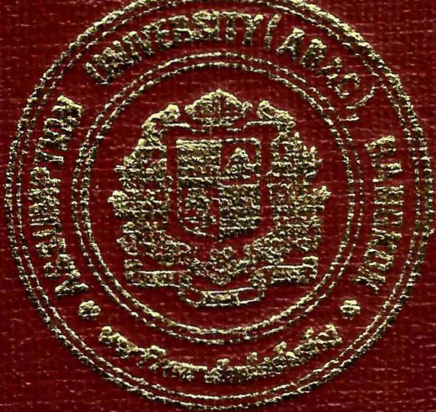

Forward Integrated Feature and Architecture Selection (FIFAS) for OCR Problems using Neural Networks

by

Efrem Dawit

Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science
In Information Technology
Assumption University

November, 2002

Forward Integrated Feature and Architecture Selection (FIFAS) for OCR Problems using Neural Networks

By

Efrem Dawit

Submitted in Partial Fulfillment of the
Requirement for the Degree of
Master of Science
in Information Technology
Assumption University

November 2002

The Faculty of Science and Technology

Thesis Approval

Thesis Title	Forward Integrated Feature and Architecture Selection (FIFAS) for OCR Problems using Neural Networks
By	Mr. Efrem Dawit
Thesis Advisor	Asst.Prof.Dr. Thotsapon Sortrakul
Academic Year	1/2002

The Department of Information Technology, Faculty of Science and Technology of Assumption University has approved this final report of the **twelve** credits course. **IT7000 Master Thesis**, submitted in partial fulfillment of the requirements for the degree of Master of Science in Information Technology.

Approval Committee:

(Asst.Prof.Dr. Thotsapon Sortrakul)
Advisor

(Dr. Naruetep Choakjarernwanit)
Committee Member

(Dr. Jirapun Daengdej)
Committee Member

(Asst.Prof.Dr. Surapong Auwatanamongkol)
Representative of Ministry of
University Affairs

Faculty Approval:

(Asst.Prof.Dr. Thotsapon Sortrakul)
Program Director

(Asst.Prof.Dr. Pratit Santiprabhob)
Dean

October / 2002

Table of Contents

List of Tables

List of Figures

Acknowledgements

I am deeply grateful to my advisor, Asst. Professor Dr. Thotsapon Sortrakul for his encouragement and guidance, and providing the opportunity for conducting this research.

I would also like to extend my thanks to my partner and wife, Weizerite Ghirmai for her restless support, encouragement and prayers. She has devoted herself for the research to make it possible. In other words, without her, this research would have failed. Besides, I would like to give an appreciation to my family for giving me delight, joy and hope.

Above all, behind this research and life at all, I give glory, praises, thanks and adoration to my Savior ***Jesus Christ*** who gives me power, grace, mercy and success in my entire life.

Abstract

Feature selection and model selection are pillars of any classification problems. Bottom-up integrated feature and architecture selection is useful for the optimal neural networks construction for an available training data set. From the algorithm, it is easy to find appropriate architecture for any number of features with acceptable classification rate. Besides, the feature selection approach along with architecture selection gives an advantage of defeating a requirement of a prior knowledge of setting fixed number of features as other researchers did. Furthermore, proposed algorithm gives a chance to decide which pillar comes first for the acceptable solution of the underlie problem. Consequently, it enables practitioners to overcome the investigation of appropriate network topology using trial and error methodology. The proposed algorithm gives us faster, reliable accuracy and less resource usage with likelihood ratio test, cross validation and regularization measures. The other advantage of this algorithm overcomes the burden of computational cost and exhaustive searching for ideal architecture even though it may be not suitable for the proponent of middle ground between accuracy and speed. The algorithm is tested on new benchmark (Geez characters) and common available character recognition feature sets ("0" - "9") handwritten English numerals.

Chapter 1

Introduction

Machine replication of human functions, like reading, is an ancient dream. However, over the last five decades, machine reading has grown from a dream to reality. Optical character recognition has become one of the most successful applications of technology in the field of pattern recognition and artificial intelligence. Character Recognition or Optical Character Recognition (OCR) is the process of converting scanned images of machine printed or handwritten text (numerals, letters, and symbols), into a computer process-able format (such as ASCII or Unicode). Many commercial systems for performing OCR exist for a variety of applications, although the machines are still not able to compete with human reading capabilities. The main principle in automatic recognition of patterns is first to teach the machine which classes of patterns that may occur and what they look like. In OCR patterns are letters, numbers and some special symbols like commas, question marks etc., while the different classes correspond to the different characters. The teaching of the machine is performed by showing the machine examples of characters of all the different classes. Based on these examples the machine builds a prototype or a description of each class of characters. Then, during recognition, the unknown characters are compared to the previously obtained descriptions, and assigned the class that gives the best match.

Since the inception of OCR technologies, there are various and different techniques for the achieving of 100% accuracy rate of the technologies with the compromise of speed and resource usages. Among these technologies, there is new and promising technique, which is Artificial Neural Networks (ANN). It has unique and inherited advantageous for solving of intractable and cumbersome problems such as pattern recognition (Character Recognition), NP problem and etc.

Multilayer Feedforward Neural Network (FNN) is one of numerous variations of neural networks due of architectures and training algorithms. Recently it has been used extensively in Optical Character Recognition. These networks may be viewed as a combined feature extractor and classifier. However in this investigation we deal independently of unified approach in which neural networks for classifier and principal component analysis (PCA) for feature extractor.

ANN has problems which are it is difficult to analyze and fully understand the decision making process. In other words, Artificial Neural Networks in OCR may be their limited predictability and generality while an advantage is their adaptive nature. In addition to this, in order to achieve the highest accuracy for the given problem(s), neural networks should go through exhaustive searching in its weight parameters space and hence it uses a lot of resources and time. The latter problem is addressed in this thesis through the framework of constructive neural networks and maximum likelihood statistic principles. Constructive neural networks are a type of feedforward neural network in which the network architecture is built during the training process. The type of architecture built can affect both generalization and convergence speed.

In the field of automated classification, classifier is not the only pillar of the classification system but also feature selection is the complementary of the classifier. Feature selection is to select a subset of features from large initial dataset that provides the best classification performance. The major advantage of feature selection is not only that it may reduce the cost but it also could remove redundant and noise features. The integration of feature and architecture selection would give the enhanced acceptable accuracy without exhaustive resources. For this particular study, it considers this approach with maximum likelihood statistics to ensure that the proposed algorithm has a unique advantage in OCR technologies.

Convergence speed and reliability are important properties of feedforward neural networks. These properties are studied by probing the combined cause of the inherit benefits of feature selection, likelihood ratio statistics and constructive architecture process in feedforward neural networks. This algorithm addresses the over burden of training loads and hence increases the cost of the design of recognition systems. The proposed algorithm is revealed to achieve the improved classification accuracy with less resource.

The usage of free user biased setting training algorithm (Resilient propagation) with the effect of regularization term is examined through a series of empirical studies on different datasets. These data sets are multi feature handwritten English numerals and new character sets from one of national script languages in East Africa - Geez characters. In addition to this, the feature dimensions of these data sets are high and medium one, respectively. The results of these studies demonstrate that high dimensional feature sets give better classification accuracy compared to medium dimensional feature sets even though the magnitude of deterioration is a bit higher. Moreover, the effect of regularization term between error term and weight parameter shows a better classification accuracy so does the generalization ability of the constructive neural networks with prominent features.

The proposed algorithm - Forward Integrated Feature and Architecture Selection (FIFAS)- has a peculiar advantage for those who in need of less minimal deteriorated accuracy with minimal resources and cost rather than unlikely 100% accuracy feat and huge amount of resources.

1.1 Purpose of the Study

No investigation in character recognition of one of written scripts in the world - Geez scripts has been conducted. The primary purpose of this thesis is to improve the unified effect of feature selection and neural networks algorithm for optical character recognition principles by applying on machine printed Geez characters and trite handwritten English numerals. This thesis is to formulate a new approach to tackle the difficulties of getting enhanced and acceptable solution on character recognition problems using neural networks paradigm and hence verify it on Geez and English characters.

Another objective of this thesis is to establish a foundation for further investigations on recognition of Geez characters.

1.2 Research Questions

In this section, I describe what I see as one central question regarding Froward Integrated Feature and Architecture Selection (FIFAS) for optical character recognition, and which is the goal of this thesis to answer.

Independent investigations of appropriate and relevant features for the "best" solution of given problem have been formulated in literatures. Similarly, suitable topology of neural networks has been investigated for so many problems too. Thus, despite the impressive empirical results from such independent methodologies in the literatures, one central question is that: *which technique(s) is suitable that can improve the principle of neural networks in order to reduce cost function, to get faster speed and to find "enhanced" or "acceptable" accuracy of recognition system using neural networks.* That is, *what will be the unified effect on the independent investigations of feature selection and architecture selection using different approaches for achieving less cost function, faster speed and "acceptable" accuracy?*

Furthermore, the above central question further introduces other central question, that is, *which component is prior of another in the design of the optical character recognition system.* In other words, is there an effect for the choice of order of investigation (selection of feature or selection of architecture) for the improved integrated feature and architecture selection of using neural networks on optical character recognition problems?

Lastly, but not least, which direction should one start the training of neural networks with and/or without feature selection? And *should one proceed constructive (forward) or pruning (backward) training of neural networks with the available features?*

1.3 Scope of the study

In optical character recognition (OCR), there are numerous techniques to solve a specific problem. Due to unexplored issue of Geez characters, this study will focus on only one technique -neural networks. Of course, Optical Character Recognition has various stages or components, which are inseparable from one anthers. Before applying neural networks principles for OCR, OCR may use the following stages in order to get high accuracy performance. These stages are preprocessing, feature extraction, classification and post processing:-

This study focuses on the following stages of OCR:

1. preprocessing the raw data
2. extracting features of each scanned image using feature extraction methods
3. classifying each image using a different choices of neural network principles

Similarly, the study covers scoped to not only Geez characters but also English numeral characters from '0' -'9'. In order to have a broad investigation, the study considered both machine printed and handwritten characters. The former one is Geez characters and the latter one refers to English numerals.

Furthermore, by the investigation of neural networks on the above characters, it is limited to only one kind of feature extractor techniques due to its unsupervised ability for the selection of principal features in line with neural networks principles.

1.4 Limitations

This study has only limited number of character images due to limited available machine printed Geez font types. This may lead to poor performance for the investigation since a large number of training sets are required for high hitting rate for neural network architecture.

Second, there is no comparison with previous studies on Geez character recognition topics since the world volt face on this script.

1.5 Thesis Contribution

This work investigates the application of optical character recognition in the training of multilayer neural networks along with automatic relevant features selection. The combined or integrated feature and architecture selection provides improved solutions and establishes a new algorithm in which it further improves for better solution. By exploring feature selection algorithm, topology selection algorithm, the combined effect of feature and architecture selection, the direction of the starting of the implementation (constructively or destructively) and the effect of dimension (number of training set and features) contribute

1. To the "bridge" forming between the neural network research community and more established disciplines in which neural networks have found application

such as optical character recognition. Specifically, this research provides improving the understanding of potentials and limitations of applying integrated feature and architecture selection;

2. The use of Geez character in the field of IT: People in business and government facilitate their workflow. In other words, OCR systems provide fast storage, recall and distribution of documents in workflow processing and other applications. Document analysis can help with the indexing for storage and recall, and can partition the image into subregions of interest for convenient access by users. Besides, automatic processing of international documents alleviates or ameliorates the challenging problems due to lack of Optical Character Recognition techniques for all available languages and script classes;

3. Lay down for an embarkation of further study on Geez characters or language: Research on automated written language recognition dates back several decades. Today, cleanly printed text in documents with simple layouts can be recognized reliably by off-the-shelf OCR software. There is also some success with handwriting recognition, particularly for isolated hand-printed characters and words, e.g., in the on-line case, the recently introduced personal digital assistants have practical value. Most of the off-line successes have come in constrained domains such as postal addresses [1], bank checks, and census forms. The analysis of documents with complex layouts, recognition of degraded printed text, and the recognition of running handwriting continue to remain largely in the research arena. Some of the major research challenges in recognizing handwriting are in: word and line separation, segmentation of words into characters, recognition of words when lexicons are large and use of language models in aiding preprocessing and recognition. Therefore, hence the core foundation for written language, that is optical character recognition (OCR), gives a way for further studies in the above-mentioned fields;

4. Many commercial systems allow recognition results of Geez character to be placed directly into spreadsheets, databases, and word processors.

1.6 Thesis Overview

The remainder of this thesis is structured as follows: In chapter 2, it has the goal of integrated feature and architecture selection followed by related works on feature selection and constructive neural network design, list out drawbacks of the related methodologies and give a suggestion for solving the drawbacks of discussed methodologies. Chapter 3 then presents the crux aim of this thesis by stating the new algorithm that will answer the research questions that are raised in the previous section. In doing so, we discuss the assumptions, reasons of these assumptions and the superiority of the proposed algorithm. Chapter 4 spells out some of its implementation details and also empirically compare it with related approaches (such as pruning, train without feature selection, and prioritization of the pillars of recognition systems) and test on two different benchmark datasets: common handwritten English numerals and a new machine printed Geez characters. Chapter 5 then summarizes and discusses the experimental results in order to depict the novelty and aptness of the proposed algorithm. Finally, Chapter 6 closes with conclusions, recommendations and future work.

Chapter 2

Literature Review

2.1 Background and Preliminaries

In this chapter, I first discuss what the goal of forward integrated feature and architecture selection (FIFAS) for the application of optical character recognition using neural networks is and then briefly review previous work on constructive neural networks, feature selection algorithm, and backward unified integrated feature and architecture selection.

2.1.1 The Goal of Integrated Feature and Architecture Selection

The notion of independent investigation of feature selection and model (topology) selection in neural networks requires extensive research and enormous cost function to achieve the best and ideal solution in the field of one of pattern recognitions - optical character recognition. Of course, neural networks has an ability to find acceptable solution using inherent integration of feature and model complexity without considering "relevant" feature sets and acceptable topology and in turn cost function. However, the general view of neural networks cannot lead to a better investigation with minimal cost function unless one considers the integrated effect of independent investigation of each pillar of recognition system. On the other hand, combined study of the independent investigation may counter balance one another and hence diminishes the advantages of each component. As a result, we consider a chance to investigate the combined effect of independent investigations of each component without removing the advantages which are inherited in each pillar. Moreover, the integrated approach should use minimal resources so that it improves the speed of the system without deteriorating the accuracy of the recognition rate.

Therefore, this thesis takes the forward integrated feature and model selection using neural networks algorithm in order to minimize the usage of huge resources, improve the speed without compromising the accuracy of the recognition system and to determine the prioritization of pillars.

2.1.2 Related Work

By the time of five years old, most children can recognize digits and letters. Small characters, large characters, handwritten, machine printed, or rotated - all are easily recognized by the young. The characters may be written on a cluttered background, on crumpled paper or even be partially occluded. We take this ability for granted until we face task of teaching a machine how to do the same. Pattern recognition is the study of how machines can observe the environment, learn to distinguish patterns of interest from their background, and make sound and reasonable decisions about the categories of the patterns. In spite of almost 50 years of research, design of a general-purpose machine pattern recognizer remains an elusive goal.

The best pattern recognizers in most instances are humans, yet we do not understand how human recognizes pattern. Rose [2] emphasizes the work of Nobel Laureate Herbert Simon whose central finding was that pattern recognition is critical in most human decision making tasks: "The more relevant patterns at your disposal, the better your decisions will be. This is hopeful news to proponents of artificial intelligence, since computer can surely be taught to recognize patterns. Indeed, successful computer programs that help banks score credit applicants, help doctors diagnose disease and help pilot land airplanes depend in the same way on patter recognition. We need to pay much more explicit attention to teaching pattern recognition."

What is Pattern Recognition? Automatic (machine) recognition, description, classification, and grouping of patterns are important problems in a variety of engineering and scientific disciplines such as biology, psychology, medicine, marketing, computer vision, artificial intelligence, and remote sensing. As Watanable [3] defines a pattern "as opposite of a chaos; it is an entity, vaguely defined, that could be given a name." For example, pattern could be a fingerprint image, a handwritten cursive word, a human face, or a speech signal. Given a pattern, its recognition/classification may consist of one of the following two tasks [3]:

1. supervised classification (e.g., discriminant analysis) in which the input pattern is identified as a member of a predefined class,
2. unsupervised classification (e.g., clustering) in which the pattern is assigned to hitherto unknown class.

Note that the recognition problem here is being posed as a classification or categorization task, where the classes are either defined by the same designer (in supervised classification) or are learned based on similarity of patterns (in unsupervised classification).

Interest in the area of pattern recognition has been renewed recently due to emerging applications which are not only challenging but also computationally more demanding (see Table 2.1). These applications include data mining (identifying a "pattern" e.g., correlation, or an outlier in millions of multidimensional patterns), document classification (efficiently searching text documents), financial forecasting,

Table 2.1: Example of pattern recognition application

Problem Domain	Application	Input Pattern	Pattern Classes
Biometrics	Sequence analysis	DNA or Protein sequence	Known types of genes or pattern
Data mining	Searching for meaningful patterns	Point of multidimensional space	Compact and well separated clusters
Document classification	Internet search	Text document	Semantic categories
Document image analysis	**Reading machine for blind**	**Document image**	**Alphanumeric characters, words**
Industrial automation	Printed circuit board inspection	Intensity or range image	Defective/non-defective nature of product
Multimedia database retrieval	Internet search	Video clip	Video genre
Biometric recognition	Personal identification	Face, iris, fingerprint	Authorized users for access control
Remote sensing	Forecasting crop yield	Multispecteral image	Land use categories, growth pattern of crops
Speech recognition	Telephone directory enquiry	Speech waveforms	Spoken words

Adapted from [4], pp 5.

organization and retrieval of multimedia database, and biometrics (personal identification based on various physical attributes such as face and fingerprints). A common characteristic of a number of these applications is that the available features (typically, in the thousands) are not usually suggested by domain experts, but must be extracted and optimized by data-driven procedures.

