Recognition of Handwritten Special Characters and Gurmukhi Numerals using Artificial Neural Networks

A
DISSERTATION

Submitted in Partial Fulfilment of the Requirements for the Award of Degree of

Master of Technology
in
Computer Science and Applications

Submitted by

Karamjeet Kaur
Registration No. 651103005

Under the Supervision of

Dr. R. K. Sharma
Professor

SCHOOL OF MATHEMATICS AND COMPUTER APPLICATIONS

THAPAR UNIVERSITY
PATIALA – 147004

NOVEMBER 2014

CERTIFICATE

I hereby certify that the work which is being presented in the dissertation titled, " **Recognition of Handwritten Special Characters and Gurmukhi Numerals using Artificial Neural Networks**", in the partial fulfilment of the requirements for the award of degree of Master of Technology in **Computer Science and Applications** submitted in School of Mathematics and Computer Applications of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Dr. R.K. Sharma, Professor and refers other researchers' work which are duly listed in the reference section.

The matter presented in this dissertation has not been submitted for award of any other degree of this or any other University.

Karamjeet Kaur

(Karamjeet Kaur)
Reg. No.: 651103005

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

9.12.14

(Dr. R.K. Sharma)
Professor, SMCA
Thapar University, Patiala

Countersigned:

(Dr. Rajesh Kumar)
Head, SMCA

(Dr. S.S. Bhatia)
Dean of Academic Affairs

School of Mathematics and Computer Applications
Thapar University, Patiala

i

Abstract

Online handwriting recognition process is a quick and natural way of communication between computer and human beings. Handwritten character recognition has been one of the most important and challenging research areas in the field of pattern recognition. Handwritten character recognition (HCR) is the process of converting handwritten text into machine processable format. It contributes immensely to the advancement of an automation process and can improve the interface between human and machine in various applications. Variations in handwriting make it difficult and challenging to achieve a high degree of accuracy. This dissertation aims to develop an online system for the recognition of handwritten Gurmukhi numerals and special characters. Gurmukhi is the script of Punjabi language which is widely spoken across the globe. For the recognition of characters/numerals, neural networks have been applied. The study focuses on the recognition of handwritten and printed characters; and the results emerging from it are presented. Handwritten character recognition has been a popular research area for many years because of its various applications. These applications are quite useful for the organizations such as railway, embassies, etc. where both English and regional languages are used. Many forms and applications are filled in regional languages. Sometimes these forms are scanned directly. HCR is a process of automatic computer recognition of characters in optically scanned and digitized pages of text. It can be online or offline. In online numeral and special character recognition, data are captured during the writing process with the help of a special pen and an electronic interface. Offline documents are scanned images of pre-written text, generally, on sheet or paper. Offline number recognition is significantly different from online recognition, because here stoke information is not available.

For Gurmukhi numerals recognition, the accuracy achieved during training mode is 96.4%; and for testing mode it is 89.6%. For special characters recognition, training mode accuracy is 97.1%; while it is 92.0% for testing mode.

The present study has been divided into six chapters. A brief review of these chapters is given below.

Chapter-1 presents the different concepts of handwritten character recognition and also those of artificial neural networks.

Chapter-2 examines and evaluates the existing literature on the subject. The different neural network techniques used in HCR have been reviewed to identify the gap in research area.

Chapter-3 demonstrates how we can use the ANNs for approximating mathematical functions.

Chapter-4 explains the methods used for data collection and details on online handwritten Gurmukhi numeral recognition using ANN.

Chapter-5 provides the description of data collection for online handwritten special character recognition using ANN.

Chapter-6 concludes the work by developing a system for handwritten character/numeral recognition through neural networks in MATLAB. It also explains the scope for further research in the area.

Acknowledgments

First and foremost, I would like to thank the Almighty for providing me the faith, self-confidence and ability to complete the arduous task of this research work.

This work would not have been possible without the encouragement and able guidance of my supervisor, Dr. R. K. Sharma, Professor, SMCA, Thapar University, Patiala. Despite his busy schedules, he was kind enough to spare time to go through the rough drafts of this dissertation a number of times with constructive suggestions leading to substantial improvement in it. His enthusiasm and optimism made this experience both rewarding and enjoyable.

I am equally grateful to Dr. Rajesh Kumar (Associate Professor and Head, School of Mathematics and Computer Applications) for his great motivation and inspiration that triggered me for this dissertation work.

I shall be failing in my duty, if I don't express my gratitude to Dr. S. S. Bhatia, Senior Professor and Dean of Academic Affairs of the University, who made infrastructural provisions such as library and lab facilities accessible to the learners like me.

I am also highly thankful to the entire faculty and staff members of School of Mathematics and Computer Applications Department for their kind help, cooperation, love and affection which made my stay at the University memorable.

Last but not the least, I am beholden to my parents for their blessings and encouragement extended to me for the successful completion of this work. My friends also deserve the mention for their constant moral support.

(Karamjeet Kaur)
Regn. No. 651103005
M.Tech. (CSA)

TABLE OF CONTENTS

List of Abbreviations

NN	Neural Network
BPNNs	Back Pre-operational Neural Networks
ANN	Architecture of Neural Network
HCR	Handwritten Character Recognition
EBP	Error Back Propagation
ED	Euclidean Distance
OCR	Optical Characters Recognition

LIST OF FIGURES

List of Tables

Chapter-1
INTRODUCTION

1.1 Definition

Neural Network is the biological structure inspired by the working of human nervous system. Neural Network is being applied widely on different application areas due to its learning ability, *i.e.*, capability to extract rules and learn from the data and create a network model which can be used for classification, pattern recognition and forecasting on the data. Most promising feature of the Neural network that other classification techniques do not possess is that it helps to simulate the network and create a model which can be used further and applied on the new data which was not previously exposed to the network (Yusuf, 2012). Dr. Robert Hecht-Nielsen defines a neural network as a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs (Maureen and Caudill, 1990). An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the working of biological nervous systems. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like human beings, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well (Werbos, 1994). In information technology, a neural network is a system of programs and data structures that approximates the operation of the human brain. A neural network usually involves a large number of processors operating in parallel, each with its own small sphere of knowledge and access to data in its local memory (Jun *et al.*, 2005).

A neural network consists of four main parts:

1. Processing units $\{u_j\}$, where each u_j has a certain activation level $a_{j(t)}$ at any point in time.
2. Weighted interconnections between the various processing units which determine how the activation of one unit leads to input for another unit.
3. An activation rule which acts on the set of input signals at a unit to produce a new output signal, or activation.
4. Optionally, a learning rule that specifies how to adjust the weights for a given input/output pair.

A processing unit u_j takes a number of input signals, say a_{1j} , a_{2j}, ..., a_{nj} with corresponding weights w_{1j} , w_{2j} ,...,w_{nj} , respectively. The net input to u_j given by:

$$net_j = SUM (w_{ij} * a_{ij})$$

The new state of activation of u_j given by:

$$a_j(t+1) = f(a_{j(t)} , net_j),$$

Where F is the activation rule and $a_{j(t)}$ is the activation of u_j at time t. The output signal o_j of unit u_j is a function of the new state of activation of u_j :

$$O_j(t+1) = f_j(aj(t+1)).$$

One of the most important features of a neural network is its ability to adapt to new environments. Therefore, learning algorithms are critical to the study of neural networks.

1.2 Basics of Neural Networks

Neural Networks are typically organized in layers. Layers are made up of a number of interconnected 'nodes' which contain an 'activation function'. Patterns are presented to the network via the 'input layer', which communicates to one or more 'hidden layers' where the actual processing is done via a system of weighted 'connections'. The hidden layers then link to an 'output layer' where the answer is output as shown in the Figure 1.1 below.

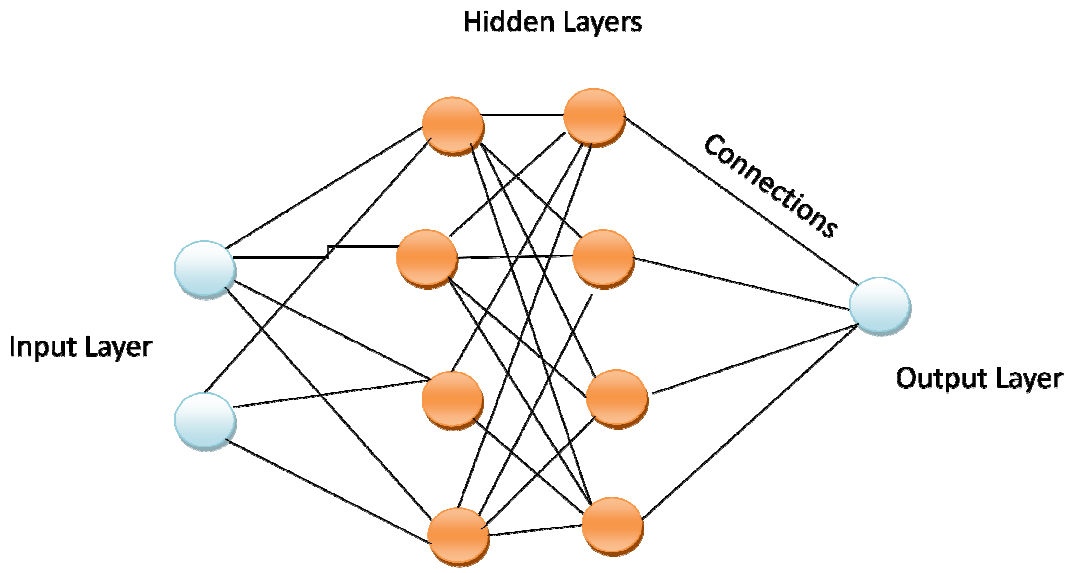


Fig. 1.1: Layers of Neural Networks

Most ANNs contain some form of 'learning rule' which modifies the weights of the connections according to the input patterns that it is presented with. In a sense, ANNs learn by example as do their biological counterparts; a child learns to recognize dogs from examples of dogs.

Although there are many different kinds of learning rules used by neural networks, this demonstration is concerned only with one; the delta rule. The delta rule is often utilized by the most common class of ANNs called 'back-propagation neural networks' (BPNNs). Back propagation is an abbreviation for the backwards propagation of error.

With the delta rule, as with other types of back propagation, 'learning' is a supervised process that occurs with each cycle or 'epoch' (*i.e.* each time the network is presented with a new input pattern) through a forward activation flow of outputs, and the backwards error propagation of weight adjustments. More simply, when a neural network is initially presented with a pattern it makes a random 'guess' as to what it might be. It then sees how far its answer was from the actual one and makes an appropriate adjustment to its connection weights (Werbos, 1994). More graphically, the process looks something like this:

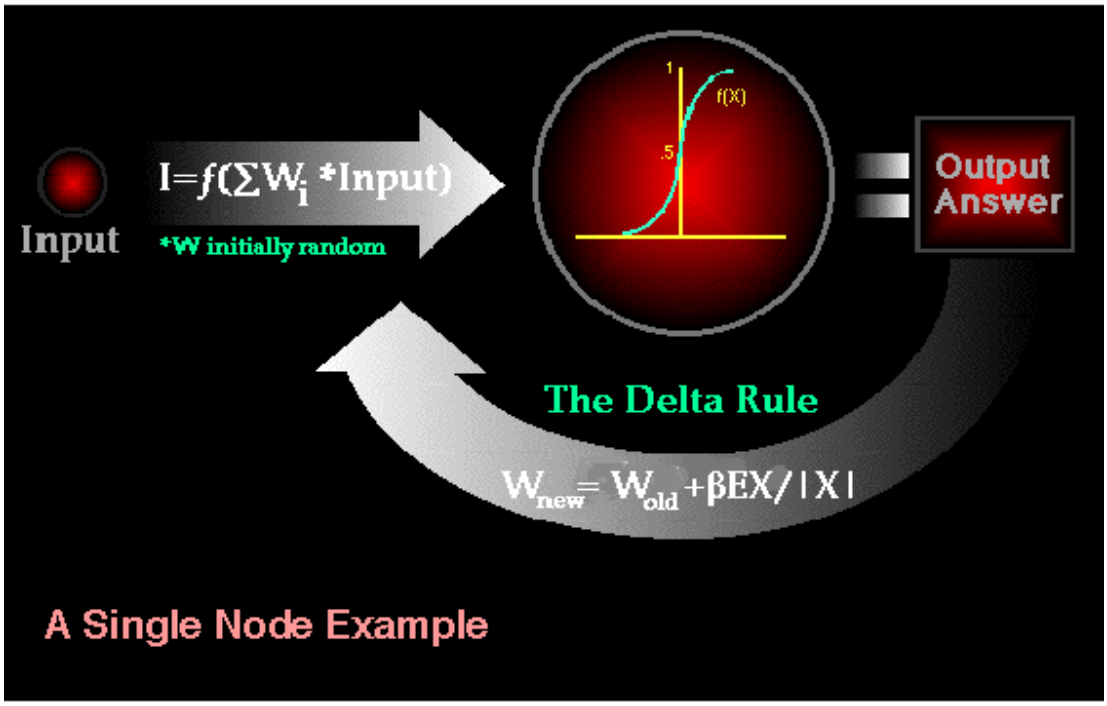


Fig. 1.2: A Single Node (Werbos, 1994)

1.3 Architecture of Neural Networks

1.3.1 Feed-forward Networks:- Feed-forward ANNs allow signals to travel one-way only; from input to output. There is no feedback (loops), *i.e.*, the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straightforward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organization is also referred to as bottom-up or top-down.

