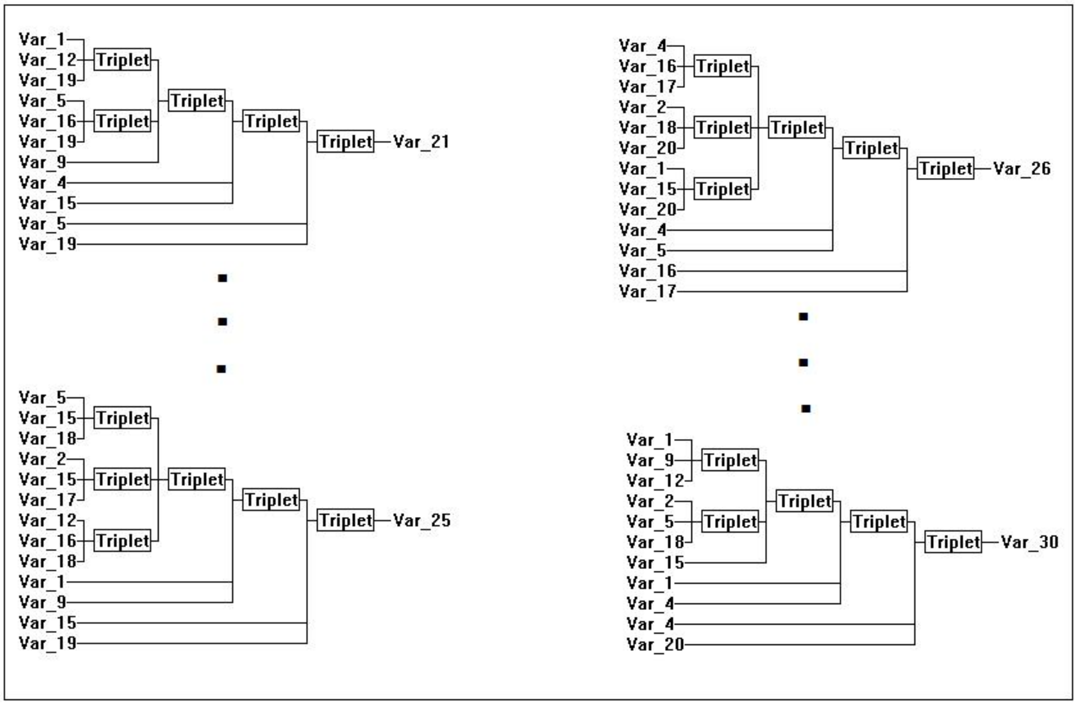


Figure 5.1: Structure of four of the ten abductive networks synthesized with CPM = 3.5 using the vertical feature subset. Var_21 is digit 0 and Var_30 is digit 9

Table 5.2: Confusion matrix showing the percentage recognition rate obtained using vertical feature subset (CPM =3.5).

٠-٩	٠	١	٢	٣	٤	٥	٦	٧	٨	٩	%c
٠	460	29	15	1	5	16	0	2	0	0	87.12
١	41	418	30	12	11	7	1	4	2	2	79.16
٢	36	5	277	16	16	51	1	11	56	59	52.46
٣	3	12	5	379	17	1	2	34	36	39	71.78
٤	4	6	6	26	464	1	0	0	13	8	87.88
٥	6	7	39	0	0	433	0	8	3	32	82.00
٦	1	3	0	1	0	3	466	27	2	25	88.26
٧	10	4	16	4	1	16	10	327	12	38	61.93
٨	2	5	44	6	5	13	41	131	233	48	44.13
٩	3	4	19	10	15	7	36	67	6	361	68.37
Overall											72.31

Table 5.3: Confusion matrix showing the percentage recognition rate obtained using Circle feature subset (CMP =1)

٠-٩	٠	١	٢	٣	٤	٥	٦	٧	٨	٩	%c
٠	457	1	1	3	19	0	0	45	0	2	86.55
١	2	514	0	0	9	0	0	0	0	3	97.35
٢	10	0	427	3	12	6	8	51	2	9	80.87
٣	1	0	2	421	0	0	20	44	8	32	79.73
٤	6	22	8	1	412	0	50	13	4	12	78.03
٥	4	2	13	0	0	507	1	0	0	1	96.02
٦	0	4	10	55	12	0	301	71	26	49	57.01
٧	31	0	11	11	18	2	38	205	211	1	38.83
٨	4	0	30	10	4	0	12	60	397	11	75.19
٩	1	1	16	21	34	0	17	12	5	421	79.73
Overall											76.93

5.3 Monolithic abductive models using the best 60 selected features

In the second experiment, the best 60 features automatically selected from the four feature subsets were used to synthesize a set of ten models for recognizing the 10 digits. In the evaluation phase, the model output with the highest value was chosen as the class of the digit represented by the input vector. The optimum model developed using CPM = 4.5 gave the highest recognition rate of 98.94%. Table 5.4 shows the confusion matrix obtained when this model was evaluated using the evaluation set. This experiment indicates a significant improvement in classifying the digit ٨ compared to only vertical feature subset (i.e. from 44.13% to 99.62%), because the misclassification between digit ٧ and digit ٨ with vertical feature subset only have been cancelled out by other more representative feature subset in the combined model. The combined model in this case uses 44 inputs from the optimum 60 inputs features which indicated a further reduction in data.

Table 5.4: Confusion matrix showing the percentage recognition rate obtained using the automatically selected features combined (CPM =4.5)

٠-٩	٠	١	٢	٣	٤	٥	٦	٧	٨	٩	%c
٠	526	1	0	1	0	0	0	0	0	0	99.62
١	0	524	2	1	0	0	0	0	0	1	99.24
٢	0	1	526	0	1	0	0	0	0	0	99.62
٣	1	1	0	518	4	0	0	0	0	4	98.11
٤	1	3	0	1	518	0	3	0	0	2	98.11
٥	0	0	0	0	0	526	0	0	1	1	99.62
٦	1	1	1	0	1	0	524	0	0	0	99.24
٧	1	0	0	0	0	0	0	525	1	1	99.43
٨	1	0	0	0	0	0	0	0	526	3	99.62
٩	5	3	1	1	1	0	3	0	1	513	97.16
Overall											**98.94**

5.4 Abductive network committee model using the angle, circle, and horizontal feature subset

To further improve the recognition rate, the concept of abductive network committees described in section 3.4 was used in this experiment. Initially, a 3-member abductive network committee model was constructed by training each committee member model on different feature subset (i.e. angle, circle and horizontal feature subsets) and combining the output of the committee to determine the final digit class. Each of the 3-member models used the full training set available with the all the features enabled as inputs. The overall committee output was determined through averaging and majority voting between the outputs of the three members. Evaluation results for the network committee model indicate slight drop in recognition rate to 98.35%. This drop in performance maybe partly due to high correlation among the feature vector representing the digits in circle feature. So, the member contributes negatively by pulling down the performance of other committee members. However, in the second part of this experiment, each of the 3-member committee models was trained with a subset of the available training set instead of the feature subset, using the 60 optimum features selected by the monolithic model in section 5.3. It is expected that the diversity created among the committee members trained on mutually exclusive subsets of the training set should help improve recognition accuracy. Evaluation results for the network committee model indicate a slight improvement in recognition accuracy, with an overall recognition rate of 99%, see Table 5.5. The plot in Figure 5.2 compares the recognition performance of the single (monolithic) model using the selected 60 features, the network committee model, and the best models developed earlier using the four feature subsets separately.