The rapidly growing and available computing power, while enabling faster processing of huge data sets, has also facilitated the use of elaborate and diverse methods for data analysis and classification. At the same time, demands on automatic pattern recognition systems are rising enormously due to the availability of large databases and stringent performance requirements (speed, accuracy, and cost). In many of emerging applications, it is clear that no single approach for classification is "best" and that multiple methods and approaches have to be used. Consequently, combining several sensing modalities and classifiers is now a common used practice in pattern recognition.

The design of a pattern recognition system essentially involve the following three aspects:

1. data acquisition and preprocessing,
2. data representation, and
3. decision making.

The problem domain dictates the choice of sensor(s), preprocessing technique, representation scheme, and the decision making model. It is generally agreed that a well-defined and sufficiently constrained recognition problem (small intraclass variations and large interclass variations) will lead to a compact pattern representation and a simple decision making strategy. Learning from a set of examples (training set) is an important and desired attribute of most pattern recognition systems. The four best-known approaches from pattern recognition are:

1. Template matching,
2. Statistical classification,
3. Syntactic or structural matching, and
4. Neural networks

These models are not necessarily independent and sometimes the same pattern recognition method exists with different interpretations. Attempts have been made to design hybrid systems involving multiple models [5].

In this paper, the crux idea is Artificial Neural Networks (ANN). It is discussed in detail along with other related approaches.

For many centuries, one of the goals of human kind has been to develop machines. One envisioned these machines as performing all cumbersome and tedious tasks so that one might enjoy a more fruitful life. The era of machine making began with the discovery of simple machines such as lever, wheel and pulley. Many equally congenial inventions followed thereafter. Nowadays engineers and scientists are trying to develop intelligent machines. Artificial neural systems are present-day examples of such machines that have great potential to further improve the quality of our life. Artificial Neural Network can be defined as follows:

> *A structure (network) composed of a number of interconnected units (artificial neurons). Each unit has an input/output (I/O) characteristic and implements a local computation or function. The output of any unit is determined by its I/O characteristic, its interconnection to other units, and (possibly) external inputs. Although "hand crafting" of the network is possible, the network usually develops and overall functionality through one or more forms of training. [6]*

Artificial neural networks represent the promising new generation of information processing networks. Advances have been made in applying such systems for problems found intractable or difficult for traditional computation. Artificial neural systems function as parallel distributed computing networks. Their most basic characteristic is their architecture. Only some of the networks provide instantaneous responses. Other networks need time to respond and are characterized by their time-domain behavior, which one often refers to as dynamics. Neural networks also differ from each other in their learning modes. There are a variety of learning rules that establish when and how the connecting weights change. Finally, networks exhibit different speeds and efficiency of learning. As a result, they also differ in their ability to accurately respond to the cues presented at the input.

In contrast to conventional computers, which are programmed to perform specific tasks, most neural must be taught, or trained. They learn new associations, new patterns, and new functional dependencies. Learning corresponds to parameter changes. Learning rules and algorithms used for experimental training of networks replace the programming required for conventional computation. Neural network users do not specify an algorithm to be executed by each computing node, as would programmers of a more traditional machine. Instead, they select what in their view is the best architecture, specify the characteristics of the neurons and initial weights, and choose the training mode for the network. Appropriate inputs are then applied to the network so that it can acquire knowledge from the environment. As a result of such exposure, the network assimilates the information that can later be recalled by the user.

In general, neural networks can be viewed as massively parallel computing systems consisting of an extremely large number of simple processors with many interconnections. Neural Network models attempt to use some organizational principles (such as learning, generalization, adaptivity, fault tolerance and distributed representation, and computation) in network of weighted directed graphs in which the nodes are artificial neurons and directed edges (with weights) are connections between neuron outputs and neurons inputs. The main characteristics of neural networks are that they have the ability to learn complex nonlinear input-output relationships, use sequential training procedures, and adapt themselves to the data.

The most commonly used family of neural networks for pattern classification tasks [7] is feed-forward network, which includes Multilayer Percepton (MP) and Radial-Basis Function (RBF) networks. These networks are organized into layers and have unidirectional connection between the layers. Another popular network is Self-Organizing Map (SOM), or Kohonen-Network [8], which is mainly used for data clustering and feature mapping. The learning process involves updating network architecture and connection weights so that a network can efficiently perform a specific classification/clustering task. The increasing popularity of neural network models to solve pattern recognition problems has been primarily due to their seemingly low dependence on domain-specific knowledge (relative to model-based and rule-based approaches) and due to the availability of efficient learning algorithms for practitioners to use.

ANNs provide a new suite of nonlinear algorithms for feature extraction (using hidden layers) and classification (e.g., Multilayer Perceptrons). In addition, existing feature extraction and classification algorithm can also be mapped on neural network architectures for efficient (hardware) implementation. In spite of the well-known neural network models are implicitly equivalent or similar to classical statistical pattern recognition methods, Neural networks do offer several advantages such as, unified approaches for feature extraction and classification and flexible procedure for finding good, moderately nonlinear solutions. However, this study tries to investigate independent approach for feature selection and classification of the underline problem. Consequently, the results will compare both approaches - unified and independent.

The following are key aspects of neural computing:-

- As the definition of ANN indicates, the overall computational model consists of a re-configurable interconnection of simple elements, or units;
- Individual units implement a local function, and the overall network of interconnected units displays a corresponding functionality. Analysis of this functionality, except through training and test examples, is often difficult. Moreover, the application usually determines, via specifications, the required functionality;
- Modifying patterns of inter-element connectivity as a function of training data is a key learning approach. In other words, the system knowledge, experience, or training is stored in the form of network interconnections;
- To be useful, neural systems must be capable of storing information (i.e., they must be "trainable"). Neural systems are trained in the hope that they will subsequently display correct associative behavior when presented with new patterns

to recognize or classify. That is, the objective in the training process is for the network to develop an internal structure enabling it to correctly identify or classify new, similar patterns and

- A neural network is a dynamic system; its state (e.g., unit outputs and interconnection strengths) changes over time in response to external inputs or an initial (unstable state).

2.1.3 Advantages and Disadvantages of ANNs

Because ANNs are a relative new computational paradigm, it is probably safe to say that the advantages, disadvantages, applications, and relationships to traditional computing are not fully understood. Expectations (some might say "hype") for this are high. Neural networks are particularly well suited for certain applications, especially trainable pattern association. The notion that artificial neural networks can solve all problems in automated reasoning, or even all-mapping problems, is probably unrealistic.

Advantages

- Inherently massive parallel
- May be fault tolerant because of parallelism
- May be designed to be adaptive
- Little need for extensive characterization of problem (other than through the training set)

Disadvantages

- No clear rules or design guidelines for arbitrary application
- No general way to asses the internal operation of the network

- Training may be difficult or impossible
- Difficult to predict future network performance (generalization)

2.1.4 The Curse of Dimensionality and Peaking Phenomena

The performance of a classifier depends on the interrelation between sample size, number of features, and classifier complexity. A naive table-lookup technique (partitioning the feature space into cells and associating a class label with each cell) requires the number of training data points to be an exponential function of the feature dimension [9]. This phenomenon is termed as "curse of dimensionality," which leads to the "peaking phenomenon" in classifier design. It is well known that the probability of misclassification of a decision rule doesn't increase as the number of features increases, as long as the class-conditional densities are completely known (or, equivalently, the number of training samples is arbitrarily large and representative of the underlying densities). However, it has been often observed in practice that the added features may actually degrade the performance of a classifier if the number of training samples that are used to design the classifier is small relative to the number of features. This paradoxical behavior is referred to as the peaking phenomenon [10][11] [12]. A simple explanation for this phenomenon is as follows: The most commonly used parametric classifier estimates the unknown parameters and plugs them in for the true parameters in class-conditional densities. For a fixed sample size, as the number of features is increased (with a corresponding increase in the number of unknown parameters), the reliability of the parameter estimates decreases. Consequently, the performance of the resulting plug-in classifiers, for a fixed sample size, may degrade with an increase in the number of feature.

The practical implication of the curse of dimensionality is that a system designer should try to select only a small number of salient features when confronted with a limited training set. All of the commonly used classifier, including multilayer feed forward networks, can suffer from the curse of dimensionality. While an exact relationship between the probability of misclassification, the number of training samples, the number of misclassification, the number of features and the true parameters of the class-conditional densities is very difficult to establish, some guideline have been suggested regarding the ratio of the sample size to dimensionality. It is generally accepted that using at least ten times as many training samples per class as the number ($\frac{n}{d} > 10$) is a good practice to follow in classifier design [10]. The more complex the classifier, the larger should the ratio of sample size to dimensionality be to avoid the curse of dimensionality.

2.1.5 Dimensionality Reduction

There are two main reasons to keep the dimensionality of the pattern representation (i.e., the number of features) as small as possible: measurement cost and classification accuracy. A limited yet salient feature set simplifies both the pattern representation and the classifiers that built on the selected representation. Consequently, the resulting classifier will be faster and will use less memory. Moreover, as stated earlier, a small number of features can alleviate the curse of dimensionality when the number of training samples is limited. On the other hand, a reduction in the number of features may lead to loss in the discrimination power and thereby lower the accuracy of the resulting recognition system. Watanabe's ugly duckling theorem [3] also supports the need for a careful choice of the features, since it is possible to make arbitrary patterns similar by encoding them with a sufficiently large number of redundant features. It is important to make a distinction between feature selection and feature extraction. The term feature election refers to algorithms that select the (hopefully) best subset of the input feature set. Methods that create new features based on the transformations or combinations of the original feature set are called feature selection algorithms. However, the terms feature selection and feature extraction are used interchangeably in the literatures. Note that often feature extraction precedes feature selection; first, features are extracted from the sensed data (see Table 2.2) and then some of the extracted features with low discrimination ability (e.g., using Principal Component or Discriminant Analysis) are discarded. The choice between feature selection and feature extraction depends on the application domain and the specific training data which are available. Feature selection leads to savings in measurement cost (since some of the features are discarded) and the selected features retain their original physical interpretation.

The main issue in dimensionality reduction is the choice of a criterion function. A commonly used criterion is the classification error of a feature subset. But the classification error itself cannot be reliably estimated when the ratio of sample size to the number of features is small.

Feature extraction methods determine an appropriate subspace of dimensionality m (either in a linear or a nonlinear way) in the original feature space of dimensionality d ($m \leqslant d$). Linear transforms, such as principal component analysis, factor analysis, linear discriminant analysis, and project pursuit have been widely used in pattern recognition for feature extraction and dimensionality reduction. The best known linear feature extractor is the principal component analysis (PCA).

In the investigation of all-rounded solution for real-world problems using neural networks, there are many questions to be answered. One such open question involves determining the most appropriate network size (architecture) for solving a specific task. In fact both large and small networks exhibit a number of advantages. When a network has too many free parameters (i.e., weights and/or units) not only is learning fast but local minima are more easily avoided. Large networks can also form as complex decision regions as the problem requires and should exhibit a certain degree of fault tolerance under damage conditions. On the other hand, both theory

[13] and experience [14]-[16] show that networks with few free parameters exhibit a better generalization performance. Moreover, knowledge embedded in small trained networks is presumably easier to interpret and thus the extraction of simple rules can be hopefully be facilitated [17]. Lastly, from an implementation standpoint, small networks only require limited resources in any physical computational environment.

2.2 Architectural Selection

In the design of neural networks, one should ponder the strenuous selection of architecture of neural networks due to difficulty and time-consuming job. For neural networks, it is equally important to find an optimal network topology, as it is to determine an optimal set of weights. It has many factors to acquire the "best" architecture for "best" solution of the underline problem. Among the factors, number of hidden layers and neurons on it are major factors to decide the architecture of neural networks. The use of dynamic neural network algorithms considerably speeds up the process of finding an appropriate network topology for a given problem. The following sections will discuss important aspects of the selection of architecture of neural networks.

2.2.1 Model Selection and Generalization

A key difficulty faced in the field of Feedforward Neural Network (FNN) is model selection. Model selection involves matching the complexity of the function to be approximated with the complexity of the model. FNN model complexity is determined by factors such as weight number, magnitude and connection topology. If a model does not have the complexity to approximate the desired function, underfitting and poor generalization occur. If a model is too complex, then it may overfit the data and also give poor generalization.

It is possible to classify FNN model selection techniques into three groups:

1. Those that perform a search through models,
2. Those that begin with an overly complex model which is then simplified, and
3. Those that begin with simple model whose complexity is increased

The first group is generally implemented by selecting various network architectures that are trained and compared. One well-respected method of comparison is cross-validation [18]. This is often computationally expensive to perform due to nonlinear optimization process employed in the training FNN's, although there have been some attempts to reduce this expense [18].

The second group of model selection techniques begins training with an oversize network. Pruning is one such technique [19]. The non-convergent technique of early stopping [20] also uses an oversize network, and works by stopping training at a point where the performance on a validation set begins to worsen. The performance on the validation set will typically start to worsen when an overly complex model starts to form. This occurs because the validation set is sampled from the underlying function of which a model is sought. Since the validation set is representative of the underlying function, the performance on this data set will worsen when a model more complex than the underlying function is formed. There are situations, however, where, it can be envisaged that early stopping will fail to produce good generalization. One situation where an overly complex model may be produced using early stopping is through the formation of an overly complex model during the training process. In this case it is still possible that the validation error will fall, since overall the model moves closer to the validation data. As training continues the overly complex model still reduces the validation error as it learns the training data. As training continues, eventually the validation error will stop falling, and may even begin to rise. At this point, however, an overly complex model already existed and there is no way for early stopping to reduce the complexity.

One way of reducing the chance that an overly complex model is formed during the training process is to use regularization to constraint the network weights in some manner. Since regularization is present throughout the training process, the chance of producing over-complexity is reduced (assuming an appropriate level of regularization is used). A regularization function commonly used in FNN's is the sum of squares of the weights magnitudes. Additional forms of regularization that have been applied to FNN's include the Weigend-Rumelhart regularization term [21] and curvature driven regularizers [9].

The main disadvantage of regularization is the difficulty in selecting the appropriate magnitude for a given problem. One method used to set the regularization level is to train a given FNN a number of times with each training runs using a different regularization magnitude. A technique such as cross-validation can be used to compare the trained FNN's [9]. Bayesian methods are another technique used to set the regularization magnitude [22][23]. A simple method introduced in [21] dynamically sets the regularization magnitude during training based on the error performance of the FNN.

The third group of model selection techniques uses constructive methods [24]. These methods consist of starting with a minimal size network (often with no hidden neurons), and sequentially adding weights according to some criterion. A number of network construction algorithms have been developed as far: Upstart algorithm [25], Add and Remove [26], and Cascade-Correlation [27], FlexNet [28], to name a few. However, the drawback with these procedures is that they either have been designed for the use of binary neurons [25] or underlie constraints such as a limited number of layers and hidden units [26][27] or only deal with fixed inputs.

2.2.2 Constructive Algorithms

There are a number of inherent advantages in using constructive algorithms that begin training with oversized network:-

1. It is often difficult to specify what size network can actually be considered an oversize network a priori. If initial network selected is too small it will be unable to converge to a good solution and hence underfit the data. On the other hand, selecting an initial network that is much larger than required makes training computationally expensive. Constructive algorithms however, initially select a minimal size network and can increase the network complexity until an appropriate level is reached. In addition, constructive algorithms will spend the majority of their time training networks smaller than the final network, as compared to algorithms that starting training with an oversize network.
2. The problem of encouraging poorly performing local minimal is avoided. The majority of FNN training algorithms is based on gradient methods, and hence can be trapped in local minima. Constructive algorithms are able to escape from a local minimal when more weights are added to the network. This can occur because the addition of more weights increases the dimensionality of the error surface and may allow the network to continue reducing the error level.

There are a number of properties that must be defined for constructive algorithms. These include the type of training algorithm, establishing how the new hidden neurons is connected to the current network, defining which weights are to be updated and in which order, deciding on criteria to determine when a new hidden neuron is added, deciding on criteria to halt network construction, and selecting how much regularization to use, if any.

In general, constructive networks use gradient-based training algorithms due to their convergence speed [29]-[31]. A number of ways of connecting a new hidden neuron to the network are existed. The two common methods are to construct a single layer of hidden neurons [29] [31][26][32] or to create a cascade of hidden neurons [27][33]. In cascade architecture each new hidden neuron receives inputs from all inputs and previously installed hidden neurons. Since the hidden neurons in cascade architecture receive additional information from some nonlinear combination of the inputs (implemented by previous hidden neurons), these neurons are termed higher-order neurons and are capable of performing a more complex function of the input variables. Cascade networks, while having more representational power, are more likely to overfit the data.

There are a variety of ways of training the new hidden neurons in constructive algorithms. These can be classified into two general methods. The first consists of training the whole network after the addition of a new hidden neuron [29], [31], [26] and [32]. The second entails only training a subset of weights, with the remaining weights being "frozen" [27][34]. The advantage of this second greedy strategy is that there are far few weights to optimize than when all the weights are trained. The disadvantage of weight freezing is that each optimization phase is unlikely to be optimal, and this can result in larger networks than those in which all the weights are optimized [35].