1.3.2 Feedback Networks:- Such networks can have signals travelling in both directions by introducing loops in the network. These networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organizations. The schematic diagram of artificial neural network and architecture of the feed-forward network with one hidden layer is given below:

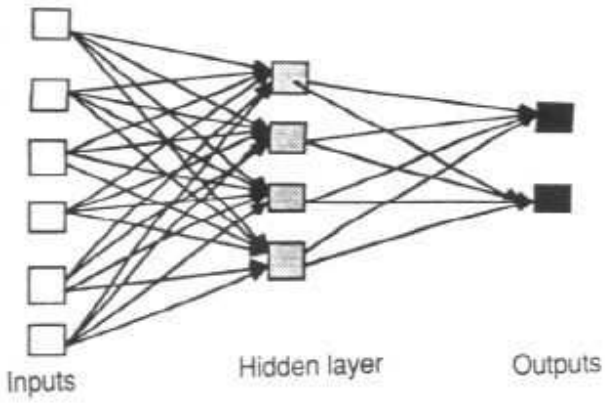


Fig. 1.3: An Example of a Simple Feed-forward Network (Werbos, 1994)

1.3.3 Network Layers

The commonest type of artificial neural network consists of three groups, or layers, of units: a layer of "input" units is connected to a layer of "hidden" units, which is connected to a layer of "output" units.

- The activity of the input units represents the raw information that is fed into the network.
- The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.
- The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units.

This simple type of network is interesting because the hidden units are free to construct their own representations of the input. The weights between the input and hidden units determine when each hidden unit is active, and so by modifying these weights, a hidden unit can choose what it represents.

We also distinguish single-layer and multi-layer architectures. The single-layer organization, in which all units are connected to one another, constitutes the most general case and is of more potential computational power than hierarchically structured multi-layer organizations. In multi-layer networks, units are often numbered by layer, instead of following a global numbering.

1.3.4 Perceptrons

The most influential work on neural nets in the '60s went under the heading of 'perceptrons', a term coined by Frank Rosenblatt. The perceptron turns out to be an MCP model (neuron with weighted inputs) with some additional, fixed, pre-processing. Units labelled A_1, A_2, A_j, A_p are called association units and their task is to extract specific, localized features from the input images. Perceptrons mimic the basic idea behind the mammalian visual system. They were mainly used in pattern recognition even though their capabilities extended a lot more.

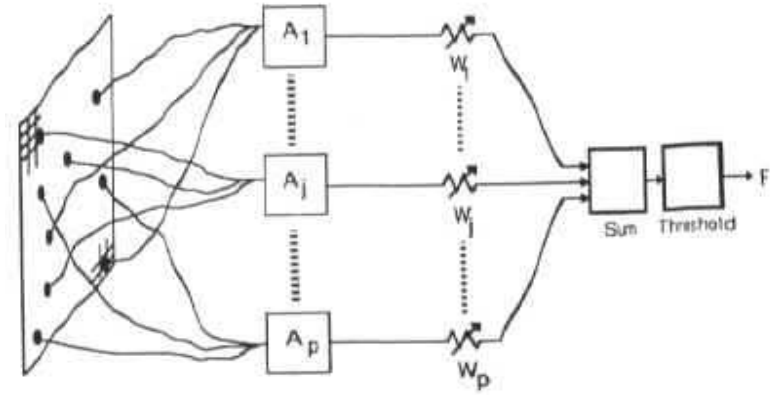


Fig. 1.4: Perceptrons

In 1969, Minsky and Papert wrote a book in which they described the limitations of single layer perceptrons (Ritu *et al.,* 2013) . The impact that the book had was tremendous and caused a lot of neural network researchers to lose their interest. The book was very well written and showed mathematically that single layer perceptrons could not do some basic pattern recognition operations like determining the parity of a shape or determining whether a shape is connected or not. What they did not realize, until the '80s is that given the appropriate training, multilevel perceptrons can do these operations.

1.4 Applications of Neural Networks

- **Character Recognition:** The idea of character recognition has become very important as handheld devices like the tablets are becoming increasingly popular. Neural networks can be used to recognize handwritten characters.

- **Image Compression:** Neural networks can receive and process vast amounts of information at once, making them useful in image compression. With the Internet explosion and more sites using more images on their sites, using neural networks for image compression is worth a look.
- **Stock Market Prediction:** The day-to-day business of the stock market is extremely complicated. Many factors weigh in whether a given stock will go up or down on any given day. Since neural networks can examine a lot of information quickly and sort it all out, they can be used to predict stock prices.
- **Travelling Salesman's Problem:** Interestingly enough, neural networks can solve the travelling salesman problem, but only to a certain degree of approximation.
- **Medicine, Electronic Nose, Security, and Loan Applications:** These are some applications that are in their proof-of-concept stage, with the acceptation of a neural network that will decide whether or not to grant a loan, something that has already been used more successfully than many humans.
- **Miscellaneous Applications:** These are some very interesting (albeit at times a little absurd) applications of neural networks.
- **Modelling and Diagnosing the Cardiovascular System:** Neural Networks are used experimentally to model the human cardiovascular system. Diagnosis can be achieved by building a model of the cardiovascular system of an individual and comparing it with the real time physiological measurements taken from the patient. A model of an individual's cardiovascular system must mimic the relationship among physiological variables at different physical activity levels.
- **Instant Physician:** An application developed in the mid-1980s called the "instant physician" trained an auto associative memory neural network to store a large number of medical records, each of which includes information on symptoms, diagnosis, and treatment for a particular case.
- **Neural Networks in Business:** Business is a diversified field with several general areas of specialization such as accounting or financial analysis. Almost any neural network application would fit into one business area or financial analysis.

There is some potential for using neural networks for business purposes, including resource allocation and scheduling.

- **Credit Evaluation:** The HNC Company, founded by Robert Hecht-Nielsen, has developed several neural network applications. One of them is the Credit Scoring system which increases the profitability of the existing model up to 27% (Jun *et al.*, 2005).

1.5 Handwritten Character Recognition

Handwritten character recognition is an important and active field of research in pattern recognition and image processing. Handwritten character recognition, abbreviated as HCR, is the process of converting handwritten text into machine processable format. Handwriting recognition is the ability of a computer to receive and interpret handwritten input from sources such as paper documents, photographs, touch-screens and other devices. Handwriting recognition system is the activity through which users transmit the handwritten data into the computer through the sources of scanner, touch screens and other devices, where computer interprets that input into its language. The mode of such input can be paper documents, photograph, etc. As discussed earlier that handwriting recognition system deals with the acquiring of data through various methods, so on this acquiring only the classification of handwriting recognition system depends on using online character recognition systems and offline character recognition systems. As it includes a set of characters, the numerous writing style variations and the constraints on the writing have a great influence on recognition methods.

The classification of handwriting recognition can be done in the following two major categories:

1.5.1 Offline Handwriting Recognition: Under such handwriting recognition, the writing is available as an image. It can be done with the help of a scanner which captures the writing optically.

1.5.2 Online Handwriting Recognition: Under such handwriting recognition, the two-dimensional coordinates of successive points are represented as a function of time and the order of strokes made by the writer are also available. The online methods have been shown

to be superior to their offline counterparts in recognizing handwritten characters due to the temporal information available with the former.

Nowadays, for devices like personal digital assistant or tablets, online handwriting recognition has become a useful feature. With this feature the demand and use of tablets has increased manifold. The presence of online handwriting recognizer for Devnagari, Gurmukhi, Bangla and other Asian scripts shall provide a natural way of communication between users and computers and it will increase the usage of personal digital assistant or tablet PCs in Indian languages. Also, some of the Asian scripts such as Devanagiri, Gurmukhi, Bangla and Tamil share many similarities and therefore advances made for one script with respect to online handwriting recognition could be useful for other such similar scripts (Sharma *et al.,* 2009).

1.6 Process to Recognize a Handwritten Character or Digit in MATLAB

Fig. 1.5 represents the steps involved in recognition of a character entered by a user by hand, *i.e.,* handwritten characters.

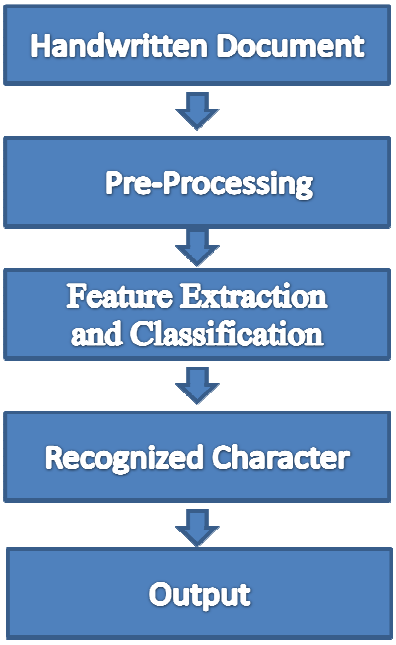


Fig. 1.5: Steps Involved in Recognition of a Handwritten Character

1.6.1 Data Base: In this process, handwritten characters/numerals are taken as an input from different users by writing with a special pen on an electronic surface (*i.e.* online characters/ numerals).

1.6.2 Pre-processing: Pre-processing is a series of operations performed on handwritten characters/numerals. It is the initial stage of character/numeral recognition. Pre-processing, as the name expresses, involves the processing that is done before the character/numeral is recognized, *i.e.*, preparing the character in the form suitable for recognition process. In HCR, typical pre-processing operations are performed through normalization.

- Normalization is one of the widely used techniques for pre-processing. It involves resizing the image to a particular size. As the character written by the user can vary in size, it can be very difficult to recognize the character/numeral if it is to be recognized as such.

1.6.3 Feature Extraction: Each character/numeral has some features which play an important role in pattern recognition. Every Indian script has its own features. Feature extraction describes the relevant shape information contained in a pattern so that the task of classifying the pattern is made easy by a formal procedure.

1.6.4 Classification: Classification stage uses the features extracted in the feature extraction stage for deciding the class membership. Classification phase is the decision-making phase of an HCR Engine. In HCR system, MLP classifiers are constructed and then these classifiers are integrated in a hidden layer. HCR system is the combination of following two steps:

Step 1: Each MLP network classifier is constructed and trained. Some Punjabi characters or numerals are formed by combining two or more characters or numerals. So, firstly, the structure of each MLP classifier is determined.

Step 2: In this step, the integrated MLP network classifiers which exist in the hidden layer are trained using the newff (feed forward neural network) algorithm, and we train until mean square error between the network output and desired output is less than 0.05.

Chapter-2
Literature Survey

2.1 Literature Survey

In this chapter, an attempt has been made to present an overview of various aspects and issues of the current study through the review of available literature on the subject. It is a modest effort to identify the gap that exists in this research area. A lot of research has been done on Feature extraction techniques by using neural networks, learning system for on-line character recognition, word extraction from on-line handwritten text lines, etc. The table given below provides a brief description of the existing literature in the subject area under study.

Table 2.1: List of Selected Research Papers Undertaken for Review

Author Name	Title	Year
Bontempi, B.; and Marcelli, A.	A Genetic Learning System for On-line Character Recognition	1994
Duneau, L.; and Dorizzi, B.	Online Cursive Script Recognition A System that Adapts to an Unknown User	1994
Goh, W. L.; Mittal, D. P.; and Babri, H.	An Artifical Neural Networks Approach to Hand-writing Recognition	1997
Verma, B.	A Feature Extraction Technique in Conjunction with Neural Network to Classify Cursive Segmented Handwritten Characters	1998
Blumenstein, M.; Verma, B.; and Basli, H.	A Novel Feature Extraction Technique for the Recognition of Segmented Handwritten Characters	2003
Liu, X. Y.; and Blumenstein, M.	Experimental Analysis of the Modified Direction Feature for Cursive Character Recognition	2004
Funanda, A.; Muramatsu, D.; and Matsumoto, T.	The Reduction of Memory and the Improvement of Recognition Rate for HMM On-line Handwriting Recognition	2004

Table 2.1 (Contd.)

Mahmud, J.	An Intelligent Feature Analyzer for Handwritten Character Recognition	2005
Liwicki, M.; Scherz, M.; and Bunke, H.	Word Extraction from On-line Handwritten Text Lines	2006
Sharma, A.; Young, D. L.; and Wan, Y. C.	High Accuracy Human Activity Monitoring Using Neural Network	2008
Pal, U.; Wakabayashi, T.; and Kimura, F.	Comparative Study of Devnagari and written Character Recognition Using Different Features and Classifiers	2009
Sharma, A.; Kumar, R.; and Sharma, R. K.	Rearrangement of Recognized Strokes in Online Handwritten Gurmukhi Words Recognition	2009
Ahranjany, S. S.; Razzazi, F.; and Ghassemian, M. H.	A Very High Accuracy Handwritten Character Recognition System for Farsi/Arabic Digits Using Convolutional Neural Networks	2010
Rajashekararadhya, S. V.; and Ranjan, P. V.	The Zone-based Projection Distance Feature Extraction Method for Handwritten Numeral/Mixed Numerals Recognition of Indian Scripts	2010
Rajput, G. G.; and Mali, S. M.	Fourier Descriptor Based Isolated Marathi Handwritten Numeral Recognition	2010
Gumah, M. E.; Schneider, E.; and Aburas, A.	Handwriting Recognition System Using Fast Wavelets Transform	2010
Tsuruoka, S. ; Fadzil, Mohd.T.; Kawanaka, T.; and Yasuji, M.	Personal Dictionaries for Handwritten Character Recognition	2010
Eraqi, M.; and Sherif, A.	An On-Line Arabic Handwriting Recognition System	2011
Dhir, R.; Singh, K.; and Rani, R.	Handwritten Gurmukhi Numeral Recognition Using Different Feature Sets	2011

Table 2.1 (Contd.)