Table 5.5: Comparison of recognition rates of two committees using two methods of combining members output

Outputs Combination Method		Recognition rates %	
		3-member committee, trained on subset of training set & all best 60 features	3-member committee, trained on angle, circle and horizontal feature subsets & full training set
1	Majority voting	99.00	96.80
2	Simple averaging	98.98	98.35

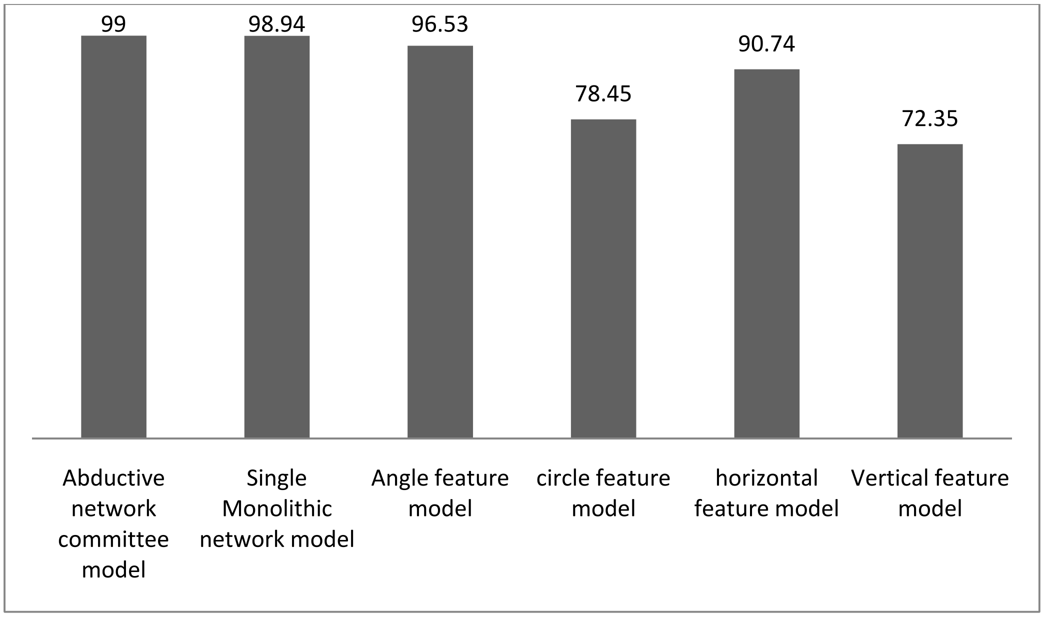


Figure 5.2: Recognition rates in percentage achieved by various abductive network classifiers

5.5 Monolithic abductive model using the FCC code feature set

In this we present the result obtained using the new contour points chain codes feature set described in chapter 4. Initially, the 32 features generated using the 2x2 block segments were used to synthesise an abductive network model. The full training set was used to develop 10 abductive network models, one for each of the 10 digits with all the 32 features enabled as model inputs. The synthesized models at CPM=3.5 uses 29 inputs out of the 32 inputs, which indicates that there is little redudancy in the new feature set, SEE APPENDIX 2 for the synthesized model network. During model evaluation using the evaluation set, the same winner-takes-all approach was used to determine the output of the combined model as the digit whose network has the largest output. Average recognition rate of 99.03% was obtained. Figure 5.3 show the recognition rate obtained for each digit.

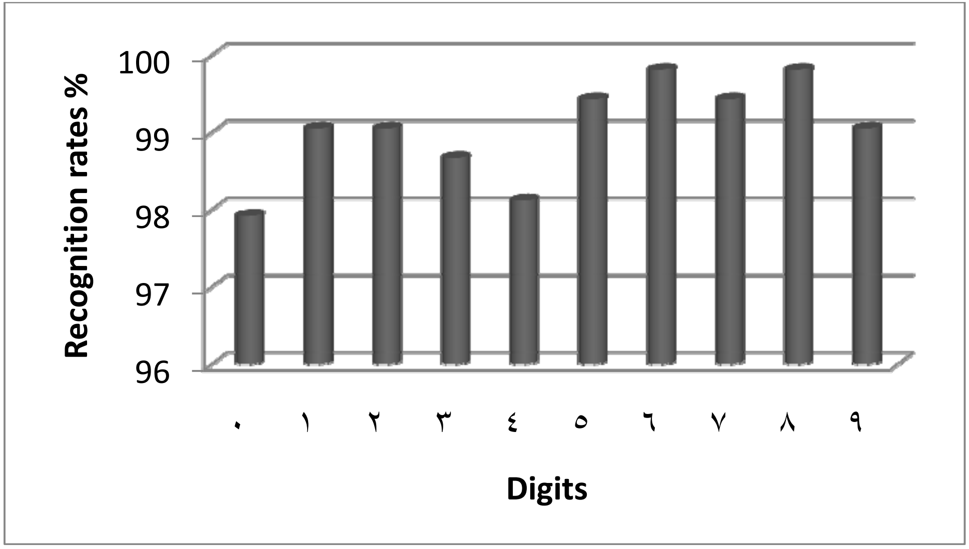


Figure 5.3: Recognition rate achieved for each digit with abductive network model (CPM=3.5) using 2x2 blocks contour point chain codes features

The same experiment was repeated using the 72 features generated from 3x3 block segments. An average recognition rate of 98.92% was achieved with an optimum abductive network model synthesized with CPM=1. However, the 128 features generated from 4x4 blocks division exceeds the 75 maximum inputs allowed by the AIM tool used. So the features were split into 2 feature subsets each consisting of 68 features. Training on the first 68 features the best AIM models uses 39 features out of the 68 available. Likewise, training on the second 68 features the optimum AIM model selected only 35 features and this brings about 58% combined reduction in features(i.e. from 128 to 74 features). Subsequently, the best 74 features selected were used to synthesize an abductive network model that yielded an average recognition rate of 98.67% on the evaluation set. Figure 5.4 shows the comparison of recognition rate for the 3 different contour points chain codes feature sets used for the experimentation.

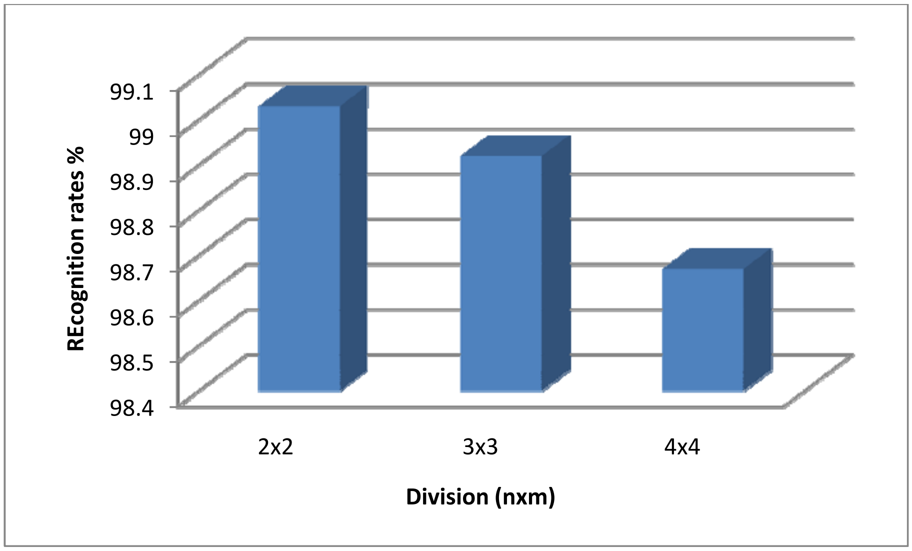


Figure 5.4: Recognition rate at different divisions

It is clear from the figure that choosing a 2x2 division gives the best results with a recognition rate of 99.03% and a considerable reduction in the number of features and processing time.

5.6 Monolithic abductive model using percentage of black pixels features

The 16 features generated by computing the percentage of black pixels in each of the 4x4 block segments described in chapter 4 were used in this experiment. The optimum AIM model developed yielded an average recognition rate of 98.79% on the evaluation set. This features demonstrated a good digits representation and discrimination power, as seen in Table 5.5 all the digits achieved high recognition rate with digit 5 reaching 100%.

Table 5.6: Confusion matrix showing percentage recognition rates obtained using the percentage of black pixels features

٠-٩	٠	١	٢	٣	٤	٥	٦	٧	٨	٩	**%c**
٠	522	3	0	0	0	0	0	2	0	1	98.86
١	1	523	1	0	0	2	0	0	0	1	99.05
٢	1	1	518	1	4	1	0	0	0	2	98.12
٣	0	1	1	520	5	0	0	0	0	1	98.48
٤	0	5	4	1	515	1	0	0	1	1	97.54
٥	0	0	0	0	0	528	0	0	0	0	100.00
٦	0	1	0	0	0	0	526	0	0	1	99.62
٧	1	0	1	2	0	0	0	522	0	2	98.86
٨	3	0	0	0	0	0	0	0	524	1	99.24
٩	0	1	0	1	1	1	0	2	4	518	98.12
Overall											**98.79**

5.7 Abductive network model using combination of contour features and percentage of black pixels features

The next experiment conducted involved the combination of the 32 features generated from the 2x2 block division and the 16 features from the percentage of black pixels computed before. The resulting 48 features were used to train an abductive network model. Model evaluation indicated that the combined features boost the average recognition rate from 99.03% to 99.22%. This performance is the highest so far recorded on this dataset. Figure 5.5 compares the recognition rates obtained with abductive network using different features combinations.