The method for adding a new neuron is standard across many constructive algorithms and in general consists of either adding a new neuron when the error fails to fall by a set of amount over a given period [29][27][26] or testing for some criterion such as a local minimum [31]. Halting network construction is equivalent to finding the best model for a given problem, and hence techniques such as early stopping can be employed [36].

The ability to control the complexity of the new hidden neuron is an important issue for constructive networks in terms of convergence speed and generalization. In cascade networks the hidden neurons become more powerful higher-order neurons as the network grows. A similar effect occurs for ridge polynomial network [40] which installs ever more complex higher-order neurons. Having neurons of too little complexity may slow convergence, while having neurons of too high complexity can cause poor generalization. A number of approaches have been taken to control hidden neuron complexity [37], or through the use of regularization. Bayesian methods have been used to automatically set regularization magnitudes in a constructive algorithm framework [38].

Dynamic network creation (DNC) algorithm proposed by Ash [29] is relatively simple algorithm that uses backpropagation (BP) [39]. In DNC a sigmoid hidden neuron is added to the single layer of hidden neurons after a period of training when the error has stopped decreasing by a given amount. After the addition of the new hidden neuron, the whole network is again trained with BP. This algorithm a number of advantages: its strong convergence follows directly from its universal approximation ability [24], and it has been shown to perform well on some simple classification tasks. One criticism that has been made of this type of constructive algorithm is that it does not scale up well [24] since all the network weights need to be trained after the addition of a new hidden neuron. This may be a problem for complex modeling requiring large networks.

2.2.3 Regularization

One method that would allow the amount of regularization to be automatically selected in constructive algorithm is to adapt this parameter as the network is constructed [41][42]. The adaptation process works by noting the validation results on a range of regularization levels. When a new neuron is added to the network, the range of regularization levels used for the new network is modified based on the previous validation results. Larger networks allow greater time for the adaptation process to refine the regularization level. In other words, network size and presence of noise can adapt the regularization level well. The regularization magnitudes selected for the noisy data set become greater as training proceeds, and are successful in preventing the network overfitting the data.

Among the most common used regularization terms, there are three regularization terms used in training neural networks:

$$\lambda S w_{ij}^2 \tag{2.2.1}$$

where $S = 2^{\frac{-Epoch}{T}}$ is a simulated annealing (SA) term, T is the temperature constant, and Epoch is the number of epochs used by the training algorithm.

$$\lambda S w_{ij}^2 sign(w_{ij}) \tag{2.2.2}$$

From [43]

$$\frac{1}{n}\sum_{j=1}^{n} {w^2}_j \tag{2.2.3}$$

The adaptive regularization technique has the advantage of being applicable to the general class of constructive algorithms. The technique does not use any training algorithm specific information, but only requires that a number of training runs be performed at each stage of network construction. It would be an easy matter to incorporate adaptive regularization in such constructive FNN's as Ash's DNC [29] and Fahlman's cascor algorithms [27].

The main disadvantage of the adaptive regularization method is the increase in computational cost. The increase in computational cost scales approximately linearly in comparison to corresponding size networks trained by an algorithm used a regularization of section 2.2.2. This is expected, since it is a result of at most three additional training stages at each point of network construction.

One difficulty introduced with regularization is the selection of the magnitude to be used. Varying this magnitude can affect generalization, with the optimal value depending on the function to be modelled, the amount of the noise in the data set, and the final size of the network. This difficulty not withstanding, casper produces good generalization over a number of regularization magnitudes. An explanation for this is that since the training algorithm used is constructive, the network can compensate for a range of regularization magnitudes by increasing network complexity through the addition of further hidden neurons. Since early stopping is also employed, network construction is halted when the complexity reaches the level necessary to produce a good model.

Second reason for poor generalization of cascaded correlation was pointed out in [44], which noted that the correlation measure in cascaded correlation forces the hidden neurons to saturate in order to obtain a large correlation value. This forces the hidden neurons to have large input weights. The effect of large weights in the network is to make the outputs jagged in appearance. This may not degrade results for some classification problems which require a definite boundary between classes. Regression problems, however, generally require the fitting of smooth functions, and so producing jagged outputs can result in poor generalization.

A series of empirical studies were performed to examine the effect of regularization on generalization in constructive cascade algorithms [45]. It was found that the combination of early stopping and regularization resulted in better generalization than the use of early stopping alone. A cubic penalty term that greatly penalizes large weights was shown to be beneficial for generalization in cascade networks. An adaptive method of setting the regularization magnitude in constructive algorithms was introduced and shown to produce generalization results similar to those obtained with a fixed, user-optimized regularization setting. This adaptive method also resulted in the construction of smaller networks for complex problems.

The major advantage of adaptive casper algorithm is that it performs automatic model selection through automatic network construction and regularization. This removes the need for the user to select these parameters, and in the process makes the adaptive casper algorithm free of parameters which must be optimized prior to the commencement of training.

There is additional other approach which is called FlexNet algorithm [28]. It created networks with as many layers and as many hidden units as are needed to solve a given problem. In addition, the user is able to choose between different connection strategies and has the option of freezing weights. However all the above mentioned algorithms are dealt with fixed inputs, they use dynamic architectural creation or selection.

The philosophy of FlexNet is based on strategies used in Cascade-Correlation (cascor): it starts with only the input and output layers and incrementally builds up a complex network architecture by first training candidate neurons and then installing the best ones. However, as the name implies, FlexNet is a highly flexible and powerful network construction algorithm, which is not limited by constraints such as cascor's one-neuron layers and deeply cascaded structure.

The main aspect of FlexNet can be summarized as follows:

- Variable number of hidden layers and units in these layers
- Variable cross-cut connections,
- Variable candidate pool training,
- Possibility of freezing weights.

Similar to the cascor procedure, FlexNet consists of two training phases: a main and a candidate training phases. In the main training phase, the current network is trained until a satisfactory performance is obtained or error stagnation is observed. In the latter case, the algorithm switches to the candidate training: Candidate units are trained separately at different positions in the hidden layers. The best candidate, i.e., the one that contributes to the highest error reduction rate, is permanently installed in the network. Cascor does not consider the benefits of installing new units in existing hidden layers. Instead, after each candidate training phase, it creates a new one-neuron hidden layer resulting in deeply cascade networks with poor generalization ability. FlexNet, on the other hand, allows multiple units per hidden layer. Further more, FlexNet does not necessarily install candidates in the newly created layer, but also checks candidates' performances in exiting hidden layers.

In addition, FlexNet does not only train individual candidate (as CasCor does) but is able to train and install sets of several candidates, which has positive effects on both convergence speed and generalization. By training not only one set of candidates, but also a pool of several sets of candidates, the chances to install weak candidate units decrease and weight space is searched more effectively.

Unlike other network construction methods, no constraints are imposed on the FlexNet procedure regarding the number of hidden layers and hidden units in these layers. Features such as different connection strategies, candidate pool training, and the option of freezing weights enables FlexNet to build optimal networks for a given problem. The drawback of this algorithm is it has not used cross-validation and early stopping principle to avoid overfitting and it might further deteriorat the accuracy of the overall of networks.

2.2.4 The Likelihood-Ratio Test

Several alternative models are often proposed to explain the same data, and objective criteria are needed to choose among models. The alternative models may be nested or non-nested. Nested models are constructed such that a simpler model can be obtained from a more complex model by eliminating one or more parameters from the more complex model. Thus choosing among models reduces to determining the appropriateness of the additional parameters. While adding features to a model is often desirable, the increased complexity comes with a cost. In general, the more parameters contained in a model, the less reliable are parameter estimates. Criteria to select among models must weigh the trade-off between increased information and decreased reliability. Beginning with the simplest case, a null model $f(x, \theta_0)$, specified by the parameter vector $\theta_0 = (\theta_1, \theta_2, \ldots, \theta_k)$, is compared to an alternative model $f(x, \theta_A)$, which shares the k parameters of the null model but also contains an additional parameter,θ_{k+1} . In comparing the null to the alternative hypothesis, we are determining the appropriateness of adding the additional parameter to the null model. In other words, we are testing the following hypotheses:

H_0: $\theta_k = 0$, versus H_A:$\theta_{k+1} \neq 0$.

In the determining of variables to be removed from the mode, the **likelihood-ratio (LR) test** is one of the techniques to remove the variables. It is a better criterion than the Wald statistic. In likelihood process, it involves estimating the model with each variable eliminated in turn and looking at the change in the log likelihood when each variable is deleted. The likelihood-ratio test for the null hypothesis that the coefficients of the terms removed are zero is obtained by dividing the likelihood for the reduced model by the likelihood for the full model. In other words, the likelihood ratio test is aimed at testing a simple null hypothesis against a simple alternative hypothesis. If the null hypothesis is true and sample size is sufficiently large, the quantity -2 times the log of the likelihood-ratio statistic has a chi-square distribution with r degrees of freedom, where r is the difference between the number of terms in the full model and the reduced model. (The model chi-square is likelihood-ratio test.) When the likelihood-ratio test is used for removing terms from a model, its significance level is compared to the cutoff value.

2.3 Feature Extraction and Selection

Devijver and Kittler define feature extraction [46] as *the problem of extracting from the raw data the information which is most relevant for classification purposes, in the sense of minimizing the within-class pattern variability while enhancing the between-class pattern variability.* It should be clear that different feature extraction methods fulfill this requirement to a varying degree, depending on the specific recognition problem and available data. A feature extraction method that proves to be successful in one application domain may turn out not to be very useful in another domain.

The purpose of feature extraction is to reduce data by measuring certain "features" or "properties" that distinguish input patterns. In feature extraction, one transforms an input observation vector to a feature vector using some orthogonal or non-orthogonal basis functions so that data in the feature space are uncorrelated. One could argue that there is only a limited number of independent features that can be extracted from a character image, so that which set of features is used is not so important. However, the extracted distortions and variations of characters may have in a specific application. Also the phenomenon called the curse of dimensionality cautions us that with limited training set, the number of features must be kept reasonably small if a statistical classifier is to be used. A rule of thumb is to use five to ten times as many training pattern of each class as the dimensionality of the feature vector. In practice, the requirements of a good feature extraction method makes selection of the best method for a given application a challenging task. One must also consider whether the characters to be recognized have been known orientation and size, whether they are handwritten, machine printed or typed, and to what degree they are degraded. Also more than one pattern class may be written in two or more distinct ways.

Feature extraction is an important step in achieving good performance of OCR systems. However, the other steps in the system also need to be optimized to obtain

Table 2.2: Overview of Feature Extraction Methods for various Representation Forms -gray level, binary and vector.

Gray scale Subimage	Binary		Vector(Skeleton)
	Solid character	Outer contour	
Template Matching	Template matching		Template matching
Deformable templates			Deformable template
Unitary Transforms	Unitary transform		Graph description
	Projection Histograms	Contour profiles	Discrete features
Zoning	Zoning	Zoning	Zoning
Geometric moments	Geometric moments	Spline curve	
Zernike moments	Zernike moments	Fourier descriptors	Fourier descriptors

the best possible performance, and these steps are not independent. The choice of feature extraction method limits or dictates the nature and output of the preprocessing step in Table 2.2. Some of extraction methods work on gray level subimages of single characters, while others work on solid 4-connected or 8-connected symbols segmented from the binary raster image, thinned symbols or skeletons, or symbol contours. Further, the type or format of the extracted features must match the requirements of the chosen classifier.

In order to recognize many variations of the same character, features that are invariant to certain transformations on the character need to be used. Invariants are features which have approximately the same values for samples of the same character that are translated, scaled, rotated, stretched, skewed or mirrored. However, not all variations among characters from the same character class (e.g., noise or degradation, and absence or presence of serifs) can be modeled by using invariants.

Size and translation invariance is easily achieved. The segmentation of individual characters can itself provide estimates of size and location, but the feature extraction method may often provide more accurate estimates.

Rotation invariance is important if the characters to be recognized and can occur in any orientation. However, if all the characters are expected to have the same rotation, then rotation-invariant features should be used to distinguish between such characters as '6' and '9', and 'n' and 'u'. Another alternative is to use orientation-invariant features, augmented with the detected rotation angle. If the rotation angle is restricted, say, to lie between clockwise 45 degree and 45 degree anticlockwise, characters that are, say 180 rotations of each other can be differentiated. The same principle may be used for size-invariant features, if one wants to recognize punctuation marks in addition to characters, and wants to distinguish between, say, '.', 'o' and 'O'; and ',' and '9'.

Skew-invariance may be useful for hand-printed text, where the characters may be more or less slanted, and multifont machine printed text, where some fonts are slanted and some are not. Invariance to mirror images is not desirable in character recognition, as the mirror image of a character may produce an illegitimate symbol or a different character.

If invariant features cannot be found, an alternative is to normalize the input images to have standard size, rotation, contrast, and so on. However, one should keep in mind that this introduces new discretization errors.

For some feature extraction methods, the characters can be reconstructed from the extracted features [47] and [48]. This property ensures that complete information about the character shape is present in the extracted features. Although, for some methods, exact reconstruction may require an arbitrarily large number of features, reasonable approximations of the original character shape can usually be obtained by using only a small number of features with the highest information content. The hope is that these features also have high discrimination power.

By reconstructing the character images from the extracted features, one may visually check that a sufficient number of features are used to capture the essential structure of the characters. Reconstruction may also be used to informally control that the implementation is correct.

2.3.1 Principal Component Analysis (PCA)

Principal Component Analysis, or PCA is one of feature selection algorithm and widely used in signal processing, statistics, and neural computing. In some application areas, this is also called the (discrete) Karhunen-Loève transform, or the Hotelling transform. PCA has been widely used in data analysis and compression. Principal component analysis (PCA) involves a mathematical procedure that transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called principal components. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible.

Traditionally, principal component analysis is performed on a square symmetric matrix of type SSCP (pure sums of squares and cross products), Covariance (scaled sums of squares and cross products), or, Correlation (sums of squares and cross products from standardized data). The analysis results for objects of type SSCP and Covariance do not differ. A Correlation object has to be used if the variances of individual variants differ much, or the units of measurement of the individual variants differ. The result of a principal component analysis on such objects will be a new object of type PCA.

We are here faced with contradictory goals: On one hand, we should simplify the problem by reducing the dimension of the representation. On the other hand we want to preserve as much as possible of the original information content. PCA offers a convenient way to control the trade-off between losing information and simplifying the problem at hand.

Briefly, the objectives of PCA in regards to feature data set are:

- To discover or to reduce the dimensionality of the data set; and
- To identify new meaningful underlying variables which are not correlated.

The mathematical technique used in PCA is called eigen analysis: we solve for the eigenvalues and eigenvectors of a square symmetric matrix with sums of squares and cross products. The eigenvector associated with the largest eigenvalue has the same direction as the first principal component. The eigenvector associated with the second largest eigenvalue determines the direction of the second principal component. The sum of the eigenvalues equals the trace of the square matrix and the maximum number of eigenvectors equals the number of rows (or columns) of this matrix.

2.4 Backward Unified Feature and Architecture Selection

Neural network feature selection is determining an appropriate feature subset from a set of candidate features, whereas architecture selection is determining an appropriate number of hidden layer nodes. In the past, the related problems of feature selection and architecture determination have been independently considered for neural networks [26][49][50][51][52][53][54][55][56][57] [58][59][60],[61][62]. An integrated approach for feature and architecture selection is meaningful since accurate neural network prediction is inherently related to both feature and architecture complexity.

The work of Jean M Steppe used back propagation to train a single output neural network with one hidden layer and one-sided (positive only) sigmoids on both the hidden and output layers [63]. The methodology presented in their work did not depend on the type of sigmoid function employed. Furthermore, this methodology could be employed in a multilayer network, but architecture selection would be become very complicated. The works of Cybenko [64], Hornik *et al.* [65], and Hecht-Nielsen [66] suggest that the use of single hidden layer is fully justifiable in view of the network's ability to accurately approximate arbitrary functions, provided a sufficient number of hidden nodes.

In integrated feature and architectural methodology some authors use the combination of statistical model and backward sequential selection [63]. The backward selection refers to starting the selection process with a large model and iteratively considering reduced models. For situations where correlated features are present, practical experience in linear regression applications indicates that a potentially better feature set may be selected via backward selection [67].

In neural networks algorithm, it often requires multiple neural networks training with different random initialization of the weight parameters and presentation order of training exemplars. As a result, one can choose the smallest error which has converged. This approach enables to avoid good local minima for better neural network architecture selection.

Based of the above notion, there is a selection algorithm for feature selection. This algorithm is designed for systematically investigating reduced neural network models with fewer feature inputs and fewer hidden nodes [63]. The algorithm uses the likelihood-ratio test statistic as a statistical model selection criterion for testing reduced models. Figures from 2.1 to 2.3 provide additional details as to how the likelihood-ratio statistic is specifically incorporated into both feature and architectural selection. The selection algorithm can be implemented with two different stopping rules. The first stopping rule is based on statistical considerations; the algorithm stops when the likelihood-ratio test statistic L is greater than the selected critical point of the F-distribution. The second stopping rule gives the practitioner more flexibility in model selection. This rule involves stopping the algorithm after a predetermined final number of features, f, have been eliminated. As long as the current number of features to be selected, the candidate feature for elimination is removed. The second stopping rule makes the selection algorithm an effective search procedure, uncovering a number of potential neural network models for consideration. Each potential model can be examined in terms of tradeoffs between accuracy and parsimony.

The number of different neural networks trained for the full and for reduced models was usually limited by the time and computing resources available. The use of multiple neural networks in this selection algorithm put practical limit on the initial number of candidate features M which can be considered. This approach is the most difficult to give or set the appropriate number of features from feature pool.