Jaremsri, M. L.; and Imprasert, Y.	Thai Handwritten Character Recognition Using Heuristic Rules Hybrid with Neural Network	2011
Kumar, M.; Jindal, M. K.; and Sharma, R. K.	Classification of Characters and Grading Writers in Offline Handwritten Gurmukhi Script	2011
Sanmorino, A.; and Yazid, S.	A Survey for Handwritten Signature Verification	2012
Dubey, R.; and Agrawal, D. K.	Comparative Analysis of Off-line Signature Recognition	2012
Kaur, A.; and Arora, M.	Neural Network based Numerical Digits Recognization Using NNT in Matlab	2013
Shokoohi, Z.; Hormat, A. M.; Mahmoudi, F.; and Badalabadi, H.	Persian Handwritten Numeral Recognition Using Complex Neural Network and Non-linear Feature Extraction	2013

Bontempi and Marcelli (1994) presented a method based on a genetic algorithm used as the engine of a learning system to produce prototypes of the characters, and on a string matcher to perform the classification. The learning mechanism, provided by a genetic algorithm, allowed the system to have both a writer independent core and an adaptation scheme to finely tune the recognizer to the writer style. The system was tested over 10 subjects, different from the ones used to obtain the training set, who were required to write, in different times, two sets of numerals, uppercase and lowercase letters. A recognition rate of 83.4% was obtained, with a reject rate of 14.7% and an error rate of 1.9% after adapting writer, the system was able to exhibit a recognition rate of 94.8% on that writer. By running again the second experiment with the new system, a recognition rate of 92.7% was obtained, with a rejection rate of 6.5% and an error rate of 0.8%, showing that by adapting to that writer, the system increased its overall performance.

Duneau and Dorizzi (1994) developed a system dedicated to the recognition of English cursive words drawn on a digitizing tablet. This system follows an analytical approach in the sense that it tries to localize the letters of the word to be recognized. The system in a

monoscriptor context has recognition rate min.-average-max. of 89.8-93.2-95.6%, respectively; and in the case of multi-scripter context rate was 93.5% for 15432 prototypes and 91.4% for 5625 prototypes, but the system is too slow and heavy during multi-scripter context, and in the case of mono-scriptor recognition rate min.-average-max. of 60.8-75.8-92.2%, respectively.

Goh *et al*. (1997) used Artificial Neural Network in handwriting recognition. This approach was very suitable for handwritten character recognition as it provides fast feature extraction and classification. Error Back Propagation (EBP) algorithm networks of relatively small sizes which can be trained in a reasonable short time were used. The emphasis has mostly been on developing a writer-independent handwritten character recognition system.

The system had the following capabilities:

- Allows the user to write in free-hand continuously forming sentences and paragraphs, interactively converting each character written to standard font. The characters that the system can recognize include '0' to **'9',** 'A' to 'Z'' '.' and ','.
- Allows the user to save a written document to file or print out **as** hardcopy.
- Allows system customization to suit individual handwriting so as to achieve high recognition accuracy through a built-in user friendly training system.
- Recognition accuracy is above 97% and response speed is on an average 1 second per character.

The authors designed handwriting recognition system and developed to recognize 38 classes of characters using the error back propagation algorithm. Handwriting recognition is a difficult task; and there is currently no single solution that matches all the different demands. The ideal goal of designing a handwriting recognition method with 100% accuracy is not easily achievable because even human beings are not able to recognize all handwritten text without doubt. The handwriting recognizer developed has good practical potential as recognition rates of 97% and higher are acceptable to most users.

Verma (1998) proposed a new technique of feature extraction in conjunction with a neural network to classify segmented cursive handwritten characters. A heuristic and neural

network-based algorithm was used to segment the characters. After segmentation the proposed technique was applied to segmented and pre-processed characters. Handwritten character recognition system can be very useful in many real world applications such as postal addresses recognition, document analysis and other applications. A good feature extraction technique can extract important features and improves the recognition rates. Cursive handwritten character recognition is independent of rotation and converts large two-dimensional character mat rice into a small one-dimensional vector with global features. The presented technique proved very effective in recognizing difficult handwriting regardless of orientation. A small one-dimensional vector from a large scale character metrics which the researcher obtained using the proposed techniques, avoids the computer memory problems for a larger database, long training time in learning and recognition processes.

Blumenstein *et al.* (2003) proposed a novel feature extraction technique for the recognition of segmented handwritten characters. The authors stated that high accuracy character recognition techniques can provide useful information for segmentation-based handwritten word recognition systems. Two neural architectures along with two different feature extraction techniques were investigated. The process of extraction was only employed for those characters obtained from handwritten words. The technique followed by the authors first proceeded to sequentially locate all non-cursive/printed character components through the use of character component analysis. Next, x-coordinates (vertical segmentations) for each connected character component (identified by a heuristic) were used to define the vertical boundaries of each character matrix. The direction-based feature extraction technique was compared to another popular technique in the literature using two neural classification schemes.

Liu and Blumenstein (2004) described and analyzed the performance of a structural feature extraction technique for the recognition of segmented/cursive characters. The Modified Direction Feature (MDF) extraction technique builds upon a previous technique proposed by the authors that extracts direction information from the structure of character contours. The modified direction technique is based on transition feature and direction

feature techniques. The main difference was in the way the feature vector was created. For MDF feature vector creation was based on the calculation of transition features from the background to foreground pixels in the vertical and horizontal directions. However, in MDF, aside from calculating the Location of Transitions (LTs), the Direction Transition (DT) values at a particular location are also stored. Therefore, for each transition, a pair of values such as [LT, DT] is recorded. A detailed analysis of the incorrectly classified characters provided by one of the MDF neural network configurations revealed four main character classes consistently giving misclassifications. These four character classes were then further scrutinized to determine the source of confusion.

Funanda *et al.* (2004) used HMM in online handwriting recognition that reduces memory usage and further improves the recognition rate. Self-Organizing Map (SOM) density tying reduced the dictionary size to $1/7$ of the original size with a recognition rate of 90.45%, only slightly less than the original recognition rate of 91.51%. Their additional feature increased recognition capability to 91.34%.

Mahmud (2005) stated the development of an efficient feature analyzer for handwritten character recognition. Feature extraction can reduce large domain of feature space and extract invariable information. Feature extraction has been viewed from multi-dimensional perspective. To cope with the fuzziness of the recognition problem, a nonlinear classifier based on back propagation algorithm was used for classification. Proposed intelligent feature analysis has been successfully implemented in the area of handwritten character recognition. To improve the generalization ability, ensemble-based approach has been used. Speed of convergence has been effectively enhanced by resilient back propagation algorithm. To limit the problem within manageable size, only character images from a particular alphabet were used for training and recognition. But the feature analysis model is not language dependent, it is entirely possible to train and test using characters from other alphabets. This feature analysis model was capable of dealing with large image databases.

Liwicki *et al.* (2006) investigated the word extraction task in on-line recognition of cursively handwritten text lines. The authors proposed a method which is based on the assumption that the size of gaps between consecutive words may considerably vary, but

humans usually leave more whitespace between two consecutive words than between two connected components that belong to the same word. They used several metrics known from off-line word segmentation for measuring the distances between two adjacent components. After that authors applied different procedures to get initial threshold for segmentation. The thresholds are based on the following statistics.

- Median White Run-length (MWR): It is the median of the set of white run-lengths between the two lines.
- Average White Run-length (AWR): It is the median of the number of white pixels in a row divided by the median number of black-white transitions in a row.

Results showed that there was a significant potential in increasing the segmentation performance over simple heuristic methods. The authors could increase the word extraction rate from 85.36% in the baseline method where a fixed threshold is used to 85.95% by computing the initial threshold with the AWR method. By applying the new word segmentation algorithm the word extraction rate could be further increased to 86.58%, which is statistically significant.

Sharma *et al.* (2008), in their research work, presented the designing of a neural network for the classification of human activity. A triaxial accelerometer sensor, housed in a chest worn sensor unit, has been used for capturing the acceleration of the movements associated. All the three axis acceleration data were collected at a base station PC via a CC2420 2.4GHz ISM band radio (zigbee wireless compliant), processed and classified using MATLAB. A neural network approach for classification was used with an eye on theoretical and empirical facts. It provided a detailed description of the designing steps for the classification of human body acceleration data. A 4-layer back propagation neural network, with Levenberg-marquardt algorithm for training, showed best performance among the other neural network training algorithms.

Pal *et al.* (2009) presented a study on handwritten character recognition for Devnagari. Devnagari is the most popular script in India. The authors used the offline handwritten character recognition process. It is the Indian national language and Hindi is the third most

popular language in the world. The alphabet of the modern Devnagari script consists of 14 vowels and 33 consonants. These characters are called basic characters. Hindi is written in Devnagari script. Nepali, Sanskrit and Marathi are also written in Devnagari script. They presented a study of Devnagari handwritten character recognition by using the twelve different classifiers and four different features computed from gradient and curvature information of the binary as well as gray-scale images. Classifiers like Projection distance, Subspace method, Linear discriminate function, Support vector machines, Modified quadratic discriminate function, Mirror image learning, Euclidean distance, Nearest neighbor, k-Nearest neighbor have been considered. The gradient and curvature features computed from grey images classifier gave an accuracy of 94.74% and 95.19%, and binary images gave an accuracy of 94.74% and 95.09%.The Euclidean Distance (ED) showed the lowest results (77.89%) among the classifiers. They observed that curvature feature provided higher results than gradient features in all the classifiers except NN and k-NN. NN and k-NN classifiers showed slightly lower results in curvature features than gradient features.

Sharma *et al.* (2009) presented a technique to recognize online handwritten Gurmukhi words. They proposed a new step as rearrangement and recognition strokes in online handwriting recognition procedure. The rearrangement of recognized strokes includes: strokes identification as dependent and major dependent strokes; the rearrangement of strokes with respect to their positions; and the combination of strokes to recognize character. The authors stated that Gurmukhi is the script of Punjabi language which is spoken across globe, and it shares many similarities to other Asian scripts such as Devanagiri and Bangle. Gurmukhi approach is written in left-to-right direction and in top-down approach. Gurmukhi script is divided into three horizontal zones. The upper zone denotes region above head line and middle zone represents the area below the head line whereas consonants and some sub-parts of vowels are present. The lower zone represents the area below middle zone where some vowels and certain half characters lie in the foot of consonants.

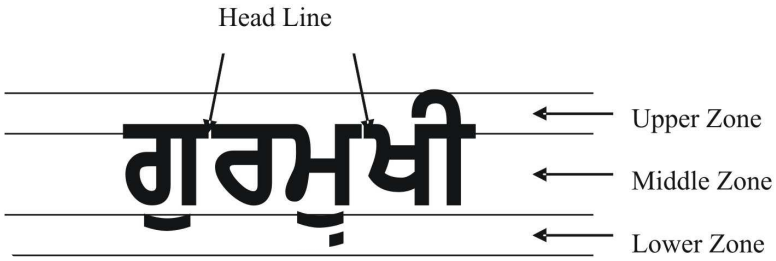


Fig. 2.1: Upper, Middle and Lower Zones in Gurmukhi Script

An overall recognition rate of 81.02% in online handwritten cursive handwriting for a set of 2576 Gurmukhi dictionary words was achieved by them.

Ahranjany *et al.* (2010) proposed a new method for handwritten Farsi/Arabic digits by fusing the recognition results of a number of convolution neural networks with gradient descent training algorithm. Convolution Neural networks are a type of neural networks that are biologically inspired from human visual system which combines feature extraction and classification stages. Their research paper consisted of two main contributions. The first one is automatic extraction of input pattern's features by using a CNN for Farsi digits and second one is fusing the results of boosted classifiers to compensate the recognizers' errors. The authors used IFH-CDB test database. The main advantage of their network was automatic feature extraction and classification in one integrated system.

Rajashekararadhya and Ranjan (2010) stated that handwriting recognition has been a challenging task in image processing and pattern recognition. Feature extraction method was probably the most effective method in achieving high recognition performance. The authors proposed a zone-based feature extraction algorithm scheme for the recognition of off-line handwritten numerals of south-Indian scripts. The character centroid is computed and numeral image (50×50) is further divided into 25 equal zones (10×10). The distance from character centroid to the pixels present in zone was computed. This procedure was sequentially repeated for all the zone/grid/box columns present in the zone. Similarly, again the character centroid was computed and image was further divided into 50 equal zones and distance from pixels present in zone was computed. This entire procedure was sequentially repeated for the entire zone present in the numeral image. There could be some zones/ zone columns that were empty for ground pixels, then the feature value of that zone column/

zone in feature vector was taken as zero. The authors obtained a maximum recognition rate of 98.05% for handwritten numerals using the support vector machine classifier.

Rajput and Mali (2010) proposed a system to guess the protein structure. For this purpose, Principal Component Analysis matrix calculation was used. PCA gave the feature vectors to compare the structure with already stored structure using the training and prediction of Neural Networks. Various components obtained from the protein taken as input were considered as input for the three different Neural Networks (artificial). The three networks had 30, 35 and 40 neurons in the hidden layer. The recognizers were trained using RPROP approach. With this system even if the components are reduced to a vast amount (from 260 to 80), performance was at least 1% more than the best possible structure prediction algorithm.

Gumah *et al.* (2010) proposed Optical Characters Recognition (OCR) technique. There were two main stages in most of OCR systems: features extraction and classification. Artificial neural networks and hidden markov models were the most popular classification methods used for OCR systems. OCR systems' efficiency can be evaluated by two parameters; the consuming time and the recognition rate. Some researchers presented systems for handwriting recognition, while others presented systems for printed words. Furthermore, some systems were designed to be used only with isolated characters or digits. Fast wavelets transform (FWT) is a mathematical algorithm designed to turn a waveform or signal in the time domain into a sequence of coefficients. The transform can be easily extended to multidimensional signals such as images, where the time domain is replaced with the space domain. In this research paper, authors presented a method where no features needed to be extracted; instead, they used the Wavelet Transform to produce a coefficient vector of the characters' images. This coefficient vector is then directly used to recognize the handwritten characters. To increase the accuracy level, the authors used two pictures of each letter in two different positions to increase the information that coefficient vector should provide. The authors could achieve 94.18% of accuracy as average.