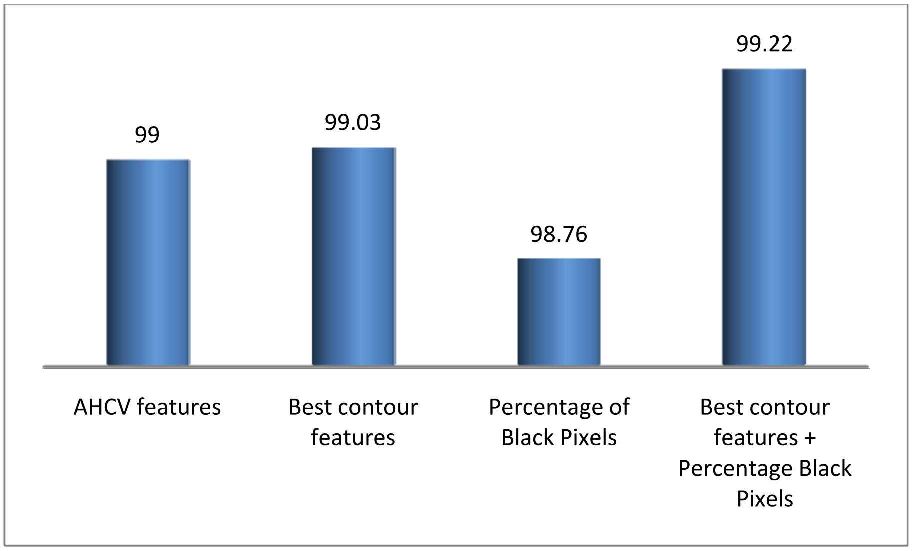


Figure 5.5: Compares the recognition rates percentage obtained with abductive network using different features combinations used in this thesis work

5.8 Artificial neural network model using the contour features

To test the effectiveness of the new feature set with a different recognition tool, an artificial neural network (ANN) model was developed with the same new contour features generated from the 2x2 block segments. The optimum ANN model used here is a multi-layer perceptron (MLP) design with 2 hidden layers each having 30 neurons. The input layer contain 32 neurons while output layer have 10 neurons one for each of the digit (0-9). The back propagation algorithm was employed as the learning algorithm. During evaluation, an average recognition rate of 98.34% was obtained. Table 5.6 show the parameters for the optimal model used and the average recognition rate obtained. Figure 5.6 compares the recognition rates for individual digits obtained with ANN and AIM classifier, while Table 5.8 shows the time complexity for the two models. This experiment demonstrated that even with ANN our contour feature achieves high recognition rate.

Table 5.7: Modelling parameters and average recognition rate for the optimal ANN model

Architecture	Multilayer Perceptron (MLP)			
No. of Neurons	Input layer 32	1^{st} hidden layer 30	2^{nd} hidden layer 30	Output layer 10
Learning Algorithm	Back propagation (BP)			
Average recognition rate	98.34%			

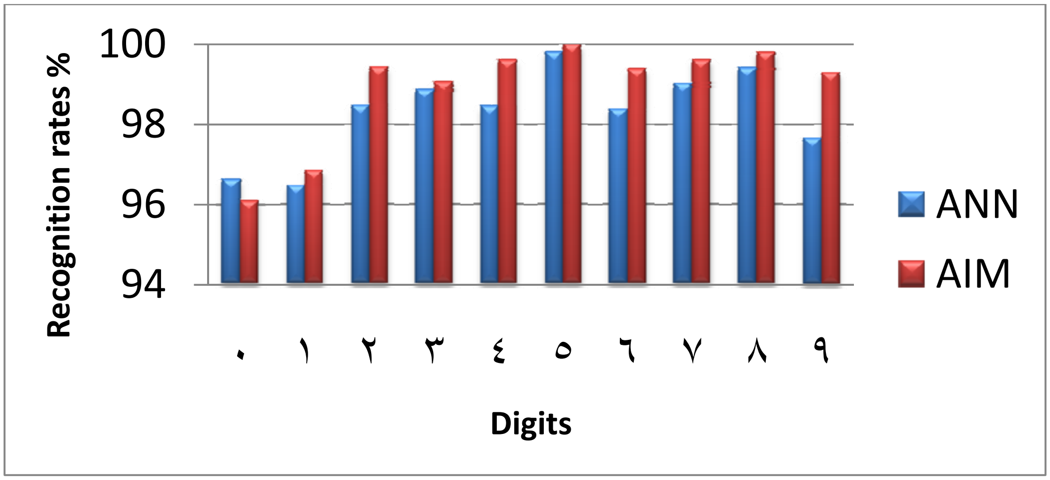


Figure 5.6: Recognition rates of Arabic (Indian) digits using ANN and AIM classifiers

Table 5.8: Time complexity for ANN and AIM classifier in Handwritten Arabic digits recognition

Time Complexity for Handwritten Arabic (Indian) Digit Recognition		
MODEL	TRAINING	TESTING
ANN	24.25 minutes	0.74 seconds
AIM	3.47 minutes	~ 1.0 seconds

Finally, the recognition rates of other modelling tools used previously on the dataset were compared with that of abductive models synthesised in this work as shown in Figure 5.7. The experiments conducted have shown that abductive network perform much better compared to other tools in terms of recognition rate as well as in terms of data reduction.

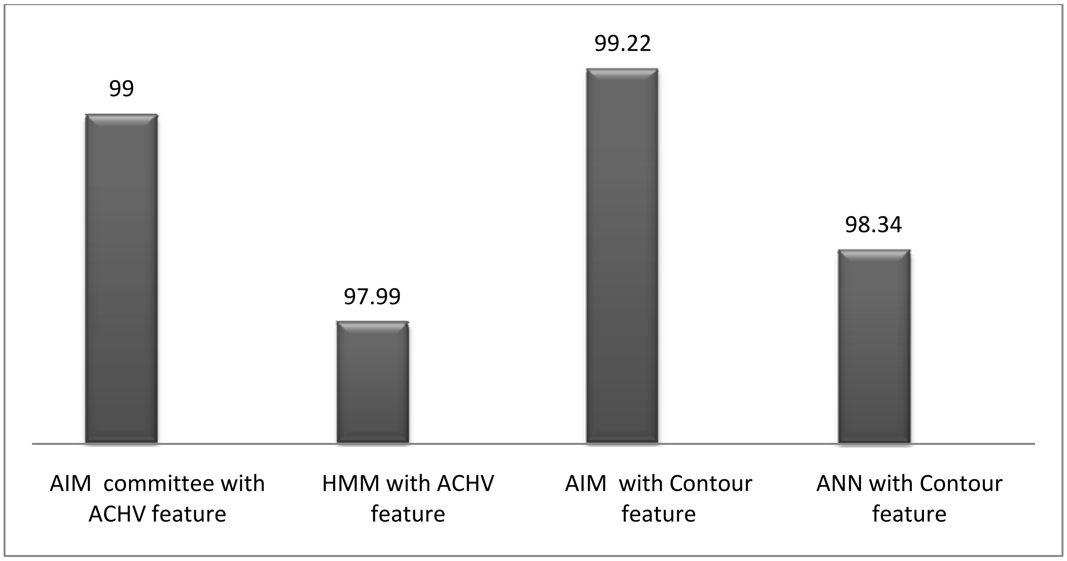


Figure 5.7: Recognition rates in percentage for different modeling tools with different feature set

5.9 Sources of misclassification

After conducting the experiments as discussed in the previous sections, the misclassified samples of the database were examined. The misclassification errors were found to be partly attributed to the following reasons;

- Errors due to bad or corrupted data. Fig. 5.8 shows samples of bad or corrupted data. These errors are difficult to rectify during pre-processing because the samples are corrupted data.
- Errors due to samples written with different style than the style used for training. For example, digit three maybe written with three upward segments or two upward segments. Fig. 5.9 shows samples of same digit written differently. This type of error can be addressed by allowing a digit to have more than one model if it is appreciably different from the basic model.
- Errors due to genuine samples been misclassified with no tangible reason other than insufficient recognition capability of the used features, e.g. the circle and vertical feature subset which causes misclassification of digit ٧ and ٨