The initial number of middle nodes H should be large enough to ensure there is sufficient complexity for accurate prediction on a test data set. However, an excessive number of middle nodes will unnecessarily increase the computational cost and the possibility of accepting a model with more middle nodes than necessary. Cover's rule Equation (2.4.1) suggests an upper-bound for H which is based on the separating capacities of families of nonlinear decision surfaces [68]. Cover shows that a family of surfaces having s degree of freedom has a natural separating capacity for $2s$ training exemplars [68]. Therefore, unless the number of training exemplars is greater than the separating capacity of $2s$, there is a large probability of ambiguous generalization [68]. Cover's theorem translates into the following upper-bound on the number of middle nodes:

$$H \leqslant \frac{0.5P - 1}{M + 1} \tag{2.4.1}$$

where P is the number of exemplars and M is the number of features. It is possible that when the number of candidate features is large and the number of exemplars is small, Cover's rule may indicate fewer middle nodes than are required for the complexity of the problem at hand (a situation where feature reduction is necessary).

In this case, one should proceed with caution since the neural network's ability to generalize to unknown data can be easily compromised in this situation. Cover's rule is only one heuristic technique to choose the initial number of hidden units. Cover gives the largest cardinality of a set of training vectors such that any possible class assignment of the vectors can be implemented (by the network) with probability one.

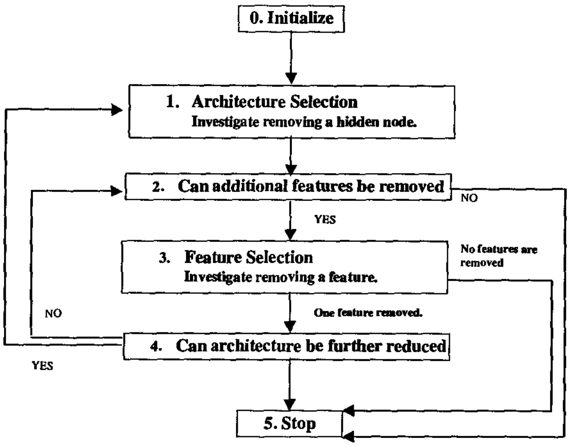
0. Initialize
1. Architecture Selection
Investigate removing a hidden node.
2. Can additional features be removed
NO
YES
3. Feature Selection
Investigate removing a feature.
No features are removed
NO
One feature removed.
4. Can architecture be further reduced
YES
5. Stop

Figure 2.1: Overview of neural network feature selection algorithm

As an alternative to using Cover's rule, the initial number of middle nodes could be determined by using the works of Baum and Hassler [13], Sontage [69], or Vapnik and Chervonenkis [70].

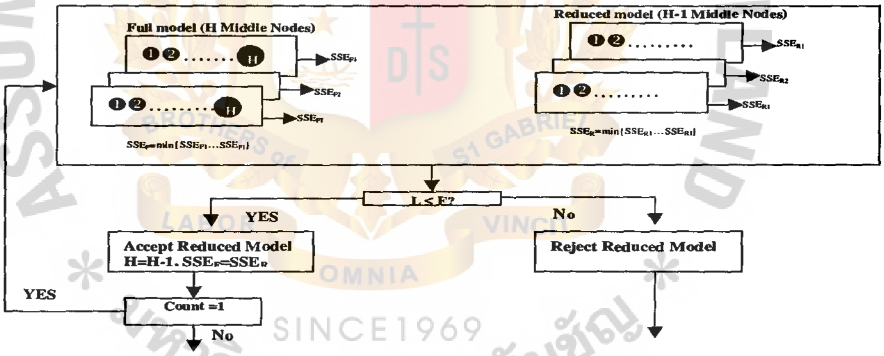

Figure 2.2: Architecture Selection

When computing resources are available, the selection algorithm is best utilized by performing more than one initialization of H with the upper-bound suggested by Equation (2.4.1). This provides the practitioner with additional insight and may reduced the risk of potentially accepting a larger than necessary network structure. The downside is that this approach can also complicate the goal of automatic architecture

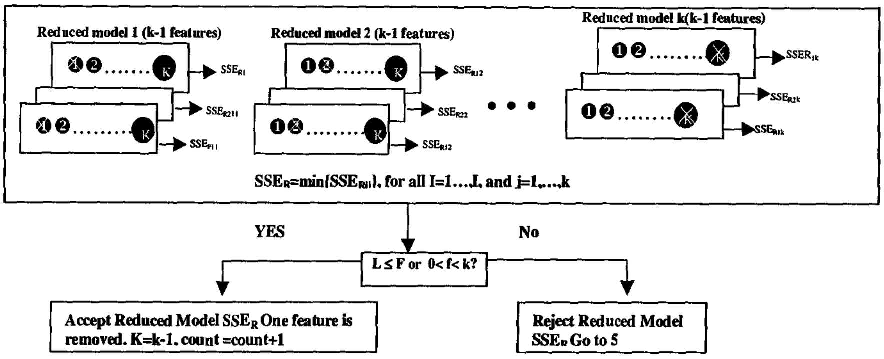
Reduced model 1 (k-1 features)
Reduced model 2 (k-1 features)
Reduced model k(k-1 features)
SSE_R1
SSE_R211
SSE_F11
SSE_R12
SSE_R22
SSE_R12
SSER_1k
SSE_R2k
SSE_RJk
SSE_R=min{SSE_Rij}, for all I=1…J, and j=1,…,k
YES
No
L≤F or 0<f<k?
Accept Reduced Model SSE_R One feature is removed. K=k-1. count =count+1
Reject Reduced Model SSE_R Go to 5

Figure 2.3: Feature Selection

and feature selection, since multiple selection runs may produce multiple solutions which must be subsequently analyzed [63]. In general, when choosing from variety of network models which all have equivalent or acceptable prediction accuracies, the smallest network model is preferred.

2.5 Drawbacks and Benefits of above methodologies

In this section, we summarize the above discussed methodologies so that it clarifies the use of the new proposed algorithm (FIFAS) by taking all possible benefits of them. In other words, it tries to take the advantages of each methodology. Therefore, they are displayed on pages 41 and 42 in tables 2.3 and 2.4.

Table 2.3: Drawbacks and Benefits of each methodology

Methodology	Drawbacks	Benefits
Neural Networks	• No clear rules or design guidelines for arbitrary application • Training may be difficult or impossible • Difficult to predict future network performance (generalization)	• Inherently massive parallel • May be fault tolerant because of parallelism • Little need for extensive characterization of problem (other than through the training set)
Resilient Backpropagation		• Few parameters that are required to be set by the user • The performance of it is insensitive to the values set • Fast training algorithm
Constructive Algorithm	• Needs additional criterion when to stop the addition of hidden units • Many algorithms use a greedy approach to construction purpose which is sub optimal in most cases. • There may be problems in achieving good generalization when care is not taken in handling hidden units with many parameters associated.	• Straight forward to set an initial network • Learning is faster than pruning algorithm • Knowledge embedded in small trained networks is easier to interpret and thus the extraction of simple rules can hopefully be facilitated • Require limited resources in physical computational environment • Better generalization • Likely to find smaller network solution than pruning algorithm
Up Start algorithm	• Designed for the use of binary neurons only • Did not use regularization term • Did not use cross-validation and early stopping principles	• Starts from minimal resources
Backward integrated feature and architecture selection	• Requires prior knowledge of the underlie problem especially to set the number of features to be the final limit • Uses pruning principle.	• Takes advantages of each of feature and architecture algorithm. • Determines "optimal" accuracy with few parameters.

Table 2.4: Drawbacks and Benefits of each methodology

Methodology	Drawbacks	Benefits
Flexible algorithm	• Did not use cross-validation and early stopping principles • It is difficult to measure the performance and where to stop the training regarding number of hidden neurons and layers • Convergence has taken longer time compare to others	• Measures the change in error when hidden unit or weight in the network removed. Such changes can be only approximated for computational efficiency and hence may introduced larger errors especially when many are to be pruned. • Add many neurons and layers at a time • No Constraints are imposed regarding the number of hidden layers and hidden units
Pruning algorithm	• Difficult to determine how big the initial network should be • More expensive than constructive	• Large initial size allows the network to learn "reasonably quickly" with less sensitive to initial conditions and local minima • The reduced complexity of the trimmed system favors improved generalization. • Avoids overfitting problems
PCA	• It is difficult to scale up the components resulted using PCA	• Uses the most expressive features and effectively approximates the input features using mean square error criterion. • Unsupervised feature selection method.
Likelihood ratio test	• Approximates the system - not calculate exactly	• Not affected by parameter effect curvatures using mean square error criterion. • More powerful than Lagrange test statistics • Not require the estimation and inversion of covariance matrix of the weight parameters.

Chapter 3

New Algorithm–Forward Integrated Feature and Architecture Selection

3.1 The FIFAS Algorithm

The motivation or inspiration was developed during the study of backward unified feature and architecture selection which is done by [63]. The proposed FIFAS algorithm considers from bottom to top design approach.

3.1.1 Statistical approach using likelihood ratio test

Nonlinear regression statistical model selection was used recently as the basis for reconstructing an integrated architecture and feature selection algorithm with backward selection (top-down) methodology in feed-forward neural networks [63]. Their criterion was based on likelihood ratio test statistic. We use bottom-up procedure: forward selection (bottom-up) approach for architecture selection only.

From statistical model selection point of view using neural networks, it is necessary to test formulated hypothesis. The hypothesis can be based on the neural networks weight parameters in which they are either equals to zero or not. In regression model, if the inclusion of additional variable(s) to the existing model does not give a better result, then one can conclude that the inclusion of additional variable (s) is not worthwhile. In other words, the decrease in the sum of squares for error when one goes from the partial (reduced) model to the full model is not statistically significant. The same analogy is used in neural network forward selection algorithm from statistical fine points.

The single-output neural network for P training exemplars is defined in Equation 3.1.1 as a univariate response nonlinear regression model

$$d = z(X, w^{*}) + \varepsilon \tag{3.1.1}$$

where d is P x 1 vector of true "desired" network outputs; z(X,w*) can be interpreted as E(d|X), a P x 1 vector of network responses conditioned on a P x M matrix of feature input variables X; w* is an s x 1 vector of unknown optimal weight parameters; and ε is a P x 1 vector of neural network errors. The least-squares estimator of w* is an s x 1 vector w_1 that minimizes the sum of square errors (SSE) with respect to w and is given as

$$w_1 = argminSSE(w) \tag{3.1.2}$$

where

$$SSE(w) = [d - z(X, w)]'[d - z(X, w)] \tag{3.1.3}$$

To present the general hypothesis test, we define w* as an s x 1 vector of optimal neural network weight parameters and S as a q x s binary selection matrix of ones or zeros which is multiplied by w* to select the q specific weights of w* which are hypothesized to be equal to zero. That is, whenever the n^{th} the weight if w* is hypothesized to be zero, there is a corresponding row in S which has a one in the n^{th} position and zeros in every other position. A formal hypothesis test for testing the statistical significance of a q-dimensional subset of weights from w* is given as

$$NullHypothesis(H_0) : Sw* = 0 \tag{3.1.4}$$

$$AlternativeHypothesis(H_A) : Sw* \neq 0 \tag{3.1.5}$$

There are two basic categories of statistical model selection criteria for evaluating the hypothesis test given in Equ. 3.1.4:

1. standard test statistics for situations where the nonlinear regression model is assumed to be correctly specified with normally distributed errors
2. specification-robust test statistics for situations where the nonlinear regression model is not correctly specified [71][72][73].

The standard selection criteria for situations where the nonlinear regression model is assumed to be correctly specified are the Wald test statistic, the Lagrange multiplier test statistic, and likelihood-ratio test statistic [71][72]. Our hypothesis for architecture selection is that the weights associated with the hidden nodes and output neurons are tested to see if they are statistically different than zero. If the tested weights are not statistically different from zero, a partial (reduced) architecture model is appropriate. For the case of feature selection algorithm, we used one of best-unsupervised feature selectors-Principal Component Analysis (PCA). The same principle for feature selection using feature similarity is now under investigation. This procedure may lead to a better result with the constructive feed-forward neural networks algorithm; consequently our approach will be strong enough to solve any classification problems.

The purpose of using one of the best feature selectors is to eliminate some redundant and/or irrelevant features before proceeding with architectural selection using neural networks. This approach enables to satisfy the normality assumption for the specific-robust test statistics [13]. The theoretical foundation for using likelihood ratio test is somehow compromised with neural network selection scenario. The advantages of using likelihood ratio test using formulated hypothesis for forward selection algorithm are

- it is not affected by parameter effect curvatures [74]

- it is more powerful than Lagrange test statistics [67]
- it does not require the estimation and inversion of covariance matrix of the weight parameters[13].

The likelihood ratio test, sometimes it is similar to Partial F statistic test for F distribution, is given by 3.1.6

$$L = \frac{\frac{SSE_P - SSE_F}{r}}{MSE_F} \tag{3.1.6}$$

where $\mathrm{SSE_P}$ is the sum of squares for error of the partial model; $\mathrm{SSE_F}$ is the sum of squares for error of the full model; $\mathrm{MSE_F}$is the mean square error of the full model: $\mathrm{SSE_P}$/(n-(k+1)); k is the number of weight parameters in the full model; and r is the number of weight parameters dropped from the full model in creating the partial model.

The difference $\mathrm{SSE_P}$-$\mathrm{SSE_F}$ is called the extra sum of squares associated with the partial model. Since this addition sum of squares for error is due to [n-(k+1)]-[n-(k-r+1)]=r weight variables, it has degrees of freedom. The extra sum of squares for error has degrees of freedom . Therefore, whenever L exceeds the critical point of the F-distribution, the partial model is rejected. For the selection of neural network architecture, we use this test.

3.1.2 Regularization Term

The purpose of including regularization term is to improve the generalization. This can be implemented by modifying the performance function in feed-forward neural networks. For this study we include a regularization term that encompasses the minimization of both mean square error (mse) and mean square of weights and biases (msw). The equation is described in Equation 3.1.7

$$msereg = \gamma mse + (1-\gamma) msw \tag{3.1.7}$$

where γ is performance ratio and

$$msw = \frac{1}{n}\sum_{j=1}^{n} w^2{}_j$$

Using this performance function will cause the network to have smaller weights and biases, and this will force the network response to be smoother and less likely to overfit.

3.1.3 Algorithm

In this section, we present the algorithm for architecture selection along with the available variable number of feature sets. For the elimination of irrelevant and/or redundant features, we have used PCA. With candidate features, we have started the construction of feed-forward neural networks with one hidden unit and then continued the construction iteratively with one hidden unit at a time until the criteria is met. The selection of the architecture depends on the way the network trained - either for a fixed feature candidate, construct the network or for a fixed architecture, search optimal number of feature(s). The comparison result is presented in experimental results section.

Procedure for the algorithm:

1. Elimination of "irrelevant" and/or "redundant" features-produce candidate features with any feature selector, in this case use PCA for successive contribution such as from 0.1% to 4.0%
2. Train the network
    (a) Construct neural networks with one hidden unit (h=1) in single hidden layer at the start
    (b) Train the neural networks with selected candidate k feature(s) for several times and consider it as full architecture, denotes $F_1,\dots,F_L$. Set $SSE_F=$

$min\{SSE_{F_1}, \ldots, SSE_{F_L}\}$

(c) Construct new networks with one additional hidden unit (h=h+1) for each successive training

(d) Train the newly constructed networks with k feature(s) for several times and consider it as full model and the previous one as partial model, denotes $F_1, \ldots, F_L$ with $SSE_F = min\{SSE_{F_1}, \ldots, SSE_{F_L}\}$ and $P_1, \ldots, P_L$ with $SSE_P = min\{SSE_{P_1}, \ldots, SSE_{P_L}\}$,respectively.

(e) Test the null hypothesis that the reduced model is equivalent to the full model

  i. Calculate the likelihood ratio

  ii. Put the decision for accepting or rejecting the partial model for a given k features

    A. If $L \leqslant F$ critical value, accept the partial model and go to 2f

    B. If the decision is rejection, then set $SSE_P = SSE_F$ for further training and go through 2c to 2e till upper bound of hidden units reaches (Note: During training stages, the algorithm records processing time, validation accuracy, regularization performance, classification rate on SSE, and the corresponding MSEREG for further comparisons of the architecture selection.)

(f) Chose the "enhanced" architecture as long as classification accuracy on MSEREG is greater than SSE against either:

  i. The first acceptable architecture or

  ii. The highest classification rate. Compare classification error with previous acceptable model. If it is the minimum until the current phase, retain it. If it is not promising, go through 2c to 2e once again

  Note: 1) The above step deals with hidden units only where as Step

3 deals with number of features. 2) The classification rate is flexible and set by user.

3. If the classification rate is not in the acceptable range, add more feature(s) depending criterion set by feature selector such as the next contribution in PCA, otherwise go to Step 5
4. Repeat Step 2
5. Stop, the final neural network model has h middle nodes in single hidden layer and k features.