Tsuruoka *et al.* (2010) proposed handwritten character recognition in two new generation methods of personal dictionary. They presented in first type only one character written by

one specific writer and its character selects from similar personal dictionary and the second type space feature obtained the recognition. The personal dictionary could not use off-line character recognition, because the characters written by a specified writer cost more. They proposed two new generation methods of personal recognition dictionary. The first was the expectation of personal dictionary (used as to improve the accuracy of the character recognition).The second was for previous personal dictionaries (used as "Weighted direction index histogram method").It was used as offline process. The personnel dictionary rate of 91% was relatively higher to that of a general dictionary, *i.e.*, 82% for character recognition.

Eraqi and Sherif (2011) stated a new technique for on-line Arabic handwriting recognition based on new on-line graphemes segmentation technique that depends on the local writing direction. Baseline detection, delayed strokes detection, Piece of Arabic Word (PAW) main stroke construction, and characters construction from the basic graphemes are issues that have been addressed in this research paper. The authors presented in this paper a new complete system for on-line Arabic handwriting recognition. The main contribution of this work is the new on-line graphemes segmentation technique that depends on the local writing direction. This system achieves a high recognition rate using a simple feature extraction and character construction rules. Recognition errors analysis showed that the contribution of segmentation errors in system errors was very little as compared to other problems which are almost confined to two main problems; writing strokes disorder and delayed strokes drifting.

Dhir *et al*. (2011) presented recognition of handwritten Gurumukhi numerals. They have used three different feature sets. First feature set is comprised of distance profiles having 128 features. Second feature set is comprised of different types of projection histograms having 190 features. Third feature set is comprised of zonal density and Background Directional Distribution (BDD) forming 144 features. The SVM classifier with Radial Basis Function (RBF) kernel has been used for classification. They have obtained the 5-fold cross validation accuracy as 99.2% using second feature set consisting of 190 projection

histogram features. On third and first feature sets, recognition rates of 99.13% and 98% are observed.

Jaremsri and Imprasert *et al.* (2011), in their research work, enhanced two major processes of the previous work of the off-line Thai handwritten character recognition using heuristic rules and neural network system. The researchers proposed mainly two functions: (1) Feature extraction enhancements to improve the features conflict resolution rule and the specialized neural network-based zigzag feature extraction. These functions are used to refine the conflict features and zigzag patterns. (2) Neural network-based recognition. Specifically, a neural network technique improves the capabilities of the recognition process to handle various styles of writing.

The researchers applied neural network technique to identify and recognize the zigzag pattern. The results showed that the additional feature conflict resolution rule could achieve the feature extraction rate of 87.85% (increased 2.13%), the feature extraction rate of the specialized neural network-based zigzag extraction could achieve 90.48% (increased 47.9%) and the recognition rate of the neural network-based recognition which combine both of the two proposed feature extraction functions could achieve 92.78% (increased 9.77%).In addition, integrating these two techniques to the other techniques as a hybrid approach would be very beneficial to the Thai handwritten character recognition system.

Kumar *et al.* (2011) have attempted grading of writers based on offline Gurmukhi characters written by them. Grading has been accomplished based on statistical measures of distribution of points on the bitmap image of characters. The gradation features used for classification are based on zoning, which can uniquely grade the characters. In this work, one hundred different Gurmukhi handwritten data sets have been used for grading the handwriting. They have used k-NN, HMM and Bayesian decision-making classifiers for classification. They have taken samples of Gurmukhi characters from one hundred different writers. In the process of grading, they have found the writer with best handwriting, based on the characters taken in 19 the data set. They have calculated average score of all writers obtained by them when the five features and three classifiers are used. It has been observed

that most of the researchers have done recognition in online as well as offline Handwriting recognition systems using HMM, knearest neighbor, SVM, EM, Naive Bayes methods.

Sanmorino and Yazid (2012), in their research paper, carried out a survey for handwritten signature verification. The process which they followed to recognize an individual's handwritten signature can be divided into two main areas depending on the data acquisition method, off-line and on-line signature verification. In this paper, the signature verification is based on three categories. In the first category, the authors judged how to get data signature which is off-line and on-line verification. Second is rule based approach, neural networks, hidden markov model and support vector machine. Third is based on pre-processing and feature extraction, which is thinning and line segmentation. Signature verification using neural network is widely used, as verification using NN is easy to implement. Just with simple linear algebra capabilities, a developer can create artificial NN system without experiencing any problem. One of the studies that make use of NN for signature verification is performed by Pan sare and Bhatia, which studies the relation between signature and classes based on signature features extraction. Based on the survey authors concluded that each verification method has its own advantages and disadvantages. It is very difficult to determine the best method as every method uses different parameters. However, if viewed from the ease of implementation and performance of verification, artificial neural networks (NN) or hidden markov model (HMM) is the right choice this time. If you have a lot of reference data, but have limited funds, you can use a neural network (NN) as a method of verification. If seen from the data obtaining signatures, on-line verification is more advisable than the off-line verification. Information obtained through on-line verification is more diverse and without noise.

Dubey and Agrawal (2012) stated that biometrics provide confirmation of humans on the basis of their biological features. Computer science, biometrics to be specific was used as an aspect f determination and access control. Handwriting is one of the most widely used biometric systems for authentication of person as well as document. Online and offline signature was existing in person identification and authentication. The authors, in their

research paper, discussed various features of offline signature recognition and verification process. Steps in signature recognition are as follows:

1) Data Acquisition: The signature to be processed by the system should be improper digital image format.
2) Pre-processing: Image capturing devices cause the need to normalizing input image of signature. This stage is further sub-divided into the following stages: Normalization, Image Binarization, Data Area Cropping and Thinning.

a) Normalization: Before any further processing takes place; a noise reduction filter is applied to the binary scanned image.

b) Image Binarization: It allows us to reduce the amount of image information (removing colour and background), so the output image is black-white. The black-white type of the image can be more easily further processed.

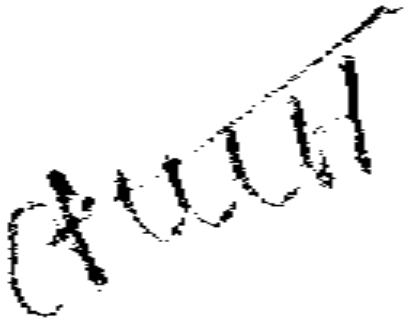


Fig. 2.2: Binarized Image

c) Data Area Cropping: The signature area is alienated from the background by using the well-known segmentation methods of vertical and horizontal projection.

Fig. 2.3: Erodated and Dilated Image

Kaur and Arora (2013) found that artificial neural networks model inspired by human nervous system was capable of learning. An important application of artificial neural network is character recognition. It finds its application in banking, security products, hospitals in robotics, etc. The paper was based on a system that recognizes English numeral given by the user which is already trained on the features of the numbers to be recognized using NNT. The features of the number given by the user are extracted and compared with the feature database, and the recognized number is displayed. Neural networks are used to solve the tasks which are very difficult to solve by conventional computer or human beings. The main aim of using this algorithm is to reduce the error which is difference between the computed value of neural network and desired value. By using artificial neural network methodology for English numeral recognition, 99% accuracy was achieved.

Shokoohi *et al.* (2013) proposed new handwritten numbers recognition by using sparse structure representation. The authors introduced a sparse structure which is a over-complete dictionary, and it is known as K-SVD algorithm. The distinction between proposed method with previous methods in addition to using the CNN and K-SVD algorithm is non-linear feature extraction. The complex neural network consists of a neural network with supervised learning architecture. CNN can be divided into two subparts: 1. automatic extractor of features, 2. learnable category. With the use of sparse structure and over-complete dictionary, the researchers investigated Persian handwritten numeral recognition on the database CENPARMI.

Chapter-3
Representing Mathematical Functions Using ANN

3.1 Artificial Neural Network

Artificial Neural Networks (also known as Neural Networks) are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the network function is determined largely by the connections between elements. The functionality of a neural network is determined by the combination of the topology (number of layers, number of units per layer, and the interconnection pattern between the layers) and the weights of the connections within the network. The topology is usually held fixed, and the weights are determined by a certain training algorithm. The process of adjusting the weights to make the network learn the relationship between the inputs and targets is called learning, or training. Such a situation is shown in Fig. 3.1. there, the network is adjusted, based on the input, hidden nodes and output, until the network output matches the target.

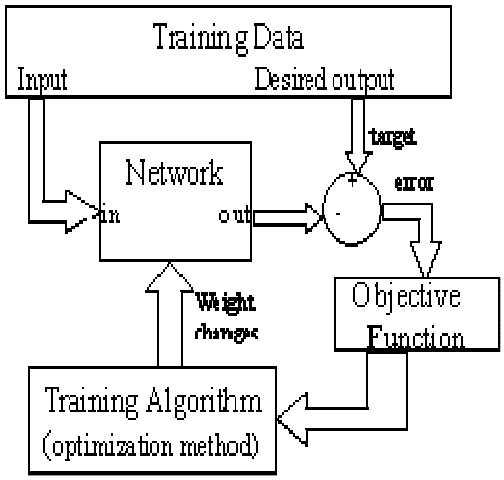


Fig. 3.1: Architecture of Neural Network

It is often posed as a function approximation problem-given training data consisting of pairs of input patterns *x*, and corresponding target *t*, the goal is to find a function *f(x)* that matches the desired response for each training input. Neural networks have been trained to

perform complex functions in various fields of application including pattern recognition, identification, classification, speech, and vision and control systems. There are a variety of designs and learning techniques that enrich the choices that a user can make.

3.1.1 Network Topologies

The topology of a network is defined by the number of layers, the number of units per layer, and the interconnection patterns between layers (Arulmozhi, 2011). There are various categories based on the pattern of connections such as newff (feed-forward back propagation network), newcf (cascade-forward back propagation network), new elm (Elman back propagation network), etc. According to best performance outcome of network, feed-forward back propagation network is the main focus of this dissertation. In all mathematical problems, this kind of network topology is used.

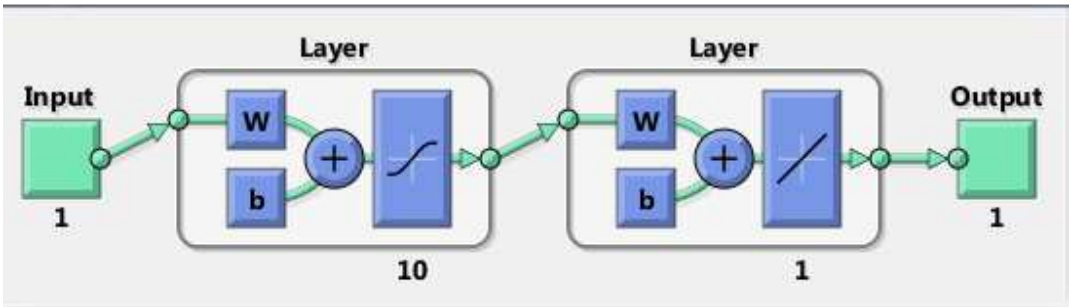


Fig. 3.2 : Network Architecture for one Input and one Output Function

Today, neural networks can be trained to solve complex problems that are difficult for conventional computers or human beings. So, to represent different mathematical functions, the artificial neural network is used to train the network. ANN is self-adjusting to a new environment without using any pre-programmed instructions.

3.1.2 Training Data

Training a network consists of an iterative process in which the network is given the desired inputs along with the correct outputs for those inputs. It then seeks to alter its parameters to try and produce the correct output (within a reasonable error margin). If it succeeds, it has learned the training set and is ready to perform upon previously unseen data. If it fails to produce the correct output it re-reads the input and again tries to produce the correct output. The MATLAB is used to train the network. The default training

optimization method is trainlm (Levenburg-Marquardt) which is a combination of gradient descent and Newton's method (Xue and Chen, 2009).

Different tuning parameters have been used to train the network. These are described in Table 3.1.

Table 3.1: Tuning (Training) Different Parameters

S. No.	Parameters	Description	Values
1.	Input	No. of neurons as of inputs	where a, b, and c are real numbers
2.	Output	No. of neurons as outputs	*Y* function output
3.	Hidden unit	Number of neurons in the hidden layer. It is denoted by S.	S = size
4.	Epochs	maximum no. of iteration	1000
5.	Training algorithm	Newff (feed-forward neural network).	//algorithm
6.	Normalization	To set the resize of function. it is denoted by R.	
7.	Activation function	Activation functions in the input layer Each hidden node and output node applies the activation function to its net input. These parameters are tuned simultaneously to minimize the prediction error on the training.	{'tansig','purelin'} The tansig function is the inverse tangent function, which gives the same performance as our standard sigmoid function. We also use purelin function for linear output.
8.	"train"	This process determines the "best" set of epochs for our data set by training the network.	Net = train(net,*x*,*y*)
9.	Plot	Plot command is used to graphic view of trained data.	Plot the progress of training.
10.	Scatter	A scatter plot might be the proper graph when we try to show how two variables may be related to each other.	
11.	Sim	Simulates the neural networks.	
12.	Mean Square Error	Mse is a network performance function. It measures the network's performance according to the mean of squared errors.	Error function

3.1.3 Training Results

After the training process these parameters give some resultant values, which are exhibited in Table 3.2.