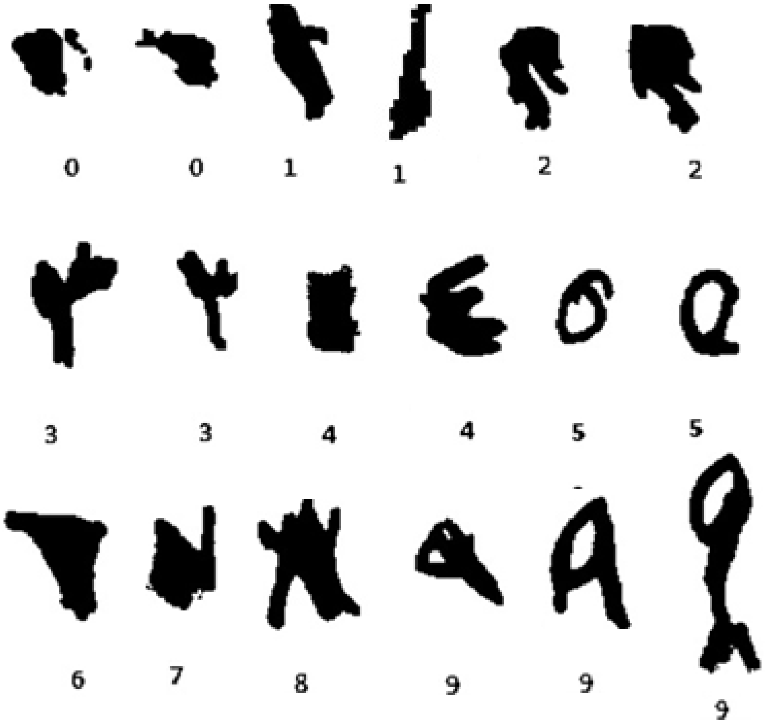


Figure 5.8: Samples of badly written or corrupted data

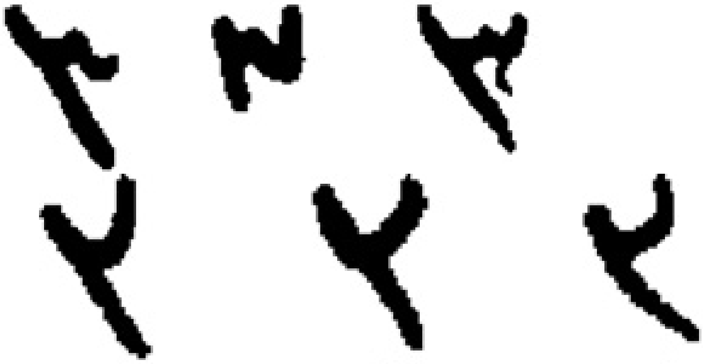


Figure 5.9: Samples of digit 3 written in different style than the style

CHAPTER 6

CONCLUSION

6.1 Summary

This thesis work demonstrates the use of abductive machine learning techniques for the recognition of handwritten Arabic (Indian) digits. Two sets of features were used in this work, and an average recognition rate of 99.0% was achieved with the feature set used in [15] by employing a 3-member committee that utilizes only 60 out of the 120 features available in the feature set. This result indicated that our proposed abductive network approach yields a better performance compared to the 97.99% reported in [15] using HMM models with 120 features. The second feature set of FCC codes which was developed by the writer as described in chapter 4 of this report contains 16, 32, 72 and 128 features. An average recognition rate of 99.03% and 99.22% were achieved with the feature subset containing 32 features only and with a combination of 16 and 32 feature, respectively. The performance obtained here is the highest recorded so far on this dataset. The experiments conducted also show the power of the AIM approach in automatically selecting the best features, thus achieving a significant data reduction and faster training and recognition speeds.

6.2 Contribution to knowledge

This work has achieved the following:

- A comprehensive literature survey has been carried out vis-à-vis Handwritten Arabic (Indian) digit recognition.

- A new approach has been introduced for recognition of handwritten digits used in Arabic. Instead of the trial and error approach adopted by researchers due to the black box nature and lack of clear picture for selecting important design parameter for previous models, abductive network as an alternative learning tool avoids many of the limitation earlier stated. Abductive network uses a well proven optimization criterion for automatically determining the network size, connectivity and coefficient of the optimum model, thus reducing the modelling effort. With this new approach we have achieved a significant improvement in recognition accuracy and smaller feature sets compared to previous work on the same dataset
- A new feature extraction technique based on Freeman's chain codes was proposed which proved to be simple and effective for recognition purposes.
- In essence, this work has proposed accurate, efficient, fast and low cost approaches for the recognition of handwritten Arabic digits.

6.3 Recommendations and future work

One of the greatest challenges that researchers of Arabic Computing field do face is the availability of standard database or adequate data for experimentation. In view of this, there will be need for more collaboration between researchers from relevant fields to come up with standard databases so as to allow direct comparison of research work and to explore more opportunities in the Arabic computing area. However, this work has outlined the advantages of abductive networks and has placed it in the perspective of Arabic computing point of view. Thus, researchers are encouraged to consider them as valuable alternative modelling tool. Hopefully, future work will consider the possibility of extending the approach to the recognition of handwritten Arabic text.

REFERENCES

[1] C.L. Liu, K. Nakashima, H. Sako, H. Fujisawa, “Handwritten digit recognition: benchmarking of state-of-the-art techniques”, Pattern Recognition, vol. 36, 2003, pp. 2271–2285.

[2] B. Al-Badr, S. A. Mahmoud, “Survey and bibliography of Arabic optical text recognition”, Journal of Signal Processing, Vol. 41, 1995, pp. 49–77

[3] L. Lorigo, V. Govindaraju, "Offline Arabic Handwriting Recognition: A Survey”, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 28, no. 5, May 2006

[4] R.K. Mehra “Group Method for Data handling (GMDH): Review and Experience”, Proceedings of the IEEE Conference on Decision and Control, New Orleans 1977, pp. 29-34

[5] www.gmdh.net

[6] F. Al-Omari, “Handwritten Indian numeral recognition system using template matching approaches”, IEEE International Conference on Computer Systems and Applications, Beirut 2001, pp. 83–88

[7] M. Ziaratban, K. Faez, and F. Faradji. “Language-based feature extraction using template-matching in Farsi/Arabic handwritten numeral recognition”, Proc. Ninth International Conference on Document Analysis and Recognition, Brazil, volume 1, 2007, pp. 297–301

[8] S. Behnke, R. Rojas, and M. Pfister “Recognition of Handwritten Digits using Structural Information”, Proceedings of the International Conference of Neural Network, Houston TX, Vol. 3, 1997 pp. 1391- 1396

[9] Romeo-Pakker K., Ameur A., Olivier C., and Lecourtier Y., “Structural analysis of Arabic handwriting: segmentation and recognition”, Machine Vision and Applications, vol. 8, No 4, 1995

[10] F. Bousalma “Structural and fuzzy techniques in the recognition of online Arabic characters”, International Journal Pattern Recognition and Artificial Intelligence vol. 13, no. 7, 1999, pp1027–40

[11] F.A. Al-Omari, O. Al-Jarrah, “Handwritten Indian numerals recognition system using probabilistic neural networks”, Adv. Eng. Inform. Vol.18, 2004, pp. 9–16

[12] S.M Awaideh, Sabri A. Mahmoud, “A multiple feature/resolution scheme to Arabic (Indian) numerals recognition using hidden Markov models”, Signal Processing, vol. 89, 2009, pp. 1176–1184

[13] S. A. Mahmoud, "Arabic Character Recognition using Fourier Descriptors and Character Contour Encoding." Pattern Recognition Journal, Vol. 27, No. 6, June 1994, pp. 815-824

[14] M.H.S. Shahrezaa, K. Faaz and A.Khotanzad, “Recognition of Handwritten Persian/Arabic Numerals by Shadow coding and Edited Probabilistic Neural Network”, Proc. Int. Conf. Image processing, Washington DC, Vol. 3, 1995, pp. 436□439

[15] S. A. Mahmoud, “Recognition of writer-independent off-line handwritten Arabic (Indian) numerals using hidden Markov models”, Signal Processing, vol. 88, 2008, pp. 844-857

[16] J. Sadri, C.Y. Suen, and T.D. Bui, “Application of support vector machines for recognition of handwritten Arabic/Persian digits”, Proc. of Second Iranian Conf. on Machine Vision and Image Processing, vol. 1, 2003, pp. 300–307

[17] H. Al-Yousefi and S. Udpa, “Recognition of Arabic Characters”, IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 14, no.8,1992, pp. 853- 857

[18] H. Soltanzadeh, M. Rahmati, Recognition of Persian handwritten digits using image profiles of multiple orientations, Pattern Recognition Letters 25, 2004, pp. 1569–1576

[19] F.N. Said, R.A. Yacoub, C.Y. Suen, “Recognition of English and Arabic numerals using a dynamic number of hidden neurons”, Proceedings of the Fifth International Conference on Document Analysis and Recognition, IEEE Computer Society, Bangalore, 1999, p. 237

[20] M. M. Altuwaijr, M.A Bayoumi, “Arabic text recognition using neural networks” IEEE Intl. Symposium on Circuit and Systems, 1994.

[21] R.E. Abdel-Aal, “Improved classification of medical data using Abductive network committees trained on different feature subsets”, Computer Methods and Programs in Biomedicine, Vol.80, no.2, 2005, pp. 141-153

[22] E.M. El-Sayed, R.E Abdel-Aal, “Spam Filtering with Abductive Networks”, International Joint Conference on Neural Networks, Hong Kong, 2008

[23] AbTech Corporation, Charlottesville, VA, *AIM User's Manual*, 1990

[24] S.J. Farlow, “The GMDH algorithm,” in *Self-Organizing Methods in Modelling:* GMDH *Type Algorithms*, S.J Farlow, Ed., Marcel-Dekker, New York (1984).