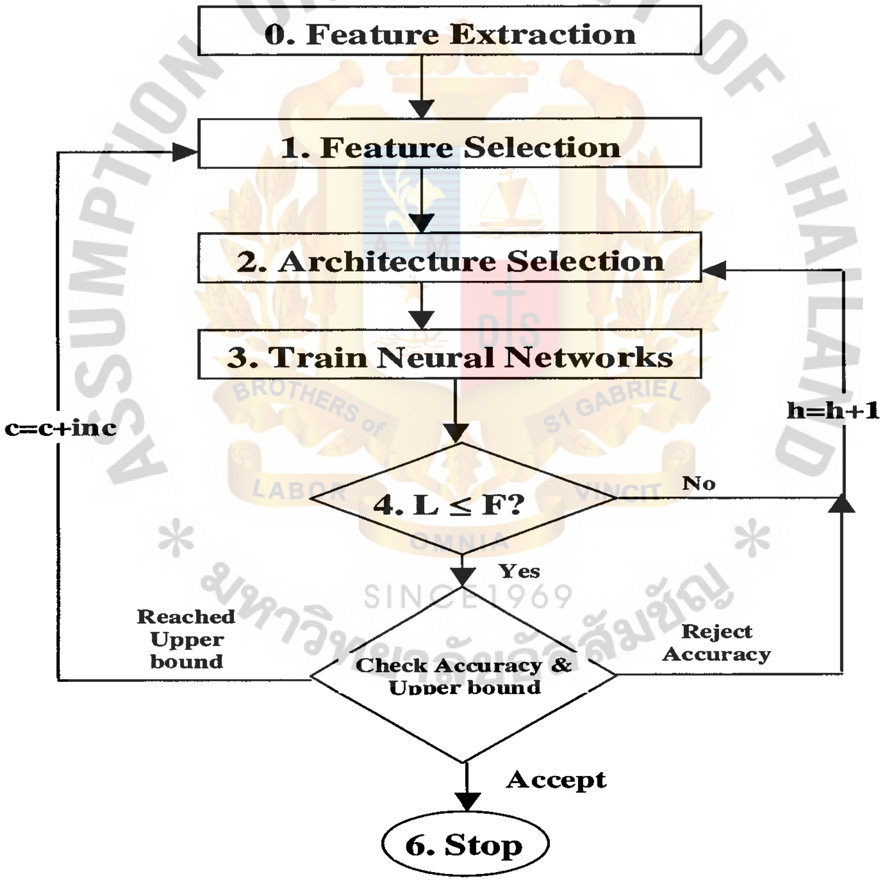

Figure 3.1: FIFAS Algorithm

Feature Extraction

In this stage, all necessary feature sets are produced from image of characters. In the mean time, before one proceed to the next stage, it is required to set the classification accuracy that be be the minimum and the increment of the contribution of feature selection in PCA. That is, Set accuracy rate = % and Incremental (inc)= %

Feature Selection

In this stage, the selection process has two steps- one transforms the data into other form so that the second step selects only prominent or relevant features from the pool of features. In this particular study, the feature selection uses PCA which requires the transformation of data into other forms unlike other feature selection algorithms such as Forward Search, Backward Search, Forward-Backward Search. For example, produce a number of features with PCA contribution of 0.1% contribution. For further contribution, it increments by the increment value set at the beginning.

Architecture Selection

Construct (design) a topology of neural networks with initial hidden neuron to 1. Consequently, the algorithm design a topology constructively one hidden neuron as a time until either it reaches the acceptable accuracy or upper bound of hidden neurons.

Checking Accuracy and Upper Bound of hidden neurons

- If the accuracy is less than the preset value and upper bound is not reached, $\Longrightarrow$ h=h+1 (h is number of hidden neurons)
- else Accuracy is less than the preset value and upper bound is reached, $\Longrightarrow$ c= c+inc (c is contribution and inc is increment)
- else Accept the integrated feature and architecture

Likelihood ratio test

Test $L \leq F$?

Train the neural networks

The algorithm trains several times and choose the minimum performance error.

The modified algorithm for the investigation of the importance of the order is to train with all features with fixed topology in the exchange of the addition of hidden units by the addition of number of features in the above steps.

Chapter 4

Implementation

4.1 Benchmark methodology

In the past there has been some criticism of feed-forward benchmarking methodology, and suggestions for improvement [75][76] and [77]. The remedies for these requirements are the data sets were partitioned into three groups- training, testing and validating data sets. Secondly, train the networks for 7 times for given hidden units and features in order to avoid the randomization effect of the weights.

The training algorithm used in this study is the Resilient Backpropogation (RPOP) algorithm [78]. RPROP was selected for a number of reasons. First, it is a gradient-based method which has fast convergence properties compared to many other gradient based algorithms such as BP and its variants [78]. A further advantage of RPROP is that there are few parameters that are required to be set by the user. In addition, RPROP's performance is relatively insensitive to the values selected. The RPROP algorithm used the following standard constant settings: η^{+}=1.2, η^{-} =0.5, Δ_{max} = 50, Δ_{min} = 10^{-6}, performance ratio(γ)= 0.5, epoch=10,000, time = Infinity, goal= 0.001 and regularization term is Equ. 2.2.3.

The architecture selection algorithm does not perform weight freezing, but trains all weights after the addition of a new hidden neuron. The network architecture constructed is a cascade. Training is begun in constructive algorithm with an initial network that has one hidden neuron, with the input connected directly to the outputs through it. These weights are initialized to random values in the range -0.7 to 0.7. Once new hidden neuron is connected to the network, training is resumed using RPROP. The weights of the new hidden neuron are initialized to random values in the range -0.1 to 0.1.

The remaining RPROP parameter is the initial update value Δ_0, which sets the initial step size taken by the weights. For the initial network this value is set to 0.2. When a new hidden neuron is added to the network, the update values are reset to values depending on their position in the network. This technique has been shown to increase convergence speed and is termed search direction biasing [80][81].

In the case of feature selection algorithm for FIFAS, the Principal Component Analysis has been used with the explanation of PCA uses singular value decomposition to compute the principal components. The input vectors are multiplied by a matrix whose rows consist of the eigenvectors of the input covariance matrix. This produces transformed input vectors whose components are uncorrelated and ordered according to the magnitude of their variance.

Those components that contribute only a small amount to the total variance in the data set are eliminated. It is assumed that the input data set has already been normalized so that it has a zero mean. As a result, the transformed components with the requirement of Variance Accounted For (VAF) parameter have been formulated. In other words, by giving the contribution factor by user, a minimum number of components that are necessary to explain the given data set have been produced. Thus, for this particularly thesis, the range of contribution factor has been given from 0.1% to 4.0% for successive increment of 0.1. Additionally, if the numbers of components (features) are the same, then the implementation phase only considered one of them in order to minimize the training time substantially.

For the analysis purpose, the network is taken the minimum values of all the seven trainings. Besides, early stopping was used as a halting criterion. The training is lasted until the MSEREG or/and SSE or validation error failed to decrease by a set amount error over a given period or reached the maximum limit of hidden units base on Cover's theorem [68]. Every network was monitored for the improvement of every 25 epochs to justify the continuation of the training. The weight parameters vary when the number of hidden units changes since it calculates by the interconnections all neurons among the adjacent layer(s) in input, hidden and output layers. This can be given by (k+1)*h + (h+1) where k is the number of input features and h is the number of hidden units in hidden layer.

For the classification purpose, the performance measure was used the percentage of patterns misclassified on the new data set in which it based on mean square regularization error and/or summation of squared error. The selection of the magnitude of regularization is one of the most elusive values since it depends on the model which is implemented, the type of data sets and the final size of the network [45]. As a result, it affects the generalization of the network. In order to be on the safest side, set the value 0.5 that will have equal chances between error and weight values. In addition to these, time elapsed to train the networks, likelihood ratio test measure and training performance on regularization error, or/and summation of square error have been recorded for comparison.

For the classification problem of the study, we have used two different data sets. The first data set is dealt with Geez characters. Geez characters are one of the scripts used in East Africa, Eritrea and Ethiopia. It has various roots and derivatives. These characters are collected from the Geez word processing software. We have selected 412 unique characters for each of 7 font types. All in all 2884 exemplars were used for neural networks functionality. The preparation process was, each character has scanned and produced character images. Then the two-dimensional pixel arrays of the input characters were preprocessed using image-processing techniques - normalize

Table 4.1: Description of Geez Character features

Feature no	Feature	Desrciption
1	Total Euler	Total number of Euler in connected region
2	Total Area	Total number of area in connected region
3	Area	The actual number of pixels in the connected region.
4	Centroidx	X coordinate center of mass of the region.
5	Centroidy	Y coordinate center of mass of the region
6	Major Axis Length	The length (in pixels) of the major axis of the ellipse that has the same second-moments as the region.
7	Minor Axis Length	The length (in pixels) of the minor axis of the ellipse that has the same second-moments as the region.
8	Eccentricity	The eccentricity of the ellipse that has the same second-moments as the region.
9	Orientation	The angle (in degrees) between the x-axis and the major axis of the ellipse that has the same second-moments as the region.
10	Convex Area	The number of pixels in convex image
11	Filled Area	The number of on pixels in filled image.
12	Euler Number	Equal to the number of objects in the region minus the number of holes in those objects.
13	Equivalent Diameter	The diameter of a circle with the same area as the region.
14	Solidity	The proportion of the pixels in the convex hull that are also in the region.
15	Extent	The proportion of the pixels in the bounding box that are also in the region.
16	X Bounding	X coordinate of upper-left corner of the rectangle
17	Y Bounding	Y coordinate of upper-left corner of the rectangle
18	Width	The width of the rectangle.
19	Height	The height of the rectangle.
20	Extremal	The extreme points in the region.
21	Thinness Ratio	Measures the roundness
22	Perimeter	Perimeter of the region
23	Compactness Ratio	Determines the regularity of an object

threshold and smoothing. Calculated spatial information in which considered features of the character using Matlab Toolbox version 6.0 [43] wherein some of them, are described on Table 4.2.

The second data set is the digit data set[1] which consists of handwritten numerals ('0'-'9') extracted from a collection of Dutch utility maps. Two hundred patterns per class (for a total of 2000 patterns) are available in the form of 30 x 48 binary images. These characters are represented in terms of the following feature sets (total

---

[1] The data set is available through the University of California, Irvine Machine Learning Repository (www.ics.uci.edu/~ mlearn/MLPRespository.html).

649 features):

1. 79 Fourier coefficients of the character shapes;
2. 216 profile correlations;
3. 64 Karhunen-Loeve coefficients;
4. 240 pixel average in 2 x 3 windows;
5. 47 Zernike moments; and
6. 6 morphological features.

4.2 Parameter settings

This subsection summarizes all necessary parameters with their values in the implementation phase of the study. The experimental results are presented and discussed in the Experimental Results and Discussion chapter.

Table 4.2: Parameter Settings

Parameter Name	Parameter value	Parameter Name	Parameter Value
Training set	10,000	Time	Infinity
Maximum Fail	50	Minimum Fail	50
Goal	0.1%	Hidden Layer Transfer Function	logsig
Output Layer Transfer Function	Purelin		
Number of Input neurons	Depends on the number of features	Number of output neurons	10 (English) and 16 (Geez)
Performance function	MSEREG and SSE	Performance Ratio	0.5
Weight Initialization of hidden layer	-0.1 to 0.1	Weight Initialization of input layer	-0.7 to 0.7
Training update	25 epochs	PCA contribution	0.1% to 4.0% (0.1 increment)
Number of exemplars	2000 (English) and 2884 (Geez)	Likelihood ratio confidence interval	99%

Chapter 5

Experimental Results and Discussions

In this chapter, the study presents the experimental results of two benchmark datasets. The organization of the results is as follows: First, describe the criteria to measure the behavior of the proposed algorithm. Then discuss briefly the motivation of using two real world datasets. Next compares the results against the measures. In order to test the robustness and usefulness of the proposed algorithm, we present empirical analysis from simulation procedures of neural networks by comparing the classification rate using MSEREG and SSE on both real world datasets. Withal this, we add in the concept of which pillar or constituent (feature or architecture) comes first toward the "acceptable" or "enhanced" integrated feature and architecture recognition system construction.

In a general sense, we adopted the following measures to evaluate the behavior of the proposed selection algorithm in terms of:

1. Number of hidden nodes in the constructive networks for a given or fixed number of features;
2. Number of features in constructive networks;

3. Classification rate, measured as the proportion of examples for which all network output values differed from the corresponding target;
4. Usual SSE measure and MSEREG measure
5. Network performance on both validating and testing dataset as well as training dataset

The experiments were conducted on two different categories of real-life public domain datasets, which is described in Section 4.1. The motivation behind this is to figure out the effect of medium-dimensional (number of features as a range between 10 and 100) and high-dimensional (number of feature greater than 100) effect [79].

The feature selection component in this algorithm is dynamic since user only gives the amount of principal contribution and hence makes available number of features for architectural selection procedure.

The experimental results are mainly focus on the first acceptable architecture with appropriate features. The first acceptable model or architecture refers or points to the architecture that found appropriate for the representation of the full model of the network using likelihood ratio test. Of course, the main objective of the study is to find the unified feature and architecture solution for OCR problems. However, the following tables and figures show the validity of the proposed algorithm (FIFAS) and describe how to select the appropriate ones. Consequently, full model or architecture considers as the model that attains the highest classification accuracy although one can consider all possible number of weight parameters or maximum number of hidden neurons according to Cover's rule as long as it does not break the requirement. This is because the maximum value can not reach with the principle of early stopping and cross-validation principles otherwise the training will be overfitted.

5.1 Handwritten English Numerals

To assess the generalization ability for handwritten English numerals (classification accuracy in new un-seen dataset during training stage), the whole dataset was partitioned into training set in which 49.50%, testing dataset 25% and the remaining 25% was assigned for validating dataset. The generalization ability is given as averages over 10 unseen new (recall) datasets.

From the feature selector, it achieves 20 prominent feature categories among 649 features that are displayed in every table as under heading of number of features.

Table 5.1: Highest accuracy based on MSEREG without considering first acceptable model of English numerals

Cover's Rule	No of Features	No of Hidden units	Epoch	Elapsed time	Classification Rate MSEREG	Classification Rate SSE	No of weight parameters
142	6	47	112	36.11	91%	89%	377
124	7	50	61	17.91	89%	84%	451
110	8	41	129	28.87	89%	83%	411
99	9	36	64	12.71	91%	82%	397
90	10	42	94	20.12	90%	87%	505
83	11	76	186	69.59	90%	73%	989
76	12	68	113	38.40	88%	81%	953
71	13	67	130	43.35	89%	79%	1,006
66	14	54	148	36.83	90%	80%	865
58	16	60	98	26.69	89%	80%	1,081
52	18	26	140	19.64	90%	83%	521
50	19	48	94	20.99	91%	79%	1,009
45	21	51	81	19.05	89%	78%	1,174
41	23	46	130	27.79	91%	81%	1,151
38	25	79	111	41.02	91%	75%	2,134
33	29	69	146	47.48	92%	74%	2,140
28	35	59	154	39.54	92%	80%	2,184
22	45	66	214	65.99	92%	81%	3,103
15	65	74	189	64.29	90%	75%	4,959
9	109	61	202	52.40	90%	77%	6,772

Table 5.1 describes the maximum accuracy that the training algorithm can attain based on mean square error of regularization performance function (MSEREG) along with the given or set parameters from section 4.2. Besides, it gives the maximum number of hidden neurons, epoch, elapsed time, number of weight parameters and

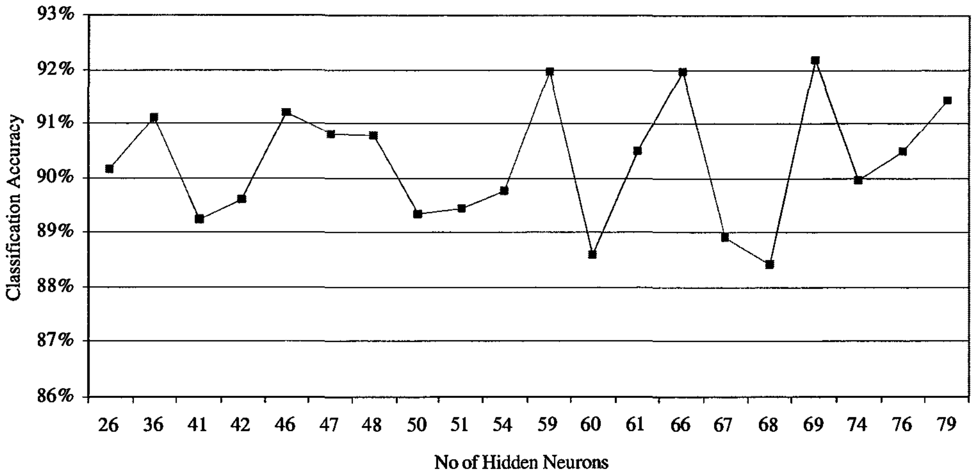

Figure 5.1: Number of Hidden Neurons Vs Highest Classification Rate based on MSEREG of English numerals

the corresponding classification accuracy based on sum of square error (SSE) to attain the displayed highest classification accuracy on MSEREG. According to Cover's rule [68], the maximum number of hidden neurons for the given number of features and training exemplars are put in the first column and the remaining tables. This table does not include the decision criterion in order to take as a reference for the subsequent decision for enhanced architecture of neural networks. Figure 5.1 shows the relationship between number of hidden units and classification accuracy for each relevant feature. This figure is the summary of table 5.1 when it sorts by number of hidden neurons. Similarly Figure 5.2 depicts how the number of weight parameters affects the classification accuracy of handwritten English numerals based on MSEREG performance function. Hence the underlining training have achieved the maximum of 92% along with 2140 weight parameters. These weight parameters are enormous to resolve the problem when they are compare to the partial model. It is reduced to 98% which is a big and considerable reduction without much reduction accuracy in classification accuracy - from 92% to 88% (4.3% in change). In the first benchmark data set which is displayed on table 5.2, there is a peculiar anomalies for the determination of classification accuracy with the relevant features. These are indicates by asterisk (*)in the classification accuracy column. As a matter of fact, the inclusion of penalty (regularization) term in the training neural networks prompts or gives forth a higher accuracy than without the term due to longer training without breaching the early stopping principles. However, some features don't conform with the theoretical foundation. The reason behind for these anomalies might be the network has not reached to the optimal stage to generalize the results. Therefore, to complete approach for the FIFAS algorithm, these results should be rejected automatically. On the other hand, the model reduction can be explained in terms of either the maximum hidden neurons using Cover's rule or the number of weight parameters. The former one has its own drawbacks and the latter has not even though both of them includes the number of input features, number of hidden units and output neurons for single hidden layer of neural networks. Therefore, the unified and enhanced solution for handwritten English numerals is 88% in classification accuracy with 89% of model reduction. In other words, 6 features and 5 hidden neurons are the appropriate solution for handwritten English numerals.