Table 3.2: Training Results

S. No.	Parameters	Results
1.	Input	No. of inputs
2.	Output	No. of outputs
3.	Time	Maximum time taken
4.	Performance	Best performance at 1000 epochs
5.	Gradient	minimum gradient magnitude
6.	Validation checks	Validation checks at 1000 epochs
7.	Graph	Graphical Representation of data

3.1.4 Testing Data and Results

We need to test our network to see if it has found a good balance between accuracy and generalization. To test network there should be different parameter values put into function. It gives different resultant values according to the trained network.

3.2 Mathematical Functions

In this work, we have approximated following functions using ANN.

1. Linear function, $y = a + bx$.
2. Quadratic function, $y = a + bx + cx^2$.
3. Sin(x) function, $y = \sin(x)$.
4. Exponential function, $y = \exp(x)$.
5. Log sig(x) function, $y = \text{logsig}(x)$.
6. Sigmoid function, $y = \text{sigmoid}(x)$.

3.2.1 Approximating One Variable Linear Function, $y = a + bx$ Using ANN

Linear functions are those whose graph is a straight line. The linear function taken in this work has one independent variable and one dependent variable. The independent variable is x and the dependent variable is y. To represent the linear function, we need to construct the design procedure such as collecting data, creating the network, configuring the network, training the network, and validating the network.

3.2.1.1 Training Results of linear function

The parameters for the network that has been trained for implementing linear function are given in Table 3.3.

Table 3.3: Training Parameters of Linear Function

S. No.	Parameters	Results
1.	Input	x = -100:0.2:100 a = -50:0.1:50 b = -50:0.1:50
2.	Output	y = a + bx
3.	Hidden unit	S = 10
4.	Normalization	R = [-10 10]
5.	Epochs	1000
6.	Time	0:00:24
7.	Performance	1.69 (The best performance at 1000 epochs)
8.	Gradient	10.2969 at 1000 epochs
9.	Mu	1.00e-05 at 1000 epochs
10.	Validation checks	0

3.2.1.2 Graphical Representation of Training Linear Function

Graphs are often an excellent way to display the results. It contains the two axis, the independent variable on the *x*-axis which shows the actual value of function of graph and the dependent variable on the *y*-axis shows the value of function by ANN. A *xy*-line graph shows the relationship between our dependent and independent variables which plot the linear line. The graph is given below in Fig 3.3 during the training process of data.

Training of Data:

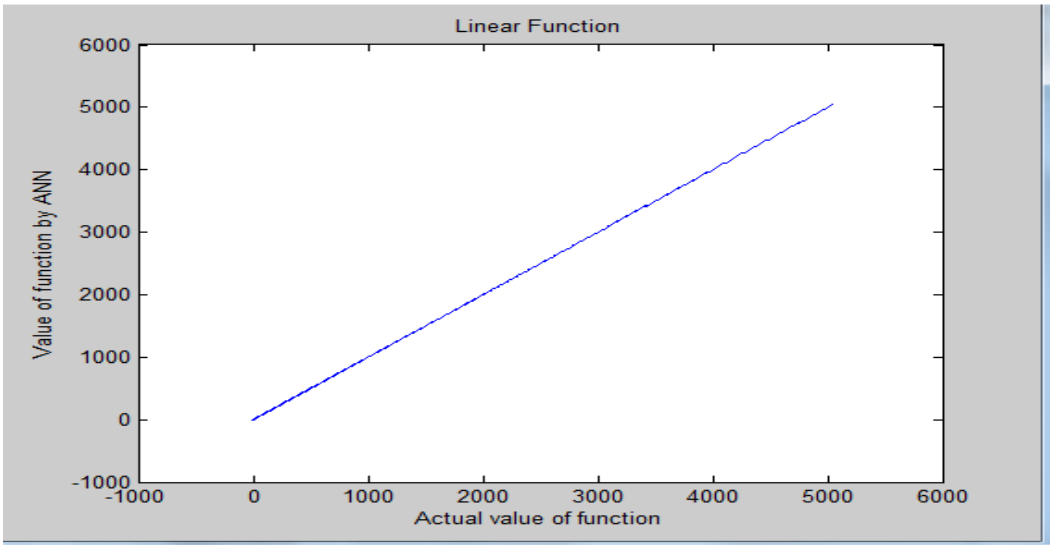


Fig. 3.3: Value of Function by ANN Vs. Actual Value of Function for the Function

3.2.1.3 Testing Results of Linear Function

We need to test our network to see if it has found a good balance between accuracy and generalization. Different parameter values are used to test the network which are shown in Table 3.4.

Table 3.4: Testing Parameters of Linear Function

S. No.	Parameters	Results
1.	Input	$x1$ = -100:0.3:100 a1 = -50:0.15:50 b1 = -50:0.15:50
2.	Output	$y1$ = a1 + b1$x1$
3.	Time	0:00:26
4.	Performance	1.6869 (The best performance at 1000 epochs)
5.	Gradient	10.8298 at 1000 epochs
6.	Mu	1.00e-05 at 1000 epochs
7.	Validation checks	0

The graph during the testing process of data is linear function as shown in Fig. 3.4.

3.2.1.4 Graphical Representation of Testing Linear Function

Testing of Data: Output of tested data of linear function

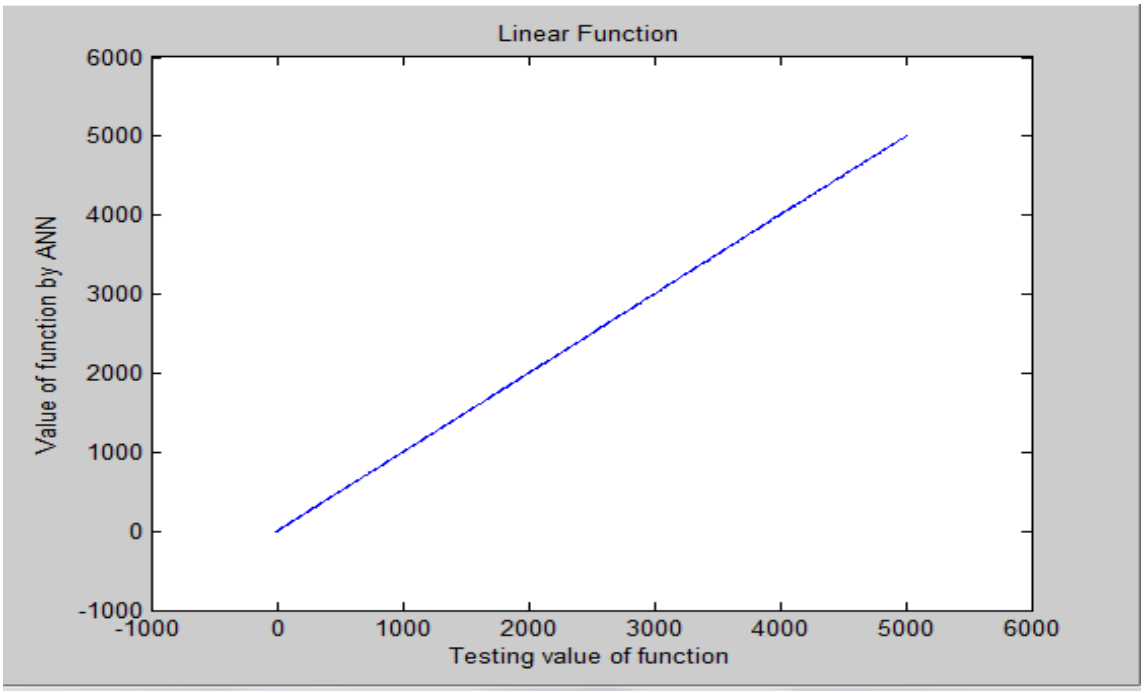


Fig. 3.4: Value of Function by ANN Vs. Testing Value of Function for the Function

3.2.2 Approximating One Variable Quadratic Function, $y = a + bx + cx^2$ Using ANN

A quadratic function is a second-degree polynomial function of the form $y = a + bx + cx^2$, where a, b, and c are real numbers. Every quadratic function has a "u-shaped" graph called a parabola. To represent the quadratic function, we need to construct the design procedure such as collecting data, creating the network, configuring the network, training the network using neural network techniques and validating the network.

3.2.2.1 Training Results of Quadratic Function

The data and results of quadratic function are highlighted in Table 3.5.

Table 3.5: Training Parameters of Quadratic Function

S. No.	Parameters	Results
1.	Input	x = -15:0.15:15 a = -5:0.05:5 b = -5:0.05:5 c = -5:0.05:5
2.	Output	$y = a + bx + cx^2$
3.	Hidden unit	S = 12
4.	Normalization	R = [-10 10]
5.	Epochs	1000
6.	Time	0:00:53
7.	Performance	0.17244(The best performance at 1000 epochs)
8.	Gradient	2.2892 at 1000 epochs
9.	Mu	1.00e-06 at 1000 epochs
10.	Validation checks	0

3.2.2.2 Graphical Representation of Training Quadratic Function

The graph formed during the training process of quadratic function is shown in Fig. 3.5.

Training of Data:

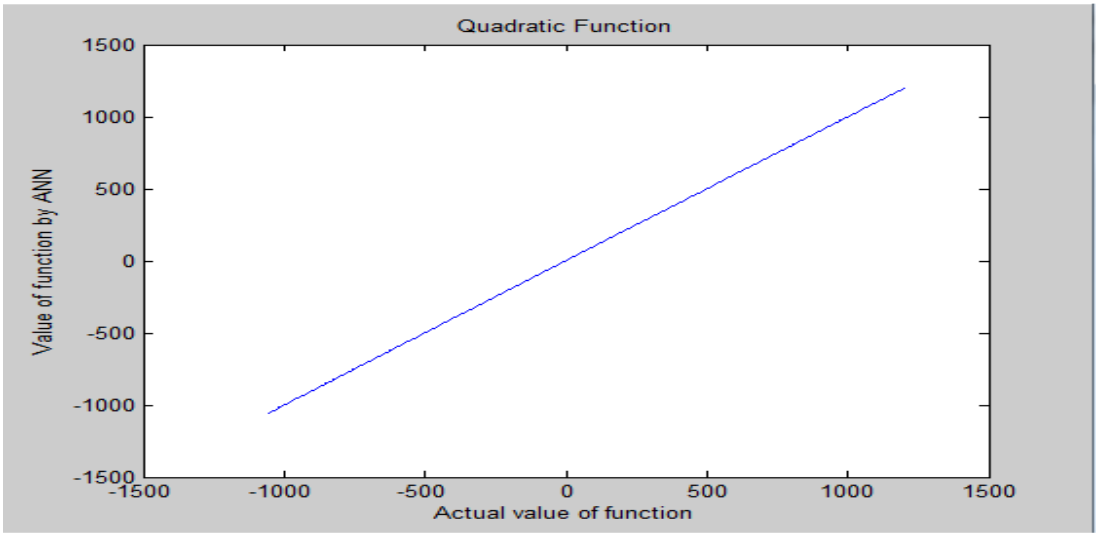


Fig. 3.5: Value of Function by ANN Vs. Actual Value of Function for the Function

3.2.2.3 Testing Results of Quadratic Function

We have used different parameter values to test our network which are shown in Table 3.6.

Table 3.6: Testing Parameters of Quadratic Function

S. No.	Parameters	Results
1.	Input	$x1$ = -15:0.06:15 a1 = -5:0.02:5 b1 = -5:0.02:5 c1 = -5:0.02:5
2.	Output	$y1 = a1 + b1x + c1x^2$
3.	Time	0:00:53
4.	Performance	1.3883e-06 (The best performance at 1000 epochs)
5.	Gradient	0.7442 at 1000 epochs
6.	Mu	1.00e-08 at 1000 epochs
7.	Validation checks	0

3.2.2.4 Graphical Representation of Testing Quadratic Function

The graph formed during the testing process of quadratic function is shown in Fig. 3.6.

Testing of Data:

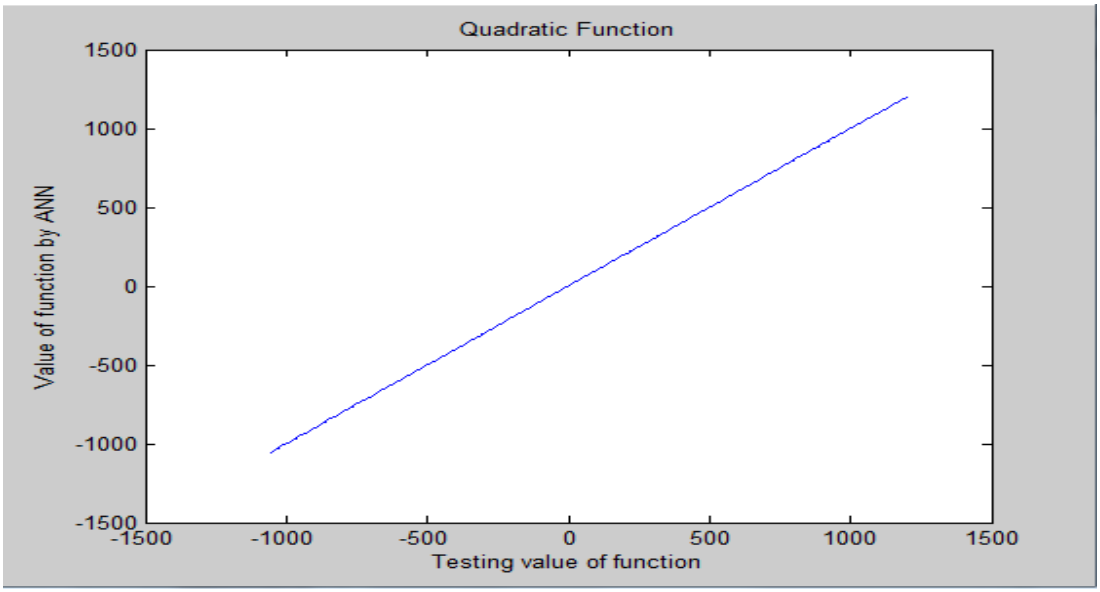


Fig. 3.6: Value of Function by ANN Vs. Testing Value of Function for the Function

3.2.3 Approximating Function, $y = \sin(x)$ Using ANN

The sin(x) function may accept both real and complex inputs. We have, however, worked on real numbers. It returns the real values in the interval [-1, 1].