[25] W. S. Lin, B. Y. Lee and C. L. Wu, “Modelling the surface roughness and cutting force for turning”, Journal of Materials Processing Technology, vol. 108, No.3, 2001, pp. 286-293.

[26] T. Kondo, et al. “GMDH-type neural networks and their application to the medical image recognition of the lungs”. Proc. of the 38th IEEE SICE annual conference, 1999, pp. 1181–6.

[27] R.E. Abdel-Aal, “Short-Term Hourly Load Forecasting Using Abductive Networks”, IEEE Trans. on Power Systems, Vol. 19, No. 1, 2004

[28] D. Jimenez, N. Walsh “Dynamically weighted ensemble neural networks for classification”, Proceedings of the international joint conference on neural networks (IJCNN’98), Anchorage, Alaska, May 1998, pp 753-756

[29] <http://faculty.kfupm.edu.sa/ICS/smasaad/Arabic%20Computing.htm>

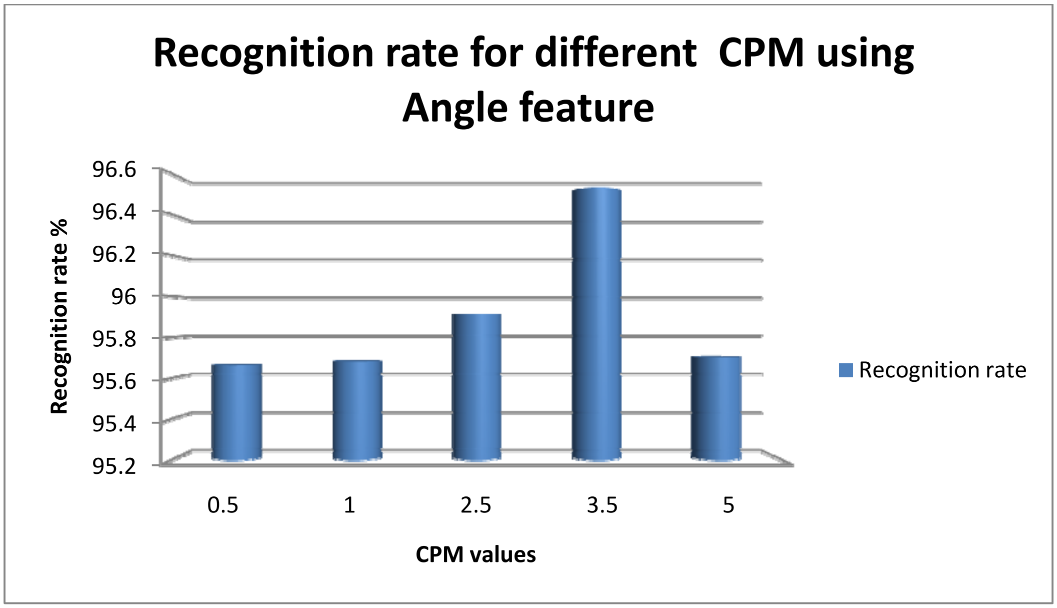
[30] R. Mingwu, Y. Jingyu, and S. Han, “Tracing Boundary Contours in Binary image”, Image and Vision Computing, Vol. 20, 2002, pp.125-131

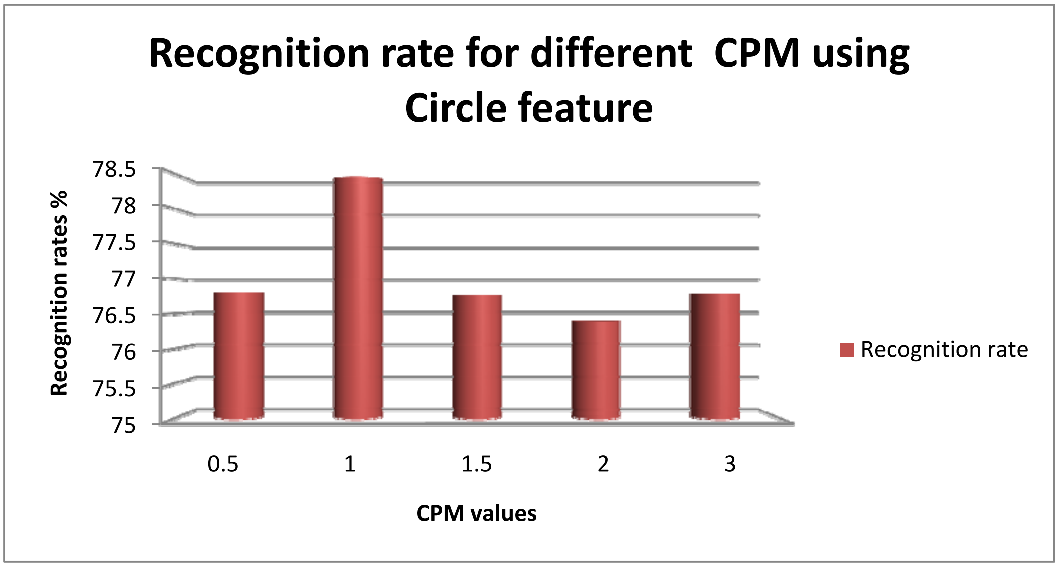
[31] H. Freeman, “Encoding of Arbitrary Geometric Configurations”, IRE Transaction on Electronic and ComputerVol.10, No. 2, 1961, pp. 260-268

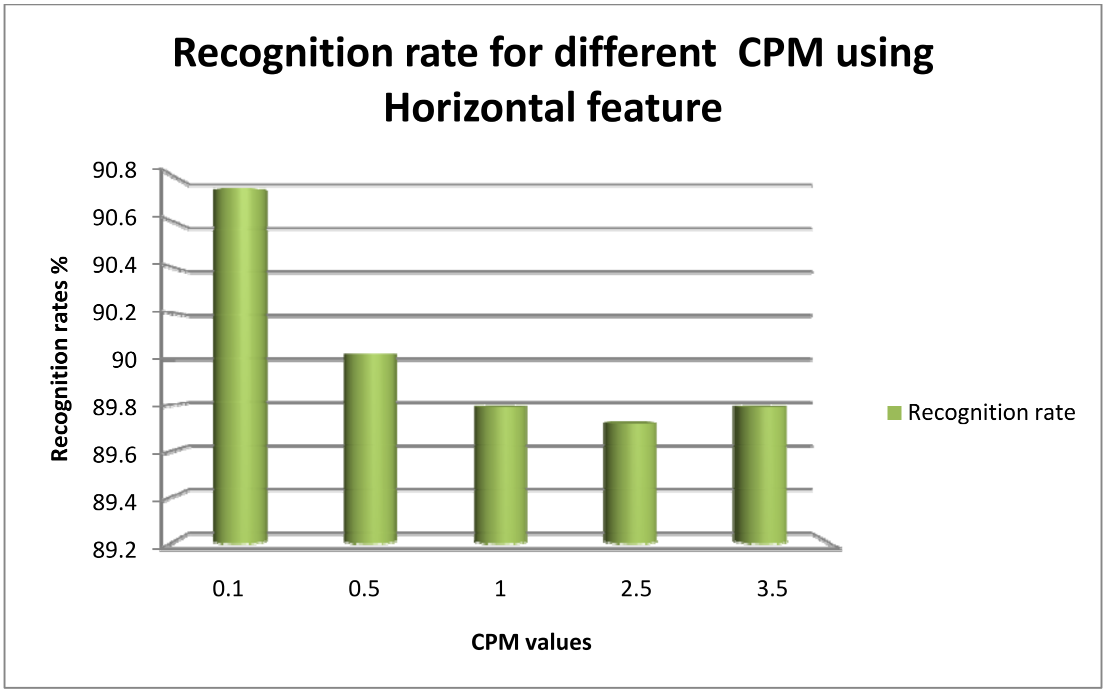
[32] H. Freeman and L. S. Davis, “ A corner finding algorithm for chain coded curves”, IEEE Transaction on Computer, Vol. 26, pp. 297-303