Figure 5.3 is given us a comprehensive results between number of features and number of hidden units of full and partial models topologies. Moreover, it puts the relationship between number of features and classification accuracy for the corresponding architectures. It is evinced that the rejection of model due to the classification accuracy based on SSE is greater than MSEREG even though it is acceptable from the likelihood ratio test. The explanation for this is that the neural networks are not accomplished the desired result in generalization of the networks. This can be overcome by letting the training until it reaches the acceptable classification accuracy as long as the model is in the acceptable territory.

In contradistinction to Tables 5.1 and 5.2, the reduction of model in terms of weight parameters is more or less constant for various relevant features whereas reduction of model in terms of Cover's rule is a substantial changes. The aim of the

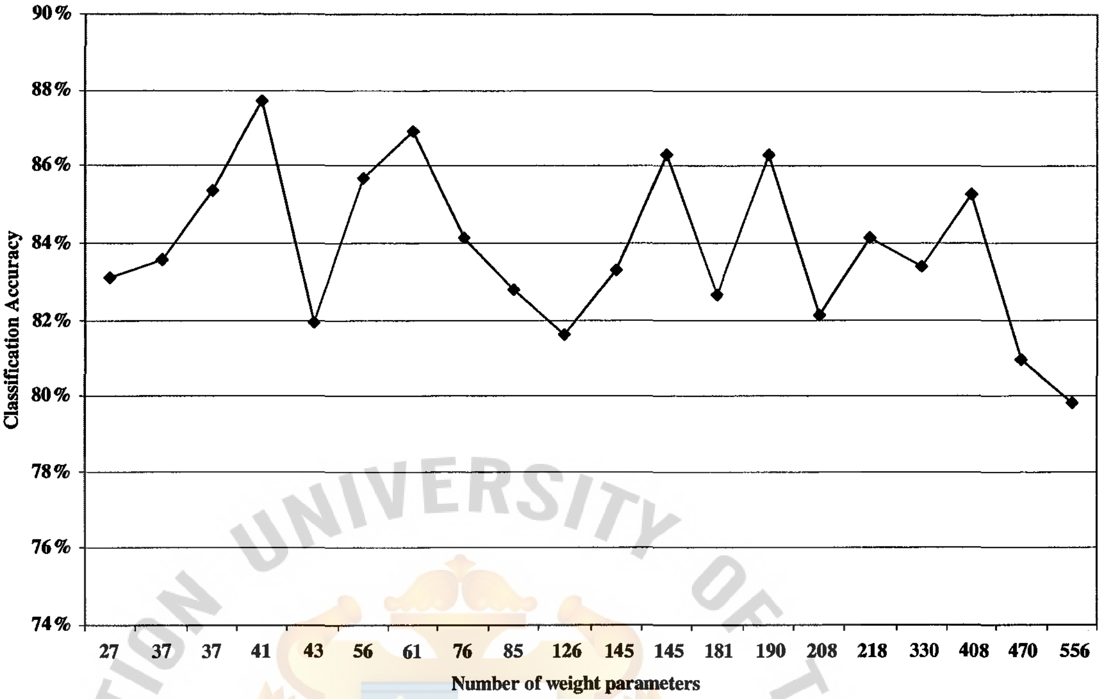

Figure 5.2: No of weight parameters Vs Classification Accuracy based on MSEREG of English numerals

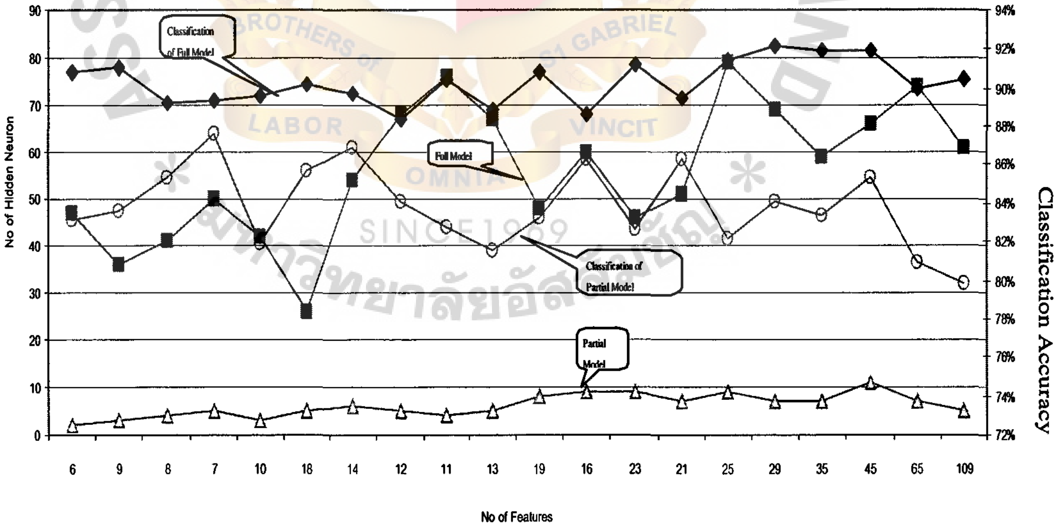

Figure 5.3: Number of Features Vs No of Hidden neurons and Classification Accuracy of English numerals

Table 5.2: First acceptable model based on MSEREG of English numerals

Cover's Rule	No of Features	No of Hidden Neurons	Model Reduced	Epoch	Elapsed Time	Classification Accuracy on MSEREG	Classification Accuracy on SSE	No of weight parameters
83	11	2	96%	42	2.99	*83%	87%	27
90	10	3	97%	54	3.84	*84%	90%	37
124	7	4	97%	36	3.26	*85%	91%	37
142	6	5	96%	89	9.40	88%	83%	41
76	12	3	96%	71	4.68	82%	79%	43
99	9	5	95%	43	3.50	*86%	89%	56
110	8	6	95%	119	9.67	*87%	88%	61
71	13	5	93%	143	9.91	*84%	91%	76
50	19	4	92%	144	9.00	*83%	86%	85
41	23	5	88%	47	3.38	*82%	89%	126
58	16	8	86%	64	5.17	*83%	86%	145
66	14	9	86%	33	2.99	86%	83%	145
52	18	9	83%	41	3.55	83%	81%	181
38	25	7	82%	52	4.03	86%	78%	190
45	21	9	80%	85	6.80	*82%	87%	208
33	29	7	79%	60	4.51	84%	81%	218
22	45	7	68%	44	3.34	*83%	91%	330
28	35	11	60%	75	6.43	85%	82%	408
15	65	7	54%	48	3.57	*81%	86%	470
9	109	5	45%	38	2.66	*80%	82%	556

proposed algorithm is to reduce the model as much as possible.

Thus based on the above two paragraphs, the choice is left to the designer of the recognition system since if one decides to train the networks as long as it saturates for generalization or decide to select only acceptable model without much resource usage. The latter case is suitable in line with the proposed algorithm.

Table 5.3 is another version of table 5.1. The only difference is that the former has produced based on decision criterion and the later does not have. This table (table 5.3) enables the designer to train the network until it reaches the maximum accuracy and without considering the first acceptable model which is the solutions of table 5.2.

Tables 5.4 and 5.5 describe the final errors during training process for highest classification and first acceptable models, respectively. It is not elicited for a trend for the all relevant features and number of hidden units that are dependent on various criteria such early stopping and cross validation to achieve the results. However one

Table 5.3: Acceptable model with highest accuracy based on MSEREG for English numerals

Cover's rule	No of Features	No Hidden Neurons	Epoch	Time Elapsed (sec)	Classification Accuracy	No of weight parameters
142	6	7	45	5.40	91%	57
124	7	50	61	17.91	89%	451
110	8	41	129	28.87	89%	411
99	9	31	92	16.09	90%	342
90	10	42	94	20.12	90%	505
83	11	18	63	7.60	90%	235
76	12	68	113	38.40	88%	953
71	13	60	141	38.77	89%	901
66	14	70	194	66.21	89%	1,121
58	16	60	98	26.69	89%	1,081
52	18	63	161	45.11	90%	1,261
50	19	27	117	16.68	91%	568
45	21	24	65	8.77	89%	553
41	23	46	130	27.79	91%	1,151
38	25	79	111	41.02	91%	2,134
33	29	78	114	41.44	92%	2,419
28	35	59	154	39.54	92%	2,184
22	45	66	214	65.99	92%	3,103
15	65	74	189	64.29	90%	4,959
9	109	77	164	57.54	90%	8,548

can observe the errors in training, testing and validating are in increase order which are the general truth of neural networks training. Moreover, one can compare the corresponding results of training, testing and validating errors from both tables.

Figure 5.4 tells us that the classification accuracy for variable relevant features supersede the fixed relevant features (variable model) except at the first relevant feature. From this, one may conclude that to get highest accuracy, it is better to use fixed topologies for variable features. However, this is not true if one consider the number of hidden neurons and hence number weight parameters along with the classification accuracy criteria. The results are displayed in table 5.2 and table 5.6 Consequently, with minimal weight parameters, it is better to find the classification accuracy which is selecting a feature and design the network constructively.

Table 5.4: Training, Testing, and Validation Error for highest accuracy model based on MSREG of English numerals

No of Features	No of Hidden Neurons	Epoch	Elapsed Time	Training Error	Testing Error	Validation Error
6	47	112.00	36.11	0.46%	1.49%	2.07%
7	50	61.00	17.91	0.65%	1.68%	2.35%
8	41	129.00	28.87	0.99%	2.01%	2.40%
9	36	64.00	12.71	1.33%	2.29%	2.77%
10	42	94.00	20.12	1.30%	2.27%	2.61%
11	76	186.00	69.59	0.85%	1.88%	2.26%
12	68	113.00	38.40	0.99%	1.98%	2.37%
13	67	130.00	43.35	1.05%	2.05%	2.49%
14	54	148.00	36.83	1.32%	2.23%	2.73%
16	60	98.00	26.69	1.27%	2.16%	2.62%
18	26	140.00	19.64	2.68%	3.40%	3.77%
19	48	94.00	20.99	1.79%	2.55%	2.96%
21	51	81.00	19.05	1.78%	2.53%	3.03%
23	46	130.00	27.79	1.98%	2.69%	3.14%
25	79	111.00	41.02	1.39%	2.17%	2.65%
29	69	146.00	47.48	1.60%	2.33%	2.66%
35	59	154.00	39.54	1.88%	2.56%	2.85%
45	66	214.00	65.99	1.86%	2.52%	2.79%
65	74	189.00	64.29	1.85%	2.51%	2.79%
109	61	202.00	52.40	2.30%	2.93%	3.04%

Table 5.5: Training, Testing, and Validation Error for the first acceptable model selection based on MSEREG of English numerals

No of Features	No of Hidden Neurons	Epoch	Elapsed Time	Training Error	Testing Error	Validation Error
6	5	89	9.40	4.54%	5.02%	5.21%
7	4	36	3.26	6.11%	6.36%	6.52%
8	6	119	9.67	5.52%	5.82%	6.16%
9	5	43	3.50	6.40%	6.62%	6.74%
10	3	54	3.84	8.08%	8.17%	8.23%
11	3	42	2.99	8.27%	8.38%	8.42%
12	3	71	4.68	8.45%	8.53%	8.59%
13	5	143	9.91	7.17%	7.29%	7.40%
14	9	33	2.99	5.54%	5.94%	6.13%
16	8	64	5.17	5.95%	6.30%	6.43%
18	9	41	3.55	5.81%	6.16%	6.36%
19	4	144	9.00	8.33%	8.42%	8.45%
21	9	85	6.80	6.09%	6.40%	6.52%
23	5	47	3.38	7.97%	8.07%	8.20%
25	7	52	4.03	7.14%	7.29%	7.34%
29	7	60	4.51	7.21%	7.40%	7.44%
35	11	75	6.43	5.96%	6.21%	6.29%
45	7	44	3.34	7.50%	7.65%	7.61%
65	7	48	3.57	7.60%	7.75%	7.78%
109	5	38	2.66	8.80%	8.89%	8.86%

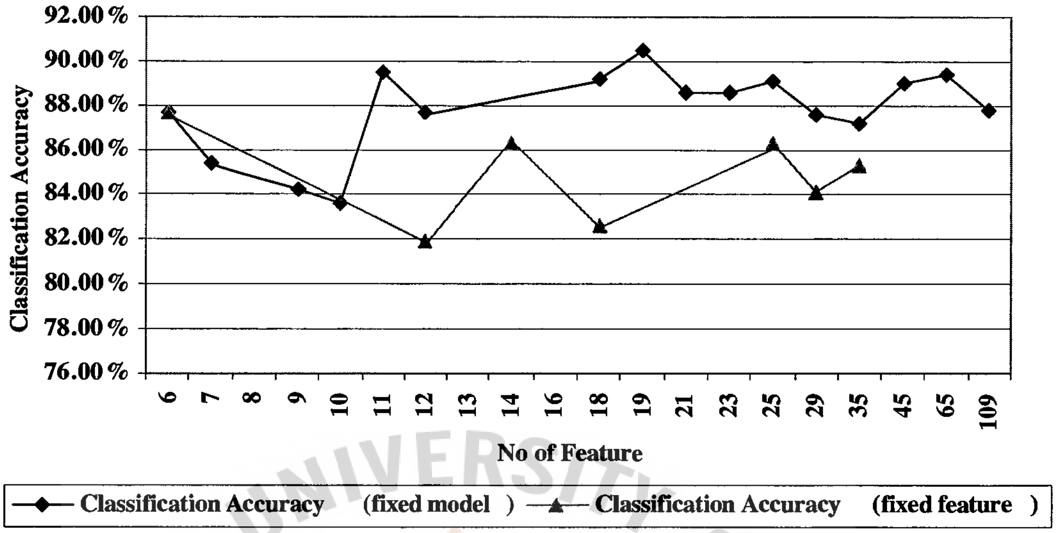

Figure 5.4: Number of Features Vs Classification Accuracy for fixed architecture and feature of English numerals

Table 5.6: First acceptable model based on MSEREG for fixed model of English numerals

Cover's Rule	No of Features	No of Hidden Neurons	Epoch	Elapsed time (sec)	Performance Error	Classification Accuracy on MSEREG	No of Weight Parameters	Model Reduction
143	6	5	89	9.40	4.06%	87.68%	41	96.5%
125	7	4	36	3.26	5.43%	85.37%	37	96.8%
100	9	2	58	3.99	6.16%	84.25%	23	98.0%
91	10	3	54	3.84	6.85%	83.55%	37	96.7%
83	11	18	63	7.60	3.37%	89.50%	235	78.4%
77	12	9	38	3.45	4.64%	87.65%	127	88.3%
53	18	61	86	23.73	3.71%	89.18%	1,221	-16.0%
50	19	27	117	16.68	2.79%	90.50%	568	45.9%
45	21	24	65	8.77	3.82%	88.64%	553	47.1%
42	23	29	120	17.85	3.83%	88.60%	726	30.3%
38	25	32	100	15.98	3.54%	89.09%	865	16.7%
33	29	12	56	5.17	4.76%	87.65%	373	64.0%
28	35	16	96	9.66	4.94%	87.18%	593	42.3%
22	45	17	65	6.92	3.94%	89.02%	800	21.7%
15	65	51	103	23.41	3.89%	89.40%	3,418	-236.9%
9	109	58	197	49.09	5.15%	87.75%	6,439	-538.6%

5.2 Machine-Printed Geez Characters

In this subsection, the newly designed benchmark has been implemented to test the effectiveness and robustness of the proposed algorithm. This benchmark is categorized under medium-dimensional category so as investigate the effect of dimension.

As displayed in table 5.8, the reduction of architecture is reached to 99% from 94% for 9 relevant features among 26 features. It conforms with the notion of the proposed algorithm as less applicable to handwritten English numerals. In other words, within the acceptable range of highest accuracy (Table 5.7, the classification accuracy based on MSEREG is more or less similar to the reduced model and better than SSE which is without penalty (regularization) term for each table. Hence, one may conclude that the trainings were reached to their generalization criterion even though the accuracy is less than the expected highest accuracy. This hurdle can be overcome by choosing the suitable value of performance ration for regularization term.

Table 5.7: Highest accuracy without considering decision of Geez

Cover's Rule	No of Features	No of Hidden units	Epoch	Elapsed time (sec)	Classification Rate	No of weight parameters	Model Reduction	Training Error	Testing Error	Validation Error
144	9	5	193	21	86.98%	56	96.5%	7.1%	7.2%	7.1%
180	7	8	152	20	87.21%	73	95.6%	7.0%	7.2%	7.0%
120	11	6	165	19	87.03%	79	95.0%	7.5%	7.6%	7.4%
160	8	10	194	27	87.31%	101	93.8%	7.0%	7.1%	7.0%
103	13	7	123	15	87.15%	106	93.2%	7.4%	7.5%	7.4%
80	17	6	180	19	87.05%	115	92.5%	7.0%	7.2%	7.0%
131	10	12	212	31	86.89%	145	90.8%	7.3%	7.4%	7.3%
111	12	12	181	26	87.06%	169	89.2%	7.0%	7.2%	7.0%
96	14	12	192	28	87.13%	193	87.5%	7.5%	7.6%	7.5%

One can easily pick up the appropriate integrated feature and architecture for Geez character from Figures 5.5 and 5.6 by identifying the first highest classification accuracy which is the turning point for reduced (partial) architecture and the corresponding relevant number of features and hence 9 features and 86.97% of classification accuracy with 87.99% of the corresponding of classification accuracy of full model.