3.2.3.1 Training Results of sin(x) Function

Different tuning parameters used to train our network are shown in Table 3.7 below.

Table 3.7: Training Parameters of sin(x) Function

S. No.	Parameters	Results
1.	Input	x = -2:0.01:2
2.	Output	y = sin(x)
3.	Hidden unit	S = 20
4.	Normalization	R = [-1 1]
5.	Epochs	1000
6.	Time	0:00:08
7.	Performance	0.000211 (performance at 365 epochs)
8.	Gradient	7.42e-08
9.	Mu	1.00e-09
10.	Validation checks	0

3.2.3.2 Graphical Representation of Training sin(x) Function

The graph formed during the training process of data is shown in Fig. 3.7.

Training of Data:

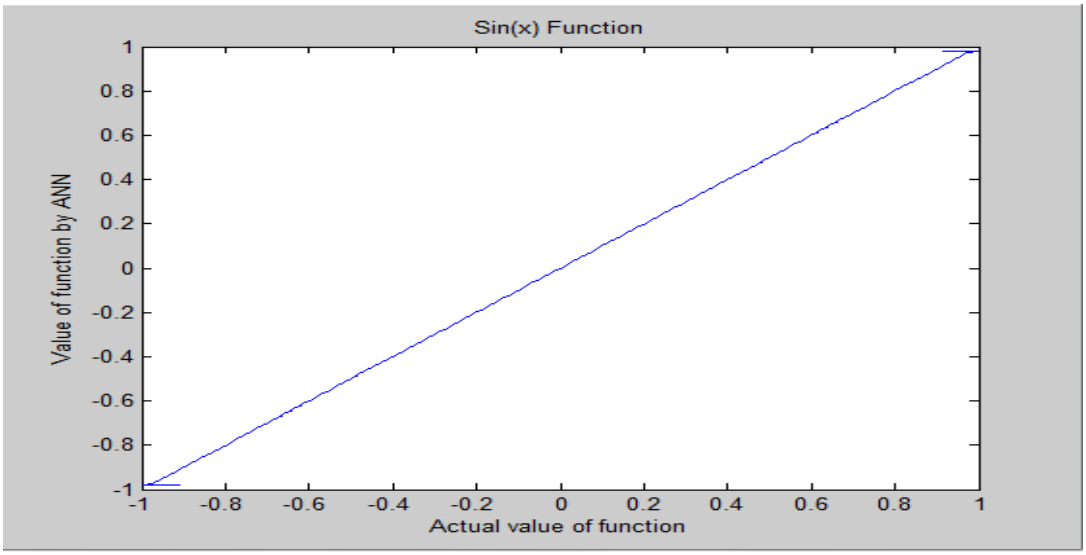


Fig. 3.7: Value of Function by ANN Vs. Actual Value of Function for the Function

3.2.3.3 Testing Results of sin(*x*) Function

Different parameter values used to test our network which provide the resultant values, are shown in Table 3.8.

Table 3.8: Testing Parameters of sin(*x*) Function

S. No.	Parameters	Results
1.	Input	*x*1 = -2:0.02:2
2.	Output	*y*1 = sin(*x*1)
3.	Time	0:00:09
4.	Performance	0.000211 (performance at 396 epochs)
5.	Gradient	6.69 -08
6.	Mu	1.00e-09
7.	Validation checks	0

3.2.3.4 Graphical Representation of Testing sin(*x*) Function

The graph formed during the testing process of sin(*x*) function is shown in Fig. 3.8.

Testing of Data:

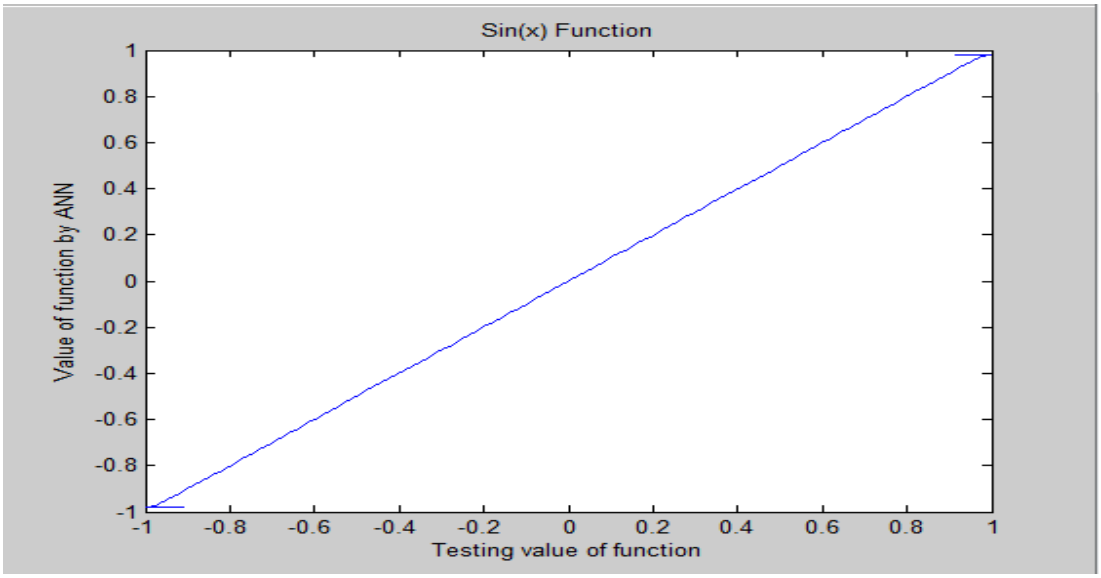


Fig. 3.8: Value of Function by ANN Vs. Testing Value of Function for the Function

3.2.4 Approximating Exponential Function, $y = \exp(x)$ Using ANN

The exponential function is a mathematical function in which an independent variable appears in one of the exponents. The function accepts both real and complex inputs. For real values exp function returns the values in the interval $(0, \infty)$.

3.2.4.1 Training Results of exp(x) Function

In the training process, we train the neural network using some parameters. The value of these parameters for exp (x) function is given in Table 3.9.

Table 3.9: Training Parameters of exp(x) Function

S. No.	Parameters	Results
1.	Input	x = -1:0.01:1
2.	Output	$y = \exp(x)$
3.	Hidden unit	S = 50
4.	Epochs	1000
5.	Normalization	R = [-20 10]
6.	Time	0:00:46
7.	Performance	2.56e-10 best performance at 1000 epochs
8.	Gradient	1.48e-07
9.	Mu	1.00e-08
10.	Validation checks	0

3.2.4.2 Graphical Representation of Training exp(x) Function

The graph formed during the training process of exp(x) function by ANN is shown in Fig.3.9.

Training of Data:

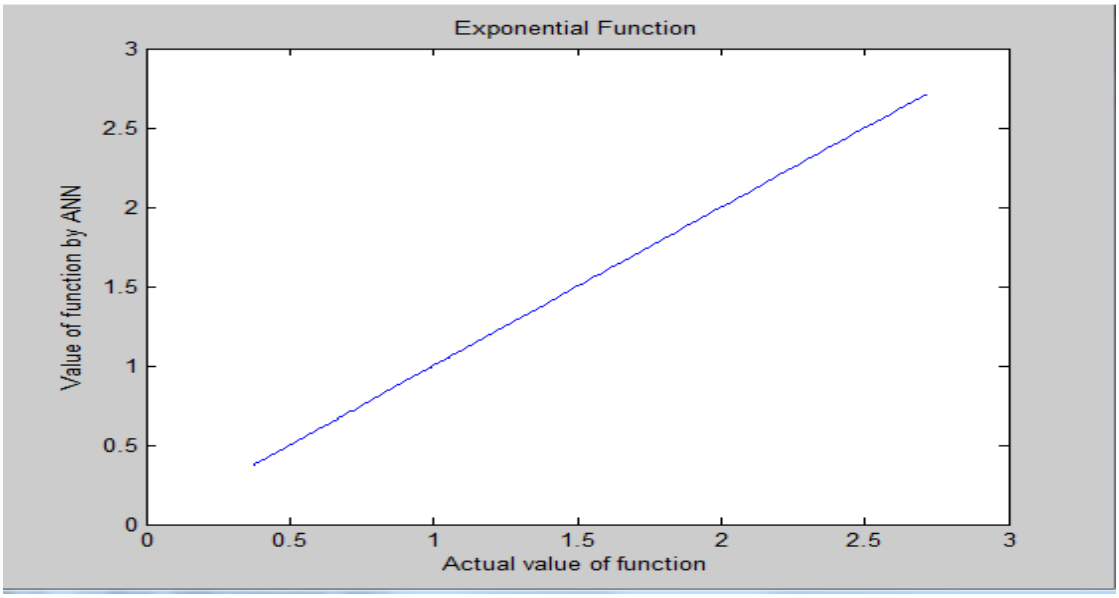


Fig. 3.9: Value of Function by ANN Vs. Actual value of Function for the Function

3.2.4.3 Testing Results of exp(*x*) function

Different parameter values that have been used to test our network are shown in Table 3.10.

Table 3.10: Testing Parameters of exp(*x*) Function

S. No.	Parameters	Results
1.	Input	*x*1 = -2:0.02:2
2.	Output	*y*1 = exp(*x*1)
3.	Time	0:00:46
4.	Performance	1.49e-09 performance at 1000 epochs
5.	Gradient	3.79e-07
6.	Mu	1.00e-07
7.	Validation checks	0

3.2.4.4 Graphical Representation of Testing exp(*x*) Function

The graph formed during the testing process of exp(*x*) function is shown in Fig. 3.10.

Testing of Data:

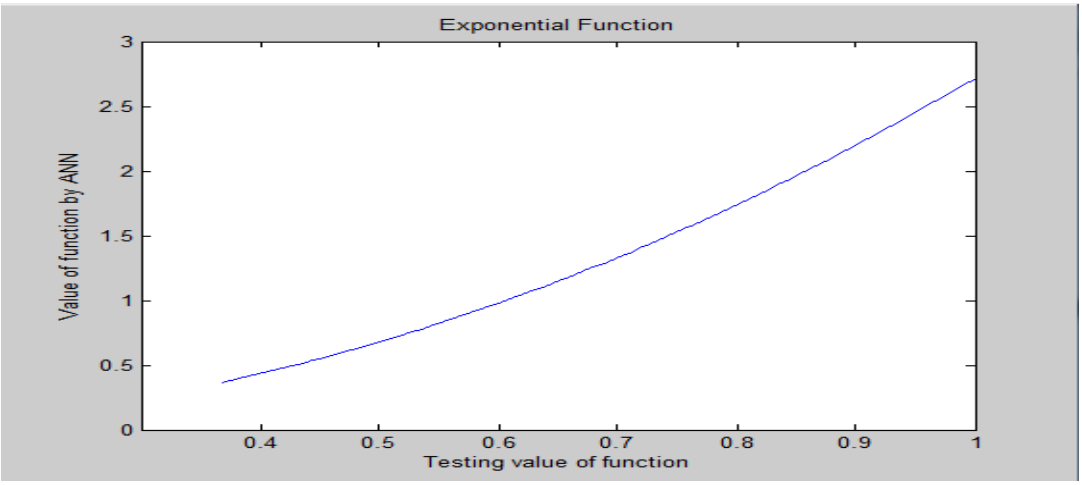


Fig. 3.10: Value of Function by ANN Vs. Testing Value of Function for the Function

3.2.5 Approximating the Function *y* = logsig(*x*) Using ANN

Logsig is a transfer function which calculates the layers output from its net input. It generates the output between 0 and 1 as the neuron's net input goes from negative to positive infinity.

3.2.5.1 Training Results of logsig(*x*) Function

Different tuning parameters have been used to train our network. These are exhibited in Table 3.11.

Table 3.11: Training Parameters of logsig(*x*) Function

S. No.	Parameters	Results
1.	Input	*x* = -10:0.01:10
2.	Output	*y* = logsig(*x*)
3.	Hidden unit	S = 15
4.	Epochs	1000
5.	Normalization	R = [-10 10]
6.	Time	0:00:01
7.	Performance	1.80e-16 at 35epochs
8.	Gradient	1.82e-08
9.	Mu	1.00e-14
10.	Validation checks	0

3.2.5.2 Graphical Representation of Training logsig(*x*) Function

The graph formed during the training process of logsig(*x*) function by ANN is shown in Fig. 3.11.