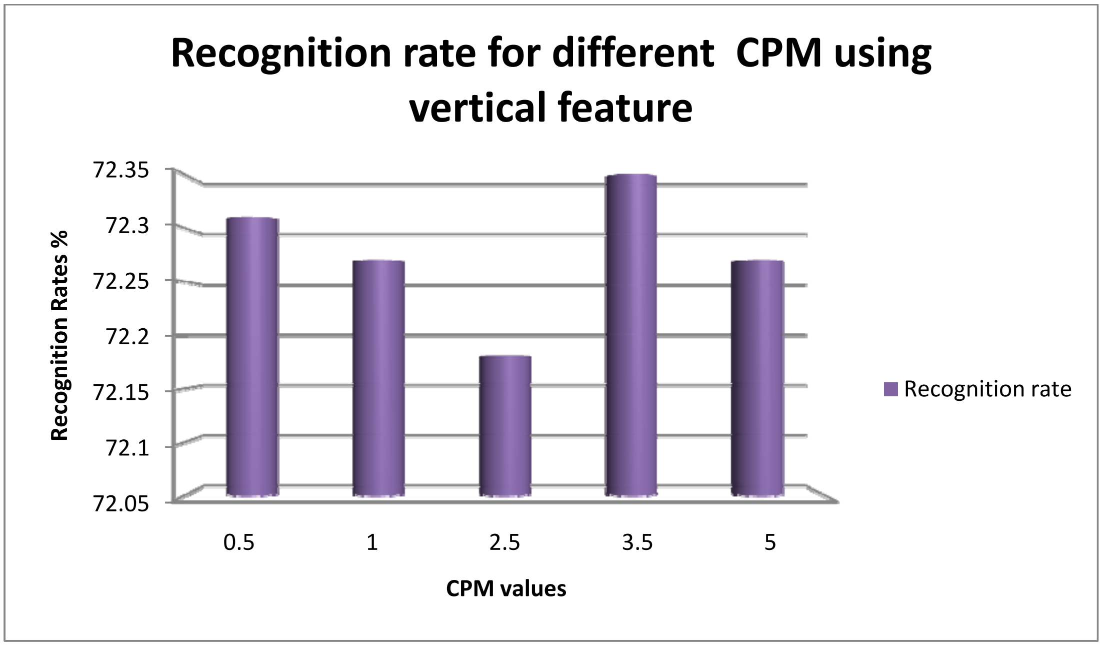
APPENDIX 1: Figures showing recognition rates obtained for different AIM models synthesized with ACHV Feature subsets

The figures below show the recognition rates obtained for different models synthesized using the each of the four feature subset in section 5.2



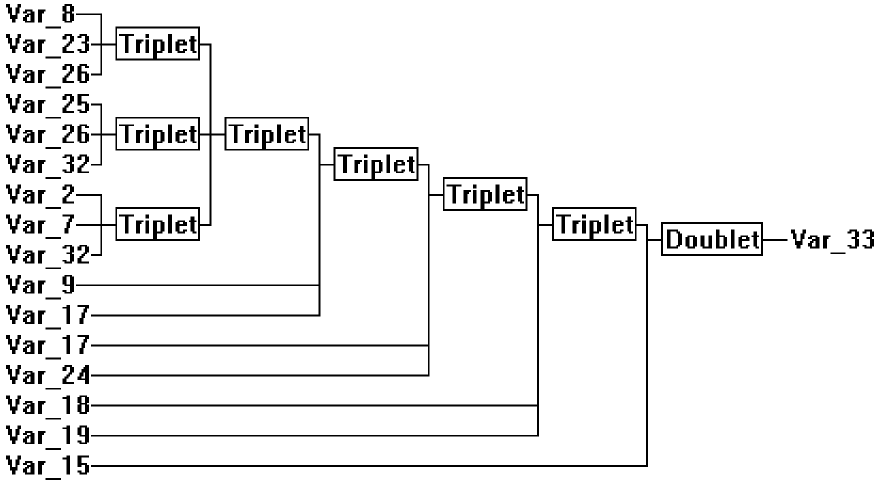


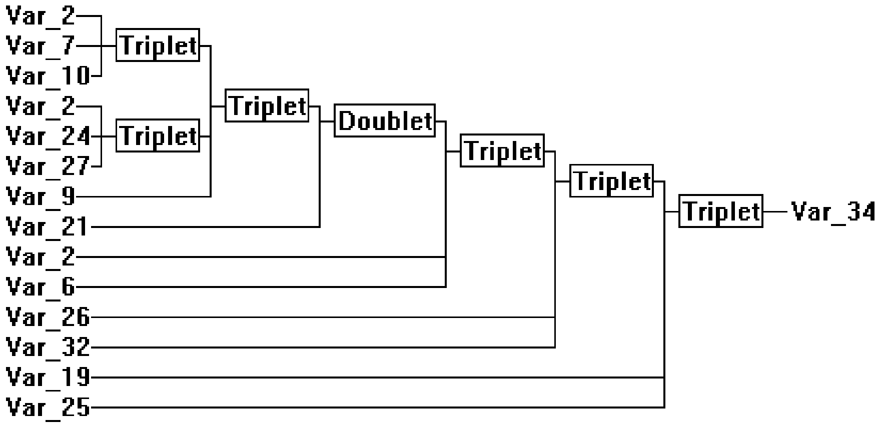

Recognition rate for different CPM using Horizontal feature
90.8
90.6
90.4
90.2
90
89.8
89.6
89.4
89.2
Recognition rates %
0.1
0.5
1
2.5
3.5
CPM values
Recognition rate

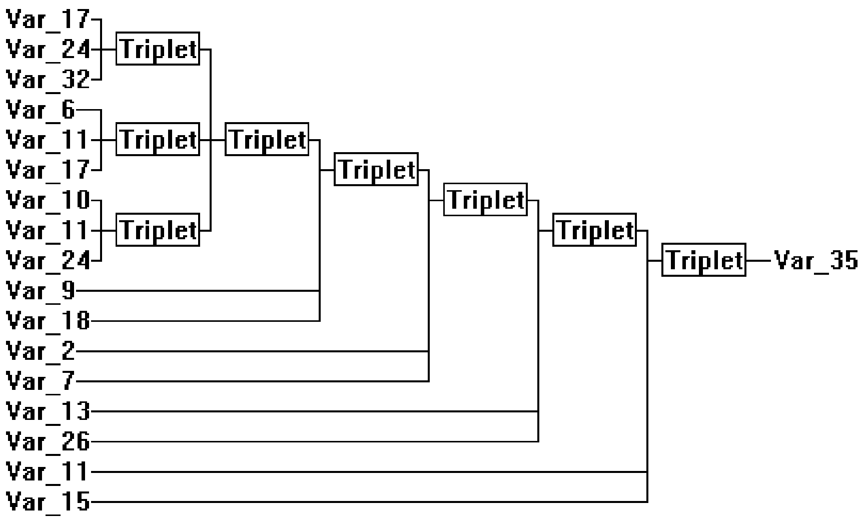

Recognition rate for different CPM using vertical feature
72.35
72.3
72.25
72.2
72.15
72.1
72.05
Recognition Rates %
0.5
1
2.5
3.5
5
CPM values
Recognition rate

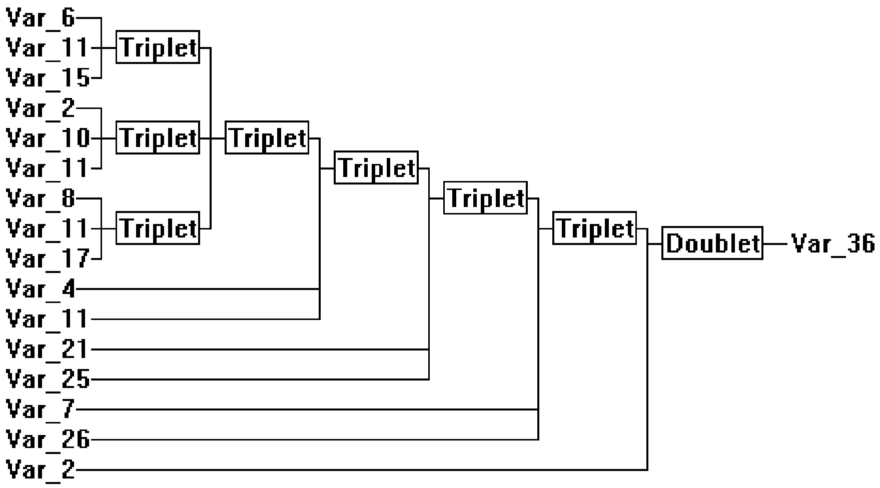
APPENDIX 2: Figures showing the ten synthesized networks using the FCC codes features

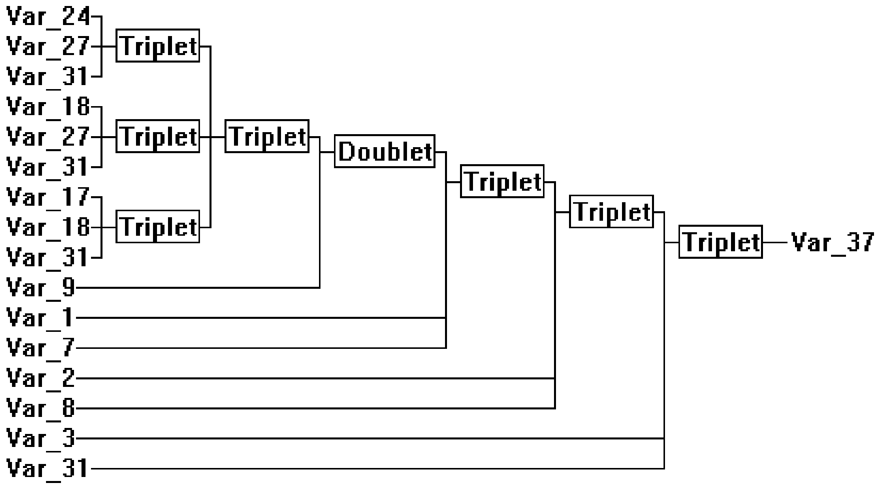
The figure below gives the ten networks one for each digit (0-9) synthesized in section 5.5. Var_33, Var_34… Var_41, Var_42 are the network output representing digit 0, 1… 8, 9.