Table 5.8: First acceptable architecture of Geez based on MSEREG

Cover's Rule	No of Features	No of Hidden units	Epoch	Classification Rate	Classification Rate SSE	No of weight parameters	Model Reduction	Training Error	Testing Error	Validation Error
160	8	2	61	86.66%	82.06%	21	98.8%	7.6%	7.7%	7.5%
120	11	2	64	86.67%	85.16%	27	98.3%	8.4%	8.6%	8.4%
111	12	2	60	86.95%	85.92%	29	98.2%	7.7%	7.8%	7.6%
103	13	2	49	86.70%	85.28%	31	98.1%	8.0%	8.1%	8.0%
180	7	4	85	86.60%	86.13%	37	97.8%	8.5%	8.7%	8.5%
131	10	3	64	86.51%	74.30%	37	97.7%	8.6%	8.8%	8.6%
80	17	2	55	86.49%	71.58%	39	97.5%	8.6%	8.8%	8.6%
144	9	4	133	86.97%	84.29%	45	97.2%	7.5%	7.6%	7.4%
96	14	6	102	86.71%	85.87%	97	93.8%	8.7%	8.9%	8.7%

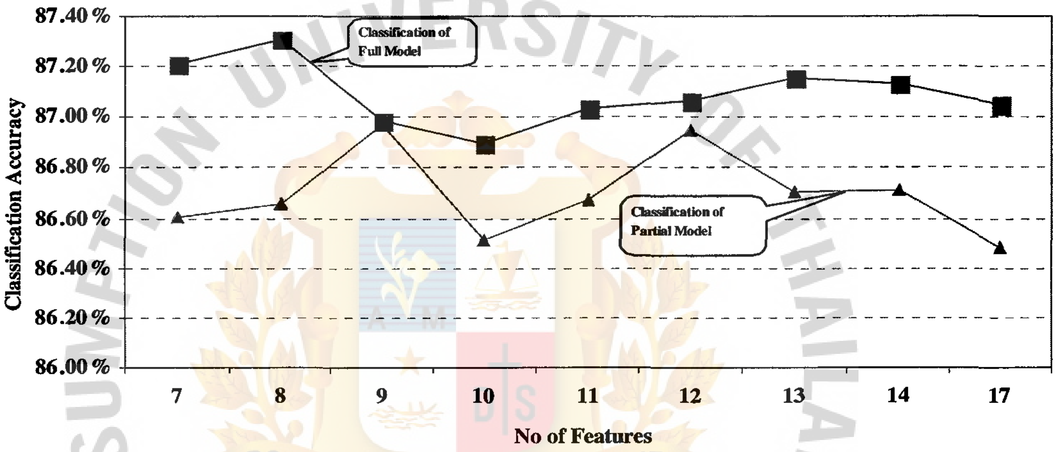

Figure 5.5: Number of Features Vs Classification Accuracy of Geez characters

From Figure 5.7, the classification accuracy for fixed features and constructive architecture is surpassed of variable features with fixed architectures. Furthermore, the number of iterations (epoch) to achieve the results for fixed feature are less than for fixed topology as it agrees with the general theory of constructive algorithm of neural networks algorithm. This leads to the conclusion that for both benchmark datasets, the training of constructive architecture with a given (fixed) feature is the best approach to design a forward integrated feature and architecture selection. In other words, the feature pillar comes first before architecture pillar.

Figure 5.8 describes the final number of hidden neurons against the classification

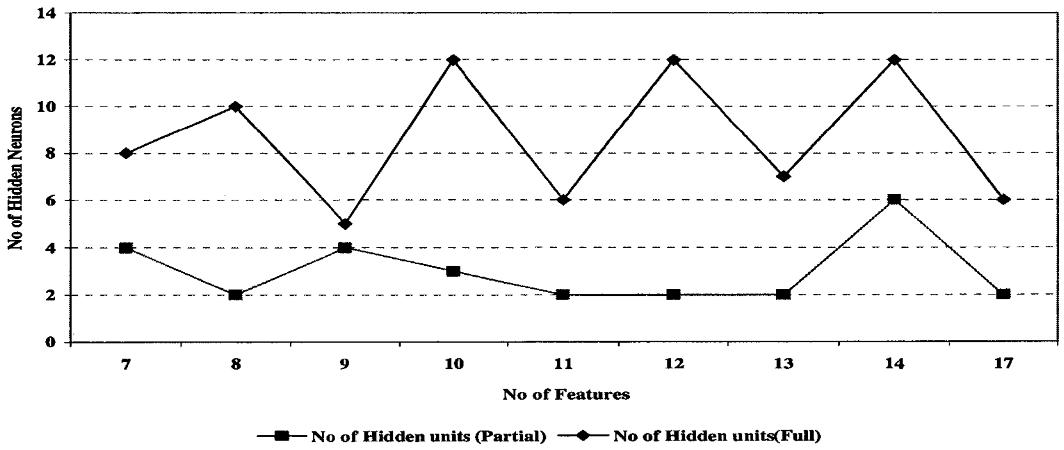

Figure 5.6: Number of Features Vs Number of hidden units of Geez characters

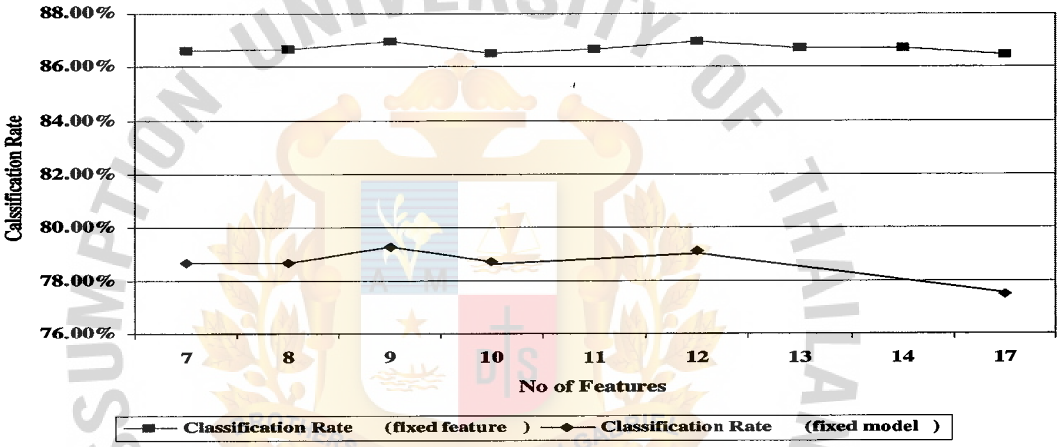

Figure 5.7: No of Features Vs Classification Rate for fixed feature and topology of Geez characters

rate of each relevant feature while the training proceed. Even though the number of hidden neurons is some how linear related for the first four relevant features, one can not conclude that the number of hidden neurons linear relates with classification accuracy since they are only the last hidden neurons when achieving the attained classification accuracy. This would be true when the training were tracing the performance error on training, testing and validating data sets.

Table 5.9: Highest accuracy based on MSEREG for acceptable model of Geez characters

Cover's Rule	No of Features	No of Hidden units	Epoch	Elapsed time (sec)	Classification Rate	No of weight parameters	Model Reduction	Training Error	Testing Error	Validation Error
180	7	4	85	9.47	86.66%	37	97.8%	7.6%	7.7%	7.5%
144	9	5	193	20.88	87.21%	56	96.5%	7.0%	7.2%	7.0%
103	13	6	139	15.54	87.03%	91	94.2%	7.5%	7.6%	7.4%
160	8	10	194	26.58	87.31%	101	93.8%	7.0%	7.1%	7.0%
80	17	7	225	24.98	87.08%	134	91.3%	6.9%	7.1%	6.9%
131	10	12	212	30.85	86.96%	145	90.8%	6.8%	7.0%	6.8%
120	11	14	175	27.45	86.82%	183	88.3%	7.4%	7.5%	7.4%
96	14	12	192	27.59	87.06%	193	87.5%	7.0%	7.2%	7.0%
111	12	20	272	51.42	87.12%	281	82.0%	7.4%	7.5%	7.3%

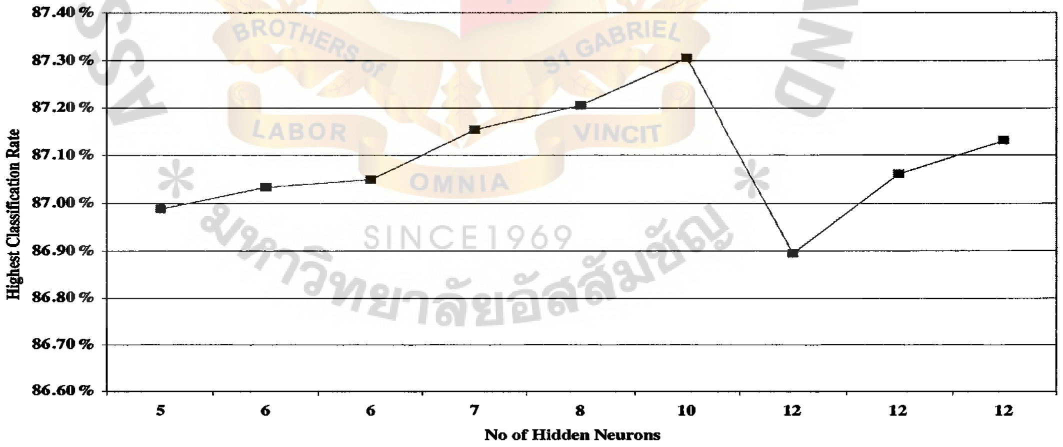

Figure 5.8: Number of Hidden Neurons Vs Highest Classification Rate of Geez characters

Chapter 6

Conclusions and Recommendations

6.1 Conclusions and Recommendations

Feature selection is very important aspect of solving the problem of pattern classification. Many collected datasets contain attributes that are redundant or irrelevant. The advantage of using only the relevant features of the data for classification are many. First, by reducing data-overfitting, a classifier with better predictive accuracy can be obtained. Second, by identifying the relevant features, the cost of future data collection can be reduced. Third, by excluding the irrelevant attributes, a simpler classifier can be obtained and the time required to classify new patterns can be reduced. Therefore, in this study, all the above truths were evinced or evident.

In this study, the selection of number of feature is dynamic one since unsupervised feature selector inherits such character where as for the works of [2], they ought to set the number of feature before hands. Thus, their methodology required a prior knowledge of the underlie problem to set the number which is most of the time ineffectual where as for our algorithm has not. The other advantage of this algorithm is that it uses minimal resources that give most advantageous accuracy. In other words, without the deterioration of the accuracy, one can achieve the desired results. Nevertheless there might be a better result for a given architecture and features with the tradeoff of computational costs if one considers accuracy is the first and sole objective. If one wants to have exhaustive "best" accuracy than a compromised result between speed and accuracy, this algorithm is less suitable. Setting values for this algorithm are very few and can be random since the adjustment to the optimal values will draw closer while the construction of the architecture in development. Besides, the classification rate for the selection of the architecture is flexible and set by user. It does not require any prior knowledge as other researcher did [63][79]. The effect of dimensions: high and medium have shown substantial effects. As a result, we conclude that high dimensions has a better position to fit the proposed algorithm since it can provide more uncorrelated feature dimensions for high classification accuracy. The training of constructive architecture with a given (fixed) feature is the best approach to design a forward integrated feature and architecture selection. In other words, the feature pillar comes first before architecture pillar.

The weak point of this algorithm for looking a better classification rate with ideal architecture is it required more resources and sometimes it fails to reach the goal. Therefore, with less maximal classification accuracy, the proposed algorithm is the best and ideal selection for the design of recognition systems for OCR problems.

6.2 Future Work

In feature selection theory, there are many theories along with advantages, disadvantages and suitability of each algorithm for searching best solution for different classes of problems. This study will proceed with a comprehensive investigation of various feature selection algorithms in conjunction with the new proposed algorithm so that it may give a firm conclusion for general principle about unified feature and architecture selection theory. Consequently, the study will recommend the algorithm for suitable types of classes.

Furthermore, future work will investigate the effect of two hidden layers and different connection strategies for newly added hidden layers and neurons with the achieved results of single hidden layer.

For long term, this study will deal with the combination of both constructive and pruning algorithms with the proposed algorithm. Besides, it will find the effect of genetic and fuzzy algorithms for the classification of characters with the proposed algorithm if there is a way to incorporate it.

Appendix A

Appendix - Geez Script

Geez characters are one of official scripts in East Africa. They have long history since the Birth of Jesus Christ. Since then they have been widely using for more than 60 million around the region. They have at least 6 derivatives from the root. All in all, they are 412 alphabets excluding numerals. Due to limited space, we present only few machine printed samples so as they can give you a clear insight how look like. For more information, one can see Unicode web page and search for Ethiopic fonts.

	120	121	122	123	124	125	126	127	128	129	12A	12B
0	ሀ 1200	ሐ 1210	ሠ 1220	ሰ 1230	ቀ 1240	ቐ 1250	በ 1260	ተ 1270	ኀ 1280	ነ 1290	አ 12A0	ኰ 12B0
1	ሁ 1201	ሑ 1211	ሡ 1221	ሱ 1231	ቁ 1241	ቑ 1251	ቡ 1261	ቱ 1271	ኁ 1281	ኑ 1291	ኡ 12A1	▨
2	ሂ 1202	ሒ 1212	ሢ 1222	ሲ 1232	ቂ 1242	ቒ 1252	ቢ 1262	ቲ 1272	ኂ 1282	ኒ 1292	ኢ 12A2	ኲ 12B2
3	ሃ 1203	ሓ 1213	ሣ 1223	ሳ 1233	ቃ 1243	ቓ 1253	ባ 1263	ታ 1273	ኃ 1283	ና 1293	ኣ 12A3	ኳ 12B3
4	ሄ 1204	ሔ 1214	ሤ 1224	ሴ 1234	ቄ 1244	ቔ 1254	ቤ 1264	ቴ 1274	ኄ 1284	ኔ 1294	ኤ 12A4	ኴ 12B4
5	ህ 1205	ሕ 1215	ሥ 1225	ስ 1235	ቅ 1245	ቕ 1255	ብ 1265	ት 1275	ኅ 1285	ን 1295	እ 12A5	ኵ 12B5
6	ሆ 1206	ሖ 1216	ሦ 1226	ሶ 1236	ቆ 1246	ቖ 1256	ቦ 1266	ቶ 1276	ኆ 1286	ኖ 1296	ኦ 12A6	▨
7	▨	~~ሗ 1217~~	ሧ 1227	ሷ 1237	▨	▨	ቧ 1267	ቷ 1277	▨	ኗ 1297	ኧ 12A7	▨
8	ለ 1208	መ 1218	ረ 1228	ሸ 1238	ቈ 1248	ቘ 1258	ቨ 1268	ቸ 1278	ኈ 1288	ኘ 1298	ከ 12A8	ኸ 12B8
9	ሉ 1209	ሙ 1219	ሩ 1229	ሹ 1239	▨	▨	ቩ 1269	ቹ 1279	▨	ኙ 1299	ኩ 12A9	ኹ 12B9
A	ሊ 120A	ሚ 121A	ሪ 122A	ሺ 123A	ቊ 124A	ቚ 125A	ቪ 126A	ቺ 127A	ኊ 128A	ኚ 129A	ኪ 12AA	ኺ 12BA
B	ላ 120B	ማ 121B	ራ 122B	ሻ 123B	ቋ 124B	ቛ 125B	ቫ 126B	ቻ 127B	ኋ 128B	ኛ 129B	ካ 12AB	ኻ 12BB
C	ሌ 120C	ሜ 121C	ሬ 122C	ሼ 123C	ቌ 124C	ቜ 125C	ቬ 126C	ቼ 127C	ኌ 128C	ኜ 129C	ኬ 12AC	ኼ 12BC
D	ል 120D	ም 121D	ር 122D	ሽ 123D	ቍ 124D	ቝ 125D	ቭ 126D	ች 127D	ኍ 128D	ኝ 129D	ክ 12AD	ኽ 12BD
E	ሎ 120E	ሞ 121E	ሮ 122E	ሾ 123E	▨	▨	ቮ 126E	ቾ 127E	▨	ኞ 129E	ኮ 12AE	ኾ 12BE
F	ሏ 120F	ሟ 121F	ሯ 122F	ሿ 123F	▨	▨	ቯ 126F	ቿ 127F	▨	ኟ 129F	▨	▨