Training of Data:

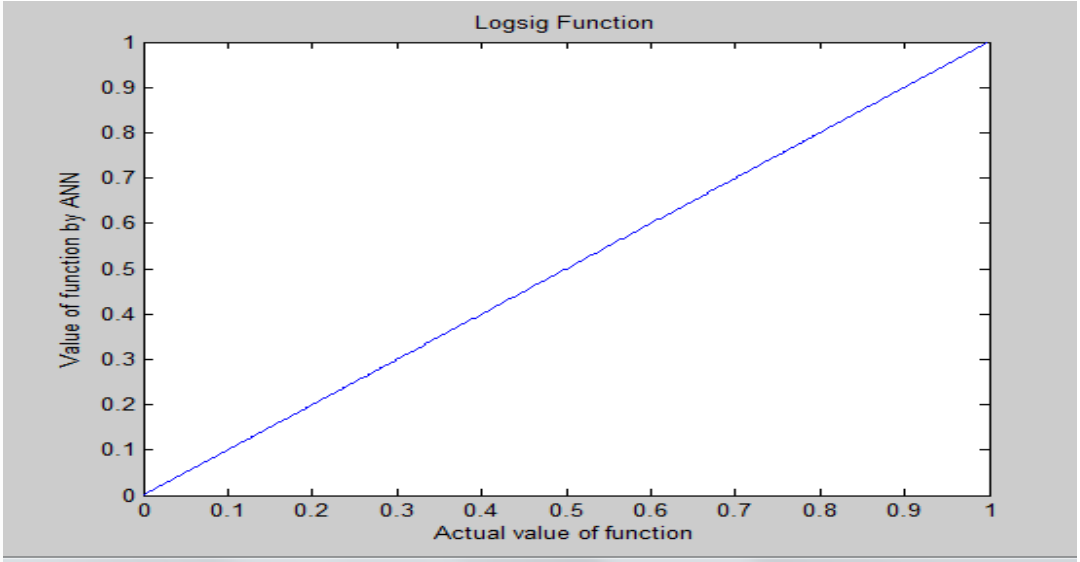


Fig. 3.11: Value of Function by ANN Vs. Actual Value of Function for the Function

3.2.5.3 Testing Results of logsig(*x*) Function

Different parameter values have been used to test our network which provide the resultant values and and are shown in Table 3.12.

Table 3.12: Testing Parameters of logsig(*x*) Function

S. No.	Parameters	Results
1.	Input	$x1$ = -10:0.02:10
2.	Output	$y1$ = logsig($x1$)
3.	Time	0:00:01
4.	Performance	8.07e-20 at 29 epochs
5.	Gradient	4.28e-12
6.	Mu	1.00e-10
7.	Validation checks	0

3.2.5.4 Graphical Representation of Testing logsig(*x*) Function

The graph formed during the testing process of logsig(*x*) function is shown in Fig. 3.12.

Testing of Data:

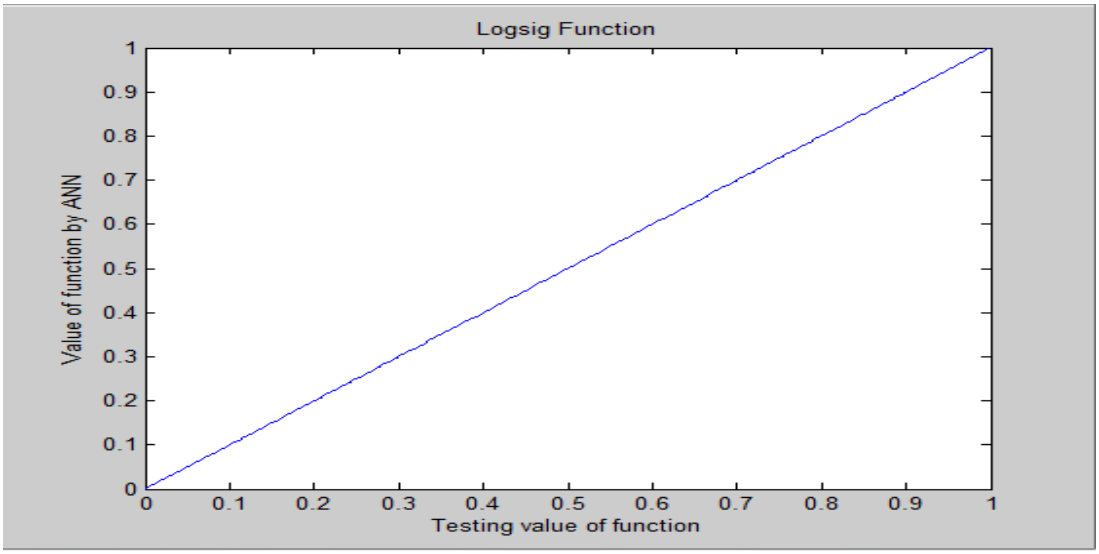


Fig. 3.12: Value of Function by ANN Vs. Testing Value of Function for the Function

3.2.6 Approximating sigmoid Function, *s* =1 / (1 + exp(-*t*) Using ANN

A sigmoid function is a mathematical function having an "S" shape (sigmoid curve). It refers to the special case of the logistic function, and is defined by the formula

$$S(t) = \frac{1}{1 + e^{-t}}.$$

Multilayer networks typically use sigmoid transfer functions in the hidden layers. Sigmoid functions are characterized by the fact that their slope must approach zero, as the input gets larger.

3.2.6.1 Training Results of sigmoid(*x*) Function

Different tuning parameters have been used to train our network which are highlighted in Table 3.13.

Table 3.13: Training Parameters of sigmoid(x) Function

S. No	Parameters	Results
1.	Input	x = -10:0.01:10
2.	Output	y = sigmoid(x)
3.	Time	0:00:07
4.	Hidden unit	S = 10;
5.	Epochs	1000
6.	Normalization	R = [-10 10]
7.	Performance	1.82e-15 at 85 epochs
8.	Gradient	6.81e-08
9.	Mu	1.00e-10
10.	Validation checks	0

3.2.6.2 Graphical Representation of Training sigmoid(x) Function

The graph formed during the training process of sigmoid(x) function by ANN which is shown in Fig. 3.13 as given below:

Training of Data:

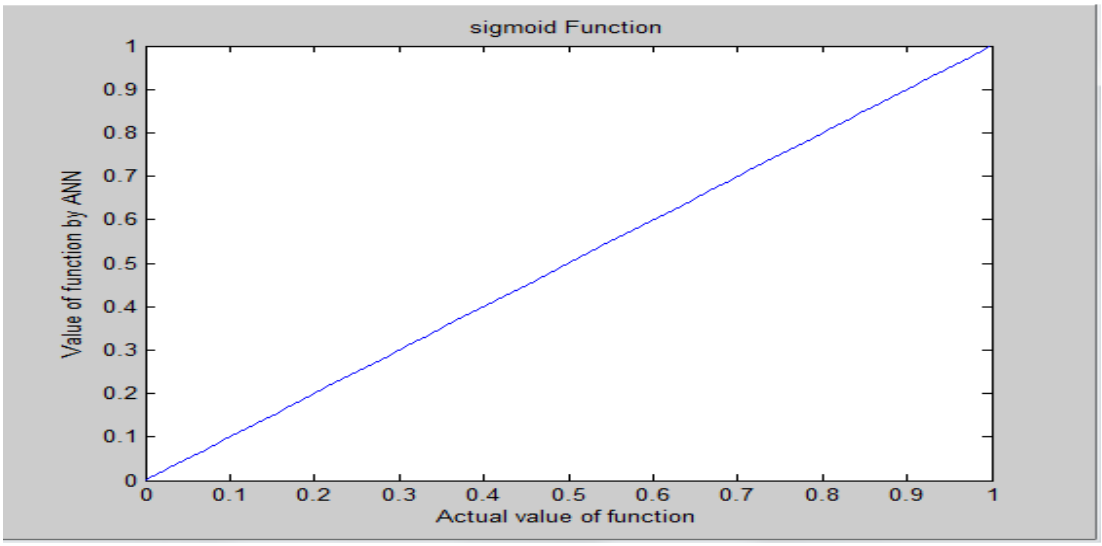


Fig. 3.13: Value of Function by ANN Vs. Actual Value of Function for the Function

3.2.6.3 Testing Results of sigmoid(x) Function

Different parameter values have been used to test our network which provide the resultant values, and are shown in Table 3.14.

Table 3.14: Testing Parameters of sigmoid(x) Function

S. No.	Parameters	Results
1.	Input	$x1$ = -10:0.02:10
2.	Output	$y1$ = sigmoid($x1$)
3.	Time	0:00:03
4.	Performance	9.30e-17 at 48 epochs
5.	Gradient	1.53e-08
6.	Mu	1.00e-10
7.	Validation checks	0

3.2.6.4 Graphical Representation of Testing sigmoid(x) Function

The graph formed during the testing process of sigmoid(x) function is shown in Fig. 3.14 as given below:

Testing of Data:

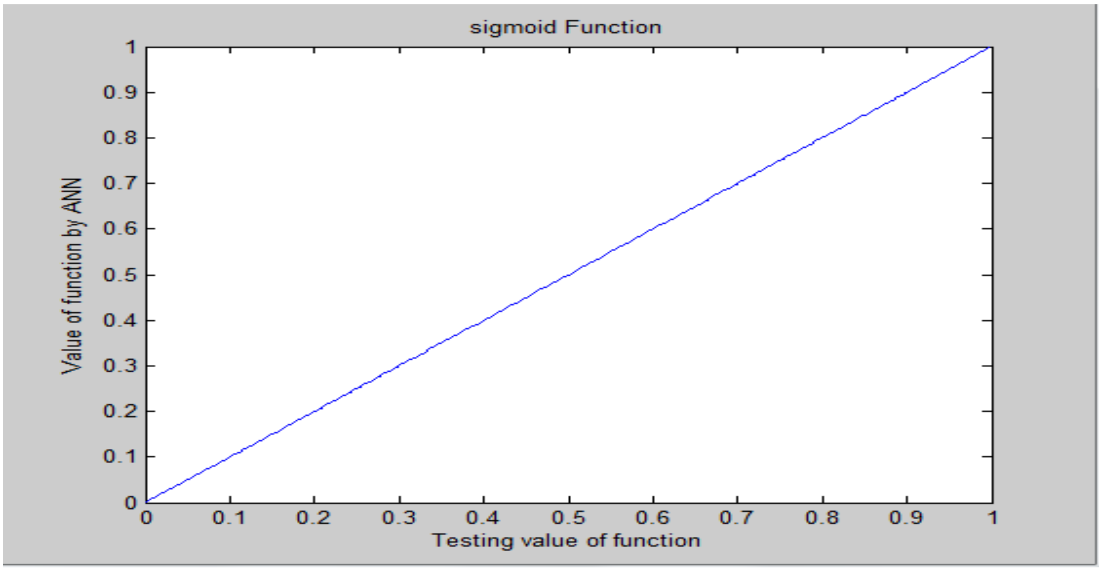


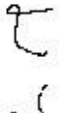
Fig. 3.14: Value of Function by ANN Vs. Testing Value of Function for the Function

Chapter-4
Recognition of Online Handwritten Gurmukhi Numerals using ANN

4.1 Online Handwritten Gurmukhi Numerals Recognition

Recognition database has been created to recognize the Gurmukhi numerals. It is shown in Table 4.1. The database has a unique stroke-id to every symbol (Gurmukhi numeral). Ten numerals of Gurmukhi numbers from zero to nine have been taken for this process.

Table 4.1: Gurmukhi Numerals Database

S.No.	Stroke ID	Symbol	Description	Description
1.	**1**		One	Ikk
2.	**2**		Two	Do
3.	**3**		Three	Tinn
4.	**4**		Four	Char
5.	**5**		Five	Punj
6.	**6**		Six	Che
7.	**7**		Seven	Satt
8.	**8**		Eight	Ath
9.	**9**		Nine	Naum
10.	**0**		Zero	Sifar

Data from 30 different people has been collected for the recognition of Gurmukhi numerals. Each writer contributed 10 samples of each numeral from zero to nine. Table 4.2 displays some of the samples of collected dataset.

Table 4.2: Handwritten Samples of Gurmukhi Numerals

Digit	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					

4.2 Methodology Used

The system automatically uses the intelligence of layered neural network to recognize the Gurmukhi numerals. The neural network applied in the system utilizes the Steepest Descent Algorithm with Adaptive Learning Rate as the learning rule, which is used for the training of the neural network (Dhir *et al*., 2011). The basic flow of the system is highlighted in Fig. 4.1.

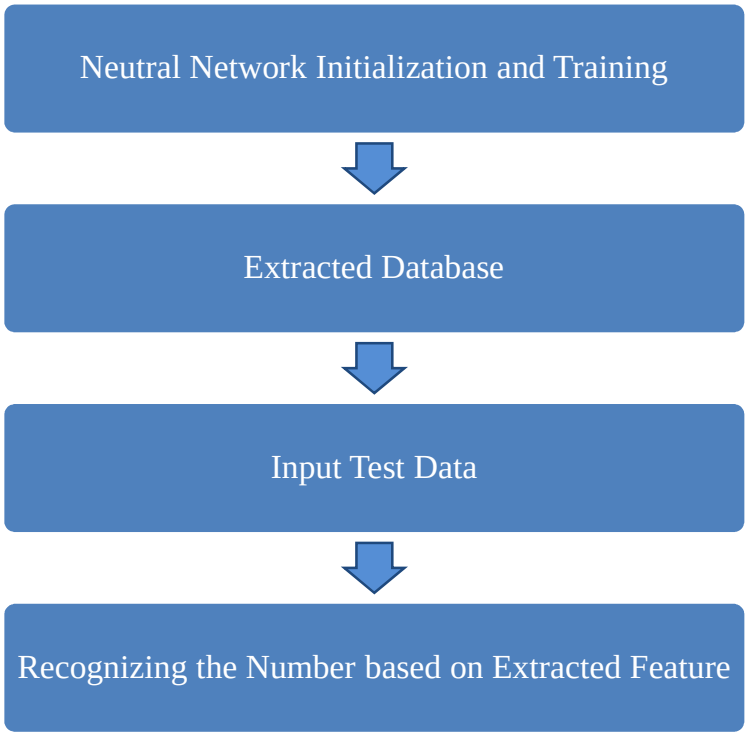


Fig. 4.1: Steps to Recognize a Handwritten Gurmukhi Numeral

4.2.1 Neural Network Architecture

The neural network training aims to find the connection between neurons that determine the global minimum of the error function. In this work, Levenberg-marquardt algorithm to train the classifier for the classification of Human activity data from the accelerometer sensor. For the activity recognition, a fast training was supposed to be used. Therefore, the Levenberg-marquardt algorithm is used for training the neural network. Finding the global minimum also requires selection of some parameters. MATLAB provides the default values for these parameters (Xue and Chen, 2009). An empirical approach was followed for modifying the parameter values for the designing of the neural network. The values of parameters used in this design are shown in Table 4.3 with a brief description of each parameter.