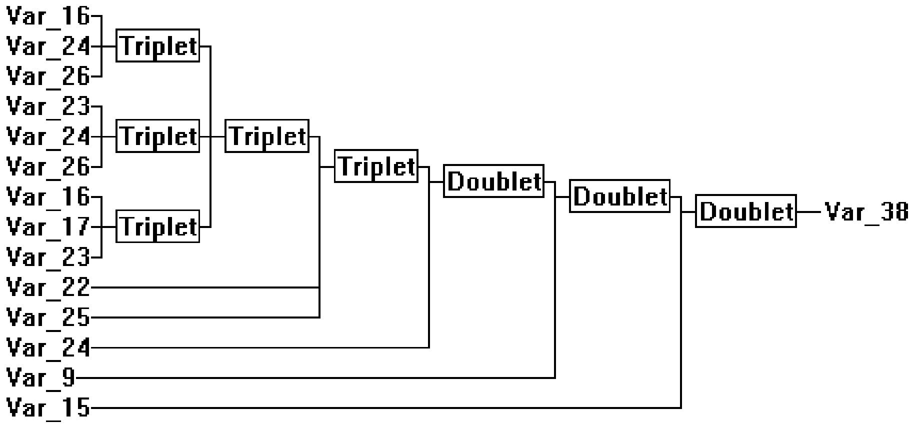


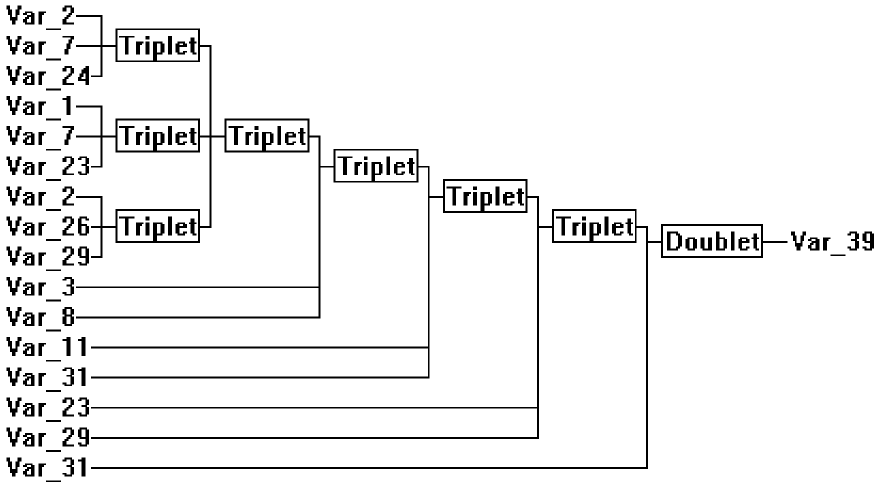


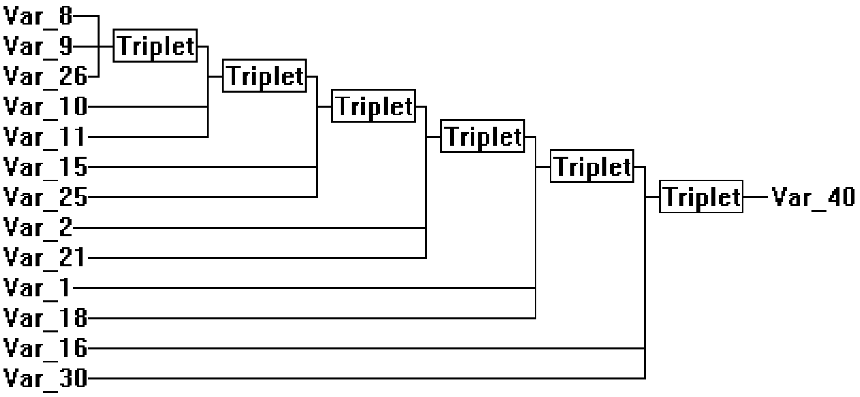

Var_17
Var_24
Var_32
Var_6
Var_11
Var_17
Var_10
Var_11
Var_24
Var_9
Var_18
Var_2
Var_7
Var_13
Var_26
Var_11
Var_15
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Var_35

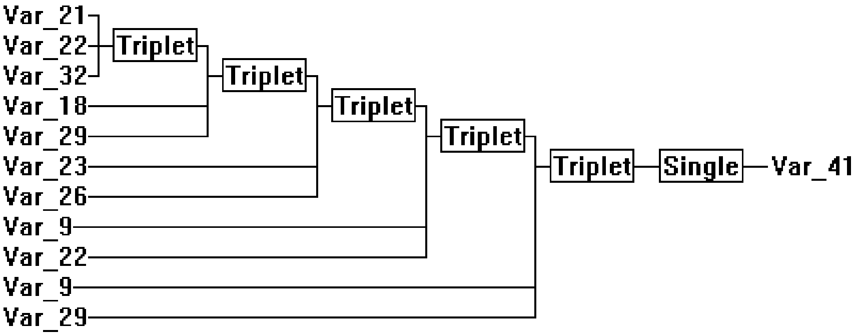

Var_6
Var_11
Var_15
Var_2
Var_10
Var_11
Var_8
Var_11
Var_17
Var_4
Var_11
Var_21
Var_25
Var_7
Var_26
Var_2
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Doublet
Var_36

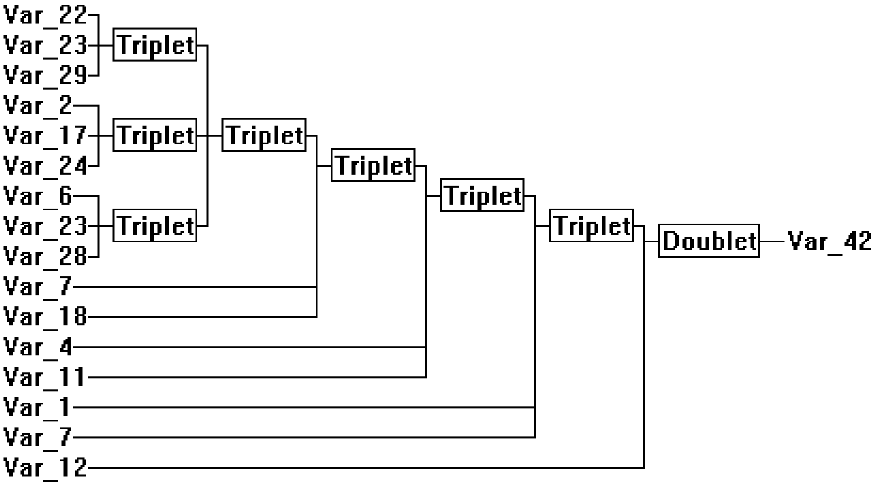

Var_24
Var_27
Var_31
Var_18
Var_27
Var_31
Var_17
Var_18
Var_31
Var_9
Var_1
Var_7
Var_2
Var_8
Var_3
Var_31
Triplet
Triplet
Triplet
Triplet
Doublet
Triplet
Triplet
Triplet
Var_37


Var_16
Var_24
Var_26
Var_23
Var_24
Var_26
Var_16
Var_17
Var_23
Var_22
Var_25
Var_24
Var_9
Var_15
Triplet
Triplet
Triplet
Triplet
Triplet
Doublet
Doublet
Doublet
Var_38


Var_2
Var_7
Var_24
Var_1
Var_7
Var_23
Var_2
Var_26
Var_29
Var_3
Var_8
Var_11
Var_31
Var_23
Var_29
Var_31
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Doublet
Var_39


Var_8
Var_9
Var_26
Var_10
Var_11
Var_15
Var_25
Var_2
Var_21
Var_1
Var_18
Var_16
Var_30
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Var_40


Var_21
Var_22
Var_32
Var_18
Var_29
Var_23
Var_26
Var_9
Var_22
Var_9
Var_29
Triplet
Triplet
Triplet
Triplet
Triplet
Single
Var_41