	12C	12D	12E	12F	130	131	132	133	134	135	136	137
0	ኰ 12C0	ዐ 12D0	ዠ 12E0	ደ 12F0	ጀ 1300	ጐ 1310	ጠ 1320	ጰ 1330	ፀ 1340	ፐ 1350		፰ 1370
1		ዑ 12D1	ዡ 12E1	ዱ 12F1	ጁ 1301		ጡ 1321	ጱ 1331	ፁ 1341	ፑ 1351	፡ 1361	፱ 1371
2	ኲ 12C2	ዒ 12D2	ዢ 12E2	ዲ 12F2	ጂ 1302	ጒ 1312	ጢ 1322	ጲ 1332	ፂ 1342	ፒ 1352	። 1362	፲ 1372
3	ኳ 12C3	ዓ 12D3	ዣ 12E3	ዳ 12F3	ጃ 1303	ጓ 1313	ጣ 1323	ጳ 1333	ፃ 1343	ፓ 1353	፣ 1363	፳ 1373
4	ኴ 12C4	ዔ 12D4	ዤ 12E4	ዴ 12F4	ጄ 1304	ጔ 1314	ጤ 1324	ጴ 1334	ፄ 1344	ፔ 1354	፤ 1364	፴ 1374
5	ኵ 12C5	ዕ 12D5	ዥ 12E5	ድ 12F5	ጅ 1305	ጕ 1315	ጥ 1325	ጵ 1335	ፅ 1345	ፕ 1355	፥ 1365	፵ 1375
6		ዖ 12D6	ዦ 12E6	ዶ 12F6	ጆ 1306		ጦ 1326	ጶ 1336	ፆ 1346	ፖ 1356	፦ 1366	፶ 1376
7			ዧ 12E7	ዷ 12F7	ጇ 1307		ጧ 1327	ጷ 1337		ፗ 1357	፧ 1367	፷ 1377
8	ወ 12C8	ዘ 12D8	የ 12E8	ዸ 12F8	ገ 1308	ጘ 1318	ጨ 1328	ጸ 1338	ፈ 1348	ፘ 1358	፨ 1368	፸ 1378
9	ዉ 12C9	ዙ 12D9	ዩ 12E9	ዹ 12F9	ጉ 1309	ጙ 1319	ጩ 1329	ጹ 1339	ፉ 1349	ፙ 1359	፩ 1369	፹ 1379
A	ዊ 12CA	ዚ 12DA	ዪ 12EA	ዺ 12FA	ጊ 130A	ጚ 131A	ጪ 132A	ጺ 133A	ፊ 134A	ፚ 135A	፪ 136A	፺ 137A
B	ዋ 12CB	ዛ 12DB	ያ 12EB	ዻ 12FB	ጋ 130B	ጛ 131B	ጫ 132B	ጻ 133B	ፋ 134B		፫ 136B	፻ 137B
C	ዌ 12CC	ዜ 12DC	ዬ 12EC	ዼ 12FC	ጌ 130C	ጜ 131C	ጬ 132C	ጼ 133C	ፌ 134C		፬ 136C	፼ 137C
D	ው 12CD	ዝ 12DD	ይ 12ED	ዽ 12FD	ግ 130D	ጝ 131D	ጭ 132D	ጽ 133D	ፍ 134D		፭ 136D	
E	ዎ 12CE	ዞ 12DE	ዮ 12EE	ዾ 12FE	ጎ 130E	ጞ 131E	ጮ 132E	ጾ 133E	ፎ 134E		፮ 136E	
F		ዟ 12DF		ዿ 12FF			ጯ 132F	ጿ 133F	ፏ 134F		፯ 136F	

Bibliography

[1] Cole R. A., and L. Hirschman. "Workshop on Spoken Language Understanding," Technical Report CSE 92-014, Oregon Graduate Institute of Science and Technology, September 1992.

[2] Rose, P.E. "Flash of Genius," *Forbes*, Nov. 1998,pp. 98-104.

[3] Watanabe, S. Pattern recognition: Human and Mechanical. New York: Academic Press, 1972.

[4] Jain, A.K. *et al.* "Statistical Pattern Recognition: A Review," *IEEE Tran. Pattern Analysis and Machine Intelligence*, Vol. 22, No. 1, Jan 2000, pp. 4-37

[5] Fu, K.S., "A Step Towards Unification of Syntactic and Statistical Pattern Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 5, No. 2, March 1983, pp. 200-205.

[6] Robert, J. Schalkoff, Artificial Neural Networks, 1997.

[7] Jain, A.K. *et al.* "Artificial Neural Networks: A tutorial," *Computer,* March 1996, pp. 31-44.

[8] Kohonen, T. Self-Organizing Maps, Springer serier in Information Sciences, Vol. 30, 1995.

[9] Bishop, C.M., Neural Networks for Pattern Recognition, Oxford: Claredon Press, 1995.

[10] Jain, A.K. and B. Chandrasekaran, Dimensionality and Sample Size Consideration in Pattern Recognition Practice, Handbook of statistics, P.R. Krishnaiah and L.N. Kanal, eds. Vol. 2, Amsterdam: North-Holland, 1992.

[11] Raudys, S.J. and V. Pikelis, "On Dimensionality, Sample size, Classification Error, and Complexity of Classification Algorithms in pattern Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 2, 1980, pp. 243-251.

[12] Raudy, S.J. and A.K. Jain, "Small Sample Size Effects in Statistical Pattern Recognition: Recommendations for Practitioners," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 3, 1991, pp. 252-264.

[13] Baum, B. E. and D. Haussler, "What size net gives valid generalization?" *Neural Computa.,* Vol. 1989, pp. 151-160.

[14] Denker, J *et al.* "Large automatic learning, rule extraction, and generalization," *Complex System*, Vol. 1, 1987, pp. 877-922.

[15] Le Cun, Y., Generalization and network design strategies,in Connectionism in Perspective, R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, and L. Steels, Ed. Amesterdam: Elsevier, 1989, pp. 143-155.

[16] Chauvin, Y., "Generalization performance of overtrained backprobagation networks," in *Neural Networks – Proc. EURASIP Wkshp 1990,* L. B. Almeida and C.J. Wellekens, Eds. Berlin: Springer-Verlag, 1990, pp. 45-55.

[17] Towell, G. G. and *et al.* "Constructive induction in knowldge-based neural networks," in *Proc. 8th Int. Wkshp. Machine Learning*, L. A Birnbaum and G. C. Collions, Eds. San Mateo, CA: Morgan Kaufmann, 1991, pp. 213-217.

[18] Moody, J. "Prediction risk and architecture selection for neural networks," in *From Statistics to Neural networks: Theory and Pattern Recognition Applications, NATO ASI Series F*, V. Cherkassky, J. Friedman, and H. Wechsler, Eds., Vol. 136, New York: Springer-Verlag, 1994, pp. 147-165.

[19] Reed, R. "Pruning Algorithm–A survey," *IEEE Trans. Neural Networks,* Vol. 4, Sept 1993, pp. 740-747.

[20] Finnoff, W., F. Hergert, and H.G. Zimmermann, "Improving model selection by nonconvergent methods," *Neural Networks,* Vol. 6, 1993, pp. 771-783.

[21] Weigend, A.S. and *et al* "Generalization by weight-elimination with application to forecasting," in *Advances in Neural Information Processing Systems 3,* R.P. Lippmann, J.E. Moody, and D.S. Touretzky, Eds. San Mateo CA:Morgan Kaufmann, 1991, pp. 875-882.

[22] McKay, D. "A practical Bayesian framework for backpropagation networks," *Neural Computa.,* Vol. 4, 1991, pp. 448-472.

[23] Buntine, W. and A. Weigend, "Bayesian backpropagation," *Complex Systems,* Vol. 5, No. 6, 1991, pp. 603-643.

[24] Kwok, T.-Y. and D.-Y Yeung, "Constructive algorithms for structure learning in feedforward neural networks for regression problems," *IEEE Trans. Neural Networks,* Vol. 8, May 1997, pp. 630-645.

[25] Frean, M. "The upstart Algorithm, A method for constructive and training feedforward neural networks," *Neural Compta.,* Vol. 2, 1990.

[26] Hirose, Y. K. Yamashita, and S. Hijiya, "Backpropagation Algorithm which varies the number of hidden units," *Neural Networks,* Vol. 4, 1991, pp. 61-66.

[27] Fahlman, S. and C. Lebiere, "The cascade correlation learning architecture", Technical Report CMU-CS-90-100, Carbegie Mellon University, Pittsburgh, PA, August 1991.

[28] Mohraz, Karim and Peter Protzel, "A Flexible Neural Network Construction Algorithm" in *Proc. of the 4th European Symposium on artificial Neural Networks (ESANN'96)*, Brussels, 1996, pp. 111-116.

[29] Ash, T. "Dynamic node creation in backpropagation networks," *Connection Sci.,* Vol. 1, No. 4, ap89, pp. 630-645.

[30] Hwang, J.-N and *et al* "Regression modeling in backpropagation and project pursuit learning," *IEEE Trans. Neural Networks,* Vol. 5, May 1994, pp. 342-353.

[31] Setiono, R. nd L. Hui, "Use of a quasi-Newton method in a feedforward neural network construction algorithm," *IEEE Trans. Neural Networks,* Vol. 6, 1995, pp. 273-277.

[32] Bartlett, E. B. "Dynamic node architecture learning: An information theoretic approach," *Neural Networks,* Vol. 7, No. 1, 1994, pp. 129-140.

[33] Courrieu, P. "A convergent generator of neural networks," *Neural Networks,* Vol. 6, 1993, pp. 835-844.

[34] Kwok, T.-Y. and D.-Y Young, "Objective functions for training new hidden units in constructive neural networks," *IEEE Trans. Neural Networks,* Vol. 8, Sept. 1997, pp. 1131-1148.

[35] ....., "Experimental analysis of input weight freezing in constructive neural networks," in *Proc. IEEE Int Conf. Neural Networks,* San Francisco, CA, Mar 1993, pp. 511-516.

[36] Prechelt, L. "Investigation of the cascor family of learning algorithms," *Neural Networks,* Vol. 10, No. 5, 1997, pp. 885-896.

[37] Lay, S., J. Hwang and S. You, "Extensions to project pursuit learning networks with parmetric smoothers," in *Proc. IEEE Int. Conf. Neural Networks,* Orlando, Fl, Vol. 3, June 1994, pp. 1325-1330.

[38] Kwok, T.-Y. and D.-Y Yeung, "Bayesian regularization in constructive neural networks," in *Proc. Int. Conf. Neural Networks,* July 1996, pp. 557-562.

[39] Rumelhart, D. and J. McClelland, Parallel Distributed Processing: Explorations in Microstructure of Cognition, Vol 1/2. Cambridge, MA:MIT Press, 1996.

[40] Shin, Y. and J. Ghosh, "Ridge polynomial networks," *IEEE Trans. Neural Networks,* Vol. 6, May 1995, pp. 610-622.

[41] ....,"Adaptive regularization in a constructive cascade network," in *Proc. Int. Conf. Neural Inform. Processing,* Kitakyushu, Japan, Oct 1998, pp. 805-808.

[42] ....,"A constructive cascade network with adaptive regularization," in *Proc. Int. Work-Conf. Artificial Natural Neural Systems,* June 1990.

[43] MATLAB User's Guide (The MathWorks Inc., version 6.1)

[44] Hwang, J.-N and *et al* "The cascade correlation learning: a project pursuit learning perspective," *IEEE Trans. Neural Networks,* Vol. 7, Mar. 1996, pp. 278-289.

[45] Treadglod, Nicholas K. and Tamas D. Gedeon, "Exploring Constructive Cascade Networks," *IEEE Trans. Neural Networks,* Vol. 10, No. 6, Nov. 1999, pp. 1335-1350.

[46] Trier, O.D., A. K. Jain and T. Taxt, "Feature Extraction methods for Character Recognition –A Survey," *Pattern Recognition,* Vol. 29, No. 4, 1996, pp. 641-662.

[47] Khotanzad, A. and Y. H. Hong, "Invariant image recognition by Zernike moments," *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol. 12, No. 5, 1990, pp. 489-497.

[48] Kuhl, F.P. and C.R Giardina, "Ellipic Fourier feature of a closed contor," *Computer Vision, Graphic and Image Processing,* Vol. 18, 1982, pp. 236-258.

[49] Bello, M. G. "Enhanced training algorithms, and integrated training/architecture selection for multilayer percepton networks," *IEEE Trans. Neural Networks,* Vol. 3, No. 6, Nov. 1992.

[50] Belum, L.M. ans K.W Bauer, "Methods of determining input features for multilayer perceptons," *Neur. Computa.* Vol. 7, No. 2, 1995

[51] Maza la de, M. "SPLITnet dynamically adjusting the number of hidden units in a neural networks," *Artificial Neural Networks* T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, Eds. Amesterdam: North-Holland, 1991, pp. 647-651.

[52] Fogel, D. B. "An information criterion for optimal neural network selection," *IEEE Trans. Neral Networks,* Vol. 2, No. 5, Sept 1991.

[53] Gutierrez, J. W. and R. Grondin, "Estimating hidden unit number for two-layer perceptrons," in *Proc. Int. Joint Conf. Neural Networks,* Washington, DC, 1989, pp. 677-681.

[54] Huang , S. and Y. Huang, "Bounds on the number of hidden neurons in multilayer perceptrons," *IEEE Trans. Neural Networks,* Vol. 2, No. 1, 1991, pp. 47-55.

[55] Kung, S. Y. and J. N. Hwang, "An algebraic analysis for optimal hidden units size and learning rates in backpropagation learning," in *Int. Conf. Neural Netwoks,I,* 363370, San Diego, CA, July 24-27, 1988.

[56] Mozer, M.C. and P. Smolensky "Skeletonization: a technique for trimming the fat from a network via relevance assessement," in *Advanced Neural Network Information Systems I,* David S. Touretzky, Ed. Sa Mateo, CA: Morgan Kaufan.

[57] Priddy, K. L. and *et al.* "Bayesian selection of important features for feedforward neural networks," *Neurocomputing*, Vol. 5, No. 2 and 3, 1993.

[58] Redding, N. J. *et al.* "Higher order separability and minimal hidden unit fan-in," in *Artificial Neural Networks,* T. Kohonen, K Mäkisara, O. Simula, and J. Kangas, Eds. Amsterdam: North-Holland, 1991, pp. 25-30.

[59] Ruck, D. W. *et al.* "Feature selection using a multilayer perceptron," *J. Neur. Net. comp.,* Vol. 2, No. 2, Fall 1990, pp.40-48.

[60] Sartori, M. A. and P.J. Antsaklis, "A simple method to derive bounds on the size and to trin multilayer neural netwoks," *IEEE Trans. Neural Networks,* Vol. 2, July 1991, pp. 467-471.

[61] ...., "An additional hidden unit test for neglected nonlinearity in multilayer feedforward networks," in *IEEE INNS Int. J. Conf. Neur. Net II,* Washington, DC, June 18-22, 1989, pp. 451-455.

[62] ..., "Learning in artificial neural networks: A statistical perspective," *Neur. Comput. 1,* 1989, pp. 425-464.

[63] Steppe, Jean M., Kenneth W. Bauer, Jr., and Steven K. Rogers, "Integrated Feature and Architecture Selection," *IEEE Trans. Neural Networks,* Vol. 7, No. 7, July 1996, pp. 1007-1014.

[64] Cybenko, G. "Continuous valued neural networks with two layers are sufficient," Dep. Computer Sci., Tufts Univ. Tech. Rep., Mar 1988.

[65] Hornik, K. M. Stinchcombe and H. White, " Multilayer feedforward networks are universal approximators," Dept. Economics, University California, san Diego, manscriput, June 1988.

[66] Hecht-Nielsen, R."Theory of the backpropagation neural network," in *Proc. Int. Joint Conf. on Neural Networks, I,* New York, IEEE Press, June 1989, pp. 593-611.

[67] Mantel, N. "Why stepdown procedures in variable selection," *Technometrics,* Vol. 12, No. 2, Aug. 1970, pp. 621-625.

[68] Cover, T. M. "Geometrical and Statistical properties of systems of linear inequalities with applications in pattern recogniion," *IEEE Trans. Elec. Comp.,* Vol. EC-14, June 1965, pp. 326-334.

[69] Sontag, E. D. "Feedforward nets for interpolation and classification," *J. Comp. Systems. Sci.,* Vol. 45, 1992, pp. 20-48.

[70] Vapnik, V. N. and A. Y. Chervonenkis, "On the convergence of relative frequencies of events to their probabilities," *Theory Probab. Its Appl.* Vol. 16, No. 2, 1971, pp. 264-280.

[71] Gallant, A. R. Nonlinear Statistical Models. Nw York: Wiley, 1987

[72] Seber, G.A.F and C.J. Wild Nonlinear Regression. New York:Wiley, 1989.

[73] White, H. Artificial Neural Networks Approximation and Learning Theory. Cambridge, U.K:Blackwell, 1993.

[74] Le Cun, Y. *et al.* "Optimal brain damage," in *Neur. Info., Proc. Syst. II,* D.S. Touretzky, Ed. San Mateo, CA: Morgon Kaufamann, 1990, pp. 598-605.

[75] Prechelt, L. "Proben 1- A set of neural network benchmark problems and benchmarking rules," Fakultät für Informatik, Universität Kahlsruhe, Germany, Tech Rep. 21/94, 1994.

[76] Flexer, A. "Statistical evaluation of neural network experiments: Minimum requirements and current practices," Austrian Res. Inst. Artificial Intel., Tech. Rep. oe-fai-tr-95-16, 1995.

[77] Prechelt, L. "A quantitative study of experimental evaluation of neural network learning algorithms", *Neural Networks,* Vol. 9, 1996, pp. 457-462.

[78] Riedmiller, M. "Advanced supervised learning in multiplayer perceptions from back propagation to adaptive learning algorithm", *Int. J. Comput. Standards Interface,* Vol. 16, 1994, pp. 265-278.

[79] Mitra, P., C.A. Murthy and S. K. Pal, "Unsupervised Feature Selection Using Feature Similarity", *IEEE Trans. Pattern Anal. and Machine Intel.* Vol. 24, No. 3, March 2002, pp. 301-312.

[80] Treadgold, N. K. and T.D. Gedeon, "A cascade network algorithm employing progressive reprop," in *Proc. Int. Work-Conf. Artificial Natural Neural Syste.,* Lanzarote, June 1997, pp. 723-732.

[81] ....,"Extending casper: A regression survey," in *Proc. Int. Conf. Neural Infom. Processing,* Nov. 1997, pp. 310-313.