Var_22
Var_23
Var_29
Var_2
Var_17
Var_24
Var_6
Var_23
Var_28
Var_7
Var_18
Var_4
Var_11
Var_1
Var_7
Var_12
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Triplet
Doublet
Var_42

APPENDIX 3: Matlab codes developed for contour tracing using the FCC algorithm

This code does the contour tracing using the Freeman' chain codes (FCC) algorithm.

```
%% calculate 8 directional chain codes

function directionList = ChaincodeLines8(image)

% define 8 neighbors
  n=[0 1;
    -1 1;
    -1 0;
    -1 -1;
    0 -1;
    1 -1;
    1 0;
    1 1];

  %initialization
  flag = 1;
  cc = [];

  %% find position of first non-zero pixel
  [x y] = find(image==1);
  x = min(x);
  imx = image(x, :);
  y = min( find (imx==1) );

  NextDir = 3;

   %main loop
  while flag==1

    flag = 0;
    for i= 1:8
      j = mod(NextDir + i, 8); % range from 0 to 7
      NextPosition = [x+n(j+1,2)  y+n(j+1,1)];

      % checking if the next position have some pixel
      if (isempty(NextPosition))
        break;
      else

      if ( NextPosition(1) > 0 && NextPosition(1) <= size(image,1) && ...
          NextPosition(2) > 0 && NextPosition(2) <= size(image,2) )

          NextPixel = image(NextPosition(1), NextPosition(2));
          if (NextPixel == 1) %find first nonzero value
             flag = 1;
             cc = [cc j]; % add it to the list of chain code
             image (x,y) = 0;
             break;
          end
       end
    end
  end

     if (flag == 1)
        NextDir = mod(j + 4, 8);
        x = NextPosition(1);
        y = NextPosition(2);
     end
end;
  image(x,y) = 0; % make the last pixel black

  directionList=cc;
 if ((length(find(image==1))>0))

   d  = ChaincodeLines8(image);
     directionList = [directionList d];
  end
```

APPENDIX 4: Matlab codes developed for handwritten Arabic digit pixel image resizing and segmentation

This part of the code does the image resizing, segmentation, and computing the chain codes frequency.

```
resize = [60 60]; % specify resizing value
 sectionNumbers = [4 4]; % block size
feature = [];

% Loading image and preprocessing

[Class
Sample]=xlsread('C:\Users\Eesaah\Documents\new_thesis_work\thesis\TrainSamplePa
th');

 for i=1:length(Sample)

    readImage =double(imread(Sample{i,1}));

    resizedImage =imresize(readImage, resize);

    edgeImage = edge(resizedImage ,'canny'); % contour formation



%%% Dividing the contour image into blocks and finding the chain codes in
%%% each block


    vsections = sectionNumbers(1);  % MUST DIVIDE THE HEIGHT OF IMAGE
    hsections = sectionNumbers(2);  % MUST DIVIDE THE WIDTH  OF IMAGE


    vsize = size(edgeImage, 1)/vsections;
    hsize= size(edgeImage, 2)/hsections;

    sections = zeros(vsize, hsize, vsections, hsections);
u=[];

    for h=1:vsections

        for v=1:hsections
            sections(:,:,h,v) = edgeImage ((h-1) * vsize + 1 : h * vsize , (v-1) * hsize + 1 : v *
hsize);

            out =[];
            out =[out ChaincodeLines8(sections(:,:,h,v))];
         for x=0:7
        u = [u sum(out==x)]; % getting the frequency of the chaincode
        end
     end
    end
  feature (i,:) = u;
 end
 xlswrite ('4x4_traindataset',feature);
xlswrite ('trainclass', Class);
```

APPENDIX 5: Matlab codes used to determine digit classification from the 10 outputs of the abductive models

```
clc
clear all
close all

 %%% % c1, c2,..,c10 representthe output of the ten AIM models

c1= xlsread('VAR_33_I.xls','d2:d5281');
c2= xlsread('VAR_34_I.xls','d2:d5281');
c3= xlsread('VAR_35_I.xls','d2:d5281');
c4= xlsread('VAR_36_I.xls','d2:d5281');
c5= xlsread('VAR_37_I.xls','d2:d5281');
c6= xlsread('VAR_38_I.xls','d2:d5281');
c7= xlsread('VAR_39_I.xls','d2:d5281');
c8= xlsread('VAR_40_I.xls','d2:d5281');
c9= xlsread('VAR_41_I.xls','d2:d5281');
c10= xlsread('VAR_42_I.xls','d2:d5281');

expectedClass=xlsread('TestClass.xls');

row = size(expectedClass,1);

PredictedClass =  output(c1,c2,c3,c4,c5,c6,c7,c8,c9,c10,row);
count = 0;
error =0;

for i =1:row
   if (PredictedClass(i)== expectedClass(i))
       count = count+1;
   else
       error = error +1;
   end
end
Recognition_rate = count/row;
Error_rate = 1- Recognition_rate
```

```
function PredictedClass =  output(c1,c2,c3,c4,c5,c6,c7,c8,c9,c10,row)


a=zeros(row,1);

for i=1:row
  if (c1(i)> c2(i) && c1(i)> c3(i)&& c1(i)> c4(i)&& c1(i)> c5(i)&& c1(i)> c6(i)&& c1(i)> c7(i) &&
c1(i)> c8(i)&& c1(i)> c9(i) && c1(i)> c10(i))
     a(i) = 0;
  elseif(c2(i)> c1(i) && c2(i)> c3(i)&& c2(i)> c4(i)&& c2(i)> c5(i)&& c2(i)> c6(i)&& c2(i)>
c7(i) && c2(i)> c8(i)&& c2(i)> c9(i) && c2(i)> c10(i))
     a(i)=1;
  elseif(c3(i)> c1(i) && c3(i)> c2(i)&& c3(i)> c4(i)&& c3(i)> c5(i)&& c3(i)> c6(i)&& c3(i)>
c7(i) && c3(i)> c8(i)&& c3(i)> c9(i) && c3(i)> c10(i))
      a(i)=2;
  elseif(c4(i)> c1(i) && c4(i)> c2(i)&& c4(i)> c3(i)&& c4(i)> c5(i)&& c4(i)> c6(i)&& c4(i)>
c7(i) && c4(i)> c8(i)&& c4(i)> c9(i) && c4(i)> c10(i))
      a(i)=3;
  elseif(c5(i)> c1(i) && c5(i)> c2(i)&& c5(i)> c3(i)&& c5(i)> c4(i)&& c5(i)> c6(i)&& c5(i)>
c7(i) && c5(i)> c8(i)&& c5(i)> c9(i) && c5(i)> c10(i))
      a(i)=4;
  elseif(c6(i)> c1(i) && c6(i)> c2(i)&& c6(i)> c3(i)&& c6(i)> c4(i)&& c6(i)> c5(i)&& c6(i)>
c7(i) && c6(i)> c8(i)&& c6(i)> c9(i) && c6(i)> c10(i))
      a(i)=5;
  elseif(c7(i)> c1(i) && c7(i)> c2(i)&& c7(i)> c3(i)&& c7(i)> c4(i)&& c7(i)> c5(i)&& c7(i)>
c6(i) && c7(i)> c8(i)&& c7(i)> c9(i) && c7(i)> c10(i))
      a(i)=6;
  elseif(c8(i)> c1(i) && c8(i)> c2(i)&& c8(i)> c3(i)&& c8(i)> c4(i)&& c8(i)> c5(i)&& c8(i)>
c6(i) && c8(i)> c7(i)&& c8(i)> c9(i) && c8(i)> c10(i))
      a(i)=7;
  elseif(c9(i)> c1(i) && c9(i)> c2(i)&& c9(i)> c3(i)&& c9(i)> c4(i)&& c9(i)> c5(i)&& c9(i)>
c6(i) && c9(i)> c7(i)&& c9(i)> c8(i) && c9(i)> c10(i))
      a(i)=8;
  else a(i)=9;
  end

end
PredictedClass=a;
```

VITA

Personal Information
Born: 1977, Kano, Nigeria
Nationality: Nigerian
Address: C/O Dr. Abdullahi Muhammad, Aminu Kano teaching hospital, department of paediatrics, PMB 3452, Kano, Nigeria.
Mobile: 0530908337
Email: eesaah@gmail.com

Education

- 2010 - MSc in Computer Engineering, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia
- 2002- BEng in Electrical & Electronic Engineering, Abubakar Tafawa Balewa University, Bauchi, Nigeria

Work Experience

- 2004-2007 Technical Customer Care Officer, Megatech Networks, Kano, Nigeria
- 2007-2010 Research Assistant, KFUPM, Dhahran, Saudi Arabia

Professional Membership
Student member of The Institute of Engineering and Technology