Table 4.3: Parameters Used for Neural Network Training

Parameters	Value Description	Default Value Description
Input	64×64	Input data
Output	1	Output data
Epochs	1000	Maximum No. of epochs
Time	00:00:00	Minimum time
Performance	0.01	Minimum performance gradient
Hidden layer size	10, 20	No. of hidden layers
Mu	0.0001	Initial Mu
Validation checks	0	0

User handwriting input with 64 pixel performs one output as digit. The neurons are arranged into various layers: Input layer, Hidden layer and Output layer. The neurons in each layer are fully interconnected by connection strengths called as weights. Also, each hidden and output layer neuron consists of a bias term associated with it. The designing of neural network involves the decision in number of hidden layers and number of neurons in each layer along with the transfer functions. In this work, MATLAB neural network has been used for implementation (Xue and Chen, 2009). The network architecture for implementation is shown in Fig. 4.2 with 10 (no. of neurons) and in Fig. 4.3 with 20 (no. of neurons). The performance of the neural network is evaluated with the increasing number of hidden layers as well as each hidden layer neurons for both the training and testing data. Here, we divide the database for training in two types. Firstly, we train the database with 10 neurons, and then with 20 neurons which is shown in Figures 4.2 and 4.3 respectively.

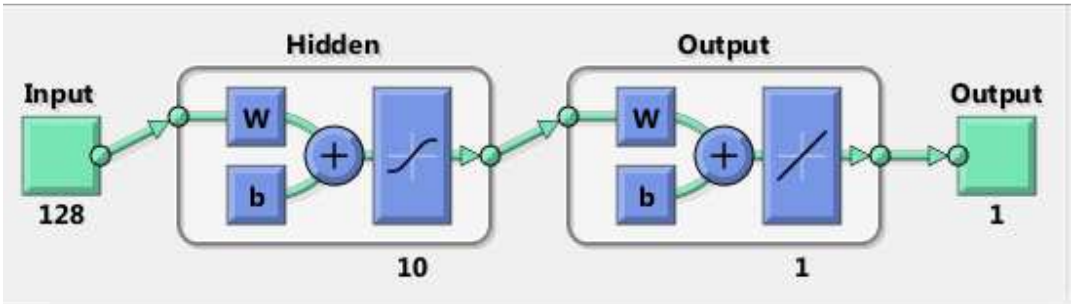


Fig. 4.2: Network Architecture for Implementation of Gurmukhi Numerals by Using 10 Neurons of ANN

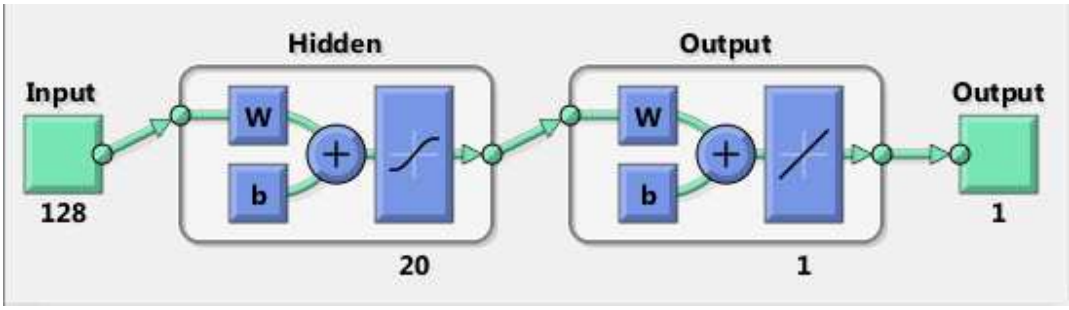


Fig. 4.3: Network Architecture for Implementation of Gurmukhi Numerals by Using 20 Neurons of ANN

4.2.2 Feature Extraction

Feature extraction is used to extract relevant features for recognition of Gurmukhi numerals. Firstly, features are computed and extracted; and then most relevant features are selected to construct feature vector which is used for recognition. The computation of features is based on statistical, structural, directional, moment, and transformation like approaches. Feature extraction is extracting information from raw data which is more relevant for the classification purpose and that minimizes the variation within a class and maximizes the variations between classes (Dhir *et al.*, 2011).

4.2.2.1 Handwritten Input Datasheet for Gurmukhi Numerals Recognition

Input the excel database sheet as input which is to be tested. Input data may be any numerals from 0 to 9. It may be 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. The samples have been taken as x and y coordinates. We have taken 50, 100, 150 and 200 different users' samples to train the

neural network as shown in Fig. 4.4 up to 64×64 points. The output shown in output datasheet is shown in Fig. 4.5.

	FY	FZ	GA	GB	GC	GD	GE	GF	GG	GH	GI	GJ	GK	GL	GM	GN	GO	GP	GQ	GR
1	269	290	300	262	266	281	300	300	232	255	262	233	300	300	272	300	270	280	281	271
2	0	0	10	0	17	0	0	0	0	0	0	0	0	0	19	16	10	0	0	0
3	259	280	290	254	256	270	290	289	226	246	252	223	289	289	262	288	260	269	274	262
4	7	7	8	8	12	0	3	10	5	0	7	9	2	3	14	10	7	5	6	0
5	250	274	281	246	247	265	280	280	217	237	244	215	284	278	252	277	251	260	266	253
6	15	13	6	21	8	2	8	21	15	2	16	17	5	8	9	6	5	12	16	2
7	246	266	270	238	237	254	280	273	210	228	236	207	273	267	242	265	241	251	256	244
8	19	20	3	31	4	4	10	30	22	4	23	24	8	11	6	3	3	20	26	4
9	238	258	259	233	227	243	270	268	203	219	229	198	262	256	232	253	231	242	248	235
10	27	29	1	41	1	6	13	39	31	6	31	30	13	16	2	3	1	28	34	6
11	230	248	248	225	217	232	262	259	197	210	222	191	253	246	222	243	221	232	243	226
12	34	38	0	53	0	10	18	49	38	9	39	41	23	22	0	8	1	39	42	7
13	222	241	243	225	213	224	253	257	190	210	213	189	246	246	212	233	210	22	233	223
14	43	48	10	67	8	21	23	63	46	23	48	55	33	38	0	16	10	55	54	19
15	216	240	237	225	205	225	243	257	184	209	213	189	245	246	205	227	207	22	232	214
16	55	63	18	82	18	37	29	80	58	33	62	67	50	53	5	24	23	72	66	19
17	212	240	235	225	198	221	233	257	184	209	213	189	247	246	198	226	200	232	230	213
18	67	77	31	99	31	49	35	94	72	47	75	83	66	69	9	41	36	87	79	29
19	207	240	228	225	198	215	226	257	184	207	213	189	254	253	188	219	200	239	223	213
20	79	91	37	115	45	63	42	108	84	57	91	96	82	83	15	59	50	97	91	42
21	207	240	228	234	198	215	222	255	184	200	213	189	253	255	180	218	200	242	230	213
22	93	106	52	122	58	80	55	124	97	68	104	110	97	100	24	75	65	112	104	54
23	207	239	228	234	198	215	222	249	184	200	223	189	251	251	178	212	200	242	232	212
24	108	118	68	138	74	96	71	129	111	80	120	124	109	107	37	85	80	126	118	65

Fig. 4.4: Datasheet of Numerals as Input

4.2.2.2 Output Datasheet of Gurmukhi Numerals Recognition

According to input, next datasheet (Fig. 4.5) shows the output window with stroke-id number.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	0	0	0	0	0	1	1	1	1	1	2	2	2	2	2	3	3	3	3	3
2																				
3																				
4																				
5																				
6																				
7																				
8																				
9																				
10																				

Fig. 4.5: Datasheet of Numerals as Output

4.2.3 Neural Network Training

Neural network is trained as shown in Figures 4.2 and 4.3. Here, we have set 1000 iterations for the network to train. It shows our performance goal, and maximum number of iterations. Other parameters of network also give the value as Time taken for training of network. It is displayed as 0:00:36, mse, as well 0.1286 as performance.

The whole data was divided into three folds; 70% data for training, 15% each for validation and testing. Of these three folds, two were used as training data sets and the rest as the test data. The training and testing process was repeated for all the possible combinations of these three folds. The training performance as calculated from the performance command was computed as 0.057 which was the best performance.

4.2.4 Training Results Pertaining to Gurmukhi Numerals' Recognition

Data is tested to check whether there is a good balance between accuracy and generalization. The 15% of data used for testing gives the resultant performance output of 0.79586. The recognition accuracy for training data of Gurmukhi numerals is given in Table 4.4 and outputs shown in Figures 4.7 to 4.9. In the table given below we present the training accuracy for different number users, *i.e.*, 50, 100, 150 and 200 by using 10 and 20 neurons in ANN.

Table 4.4: Recognition Accuracy for Training Data of Gurmukhi Numerals

S. No.	Size of Training Data	No. of Neurons in Hidden Layer	Recognition Accuracy (%)
1.	50	10	91.1
2.	50	20	94.2
3.	100	10	90.0
4.	100	20	96.1
5.	150	10	90.1
6.	150	20	93.3
7.	200	10	92.2
8.	200	20	96.4

The best accuracy of 92.2% has been achieved with 10 neurons of ANN, while it is 96.4% with 20 neurons.

4.2.5 Testing Process and Results of Recognizing Gurmukhi Numerals

The testing has been done on database which is not used in the training of neural network. The basic flow of testing the system is shown in Fig. 4.6.

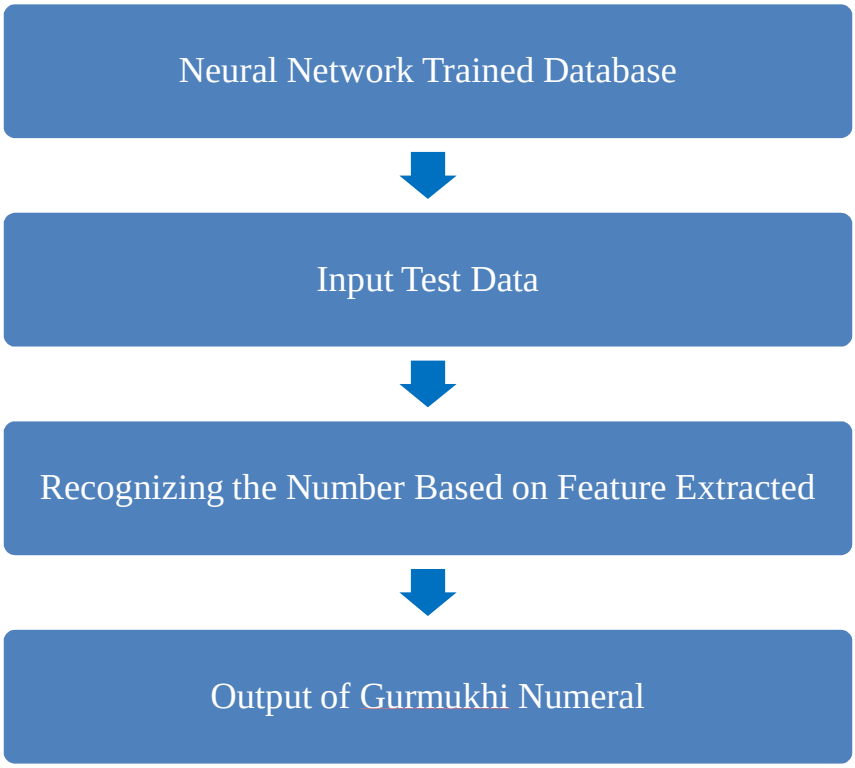
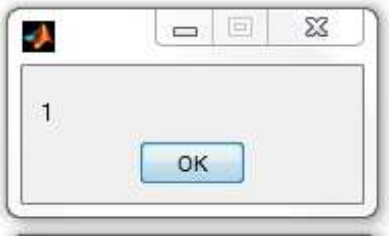


Fig. 4.6: Steps for Testing to Recognize a Handwritten Gurmukhi Numeral

The testing results of recognizing Gurmukhi numerals are shown in figures as given below:

```
>> final_DA_test

O =

    0.1309

>> final_DA_test
```

```
>> final_DA_test

O =

    1.0673
```




```
O =

    2.3936

>> final_DA_test

O =

    2.3936

>>
```

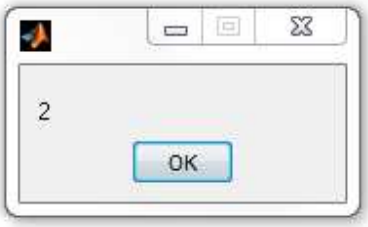
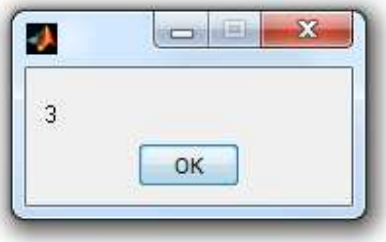
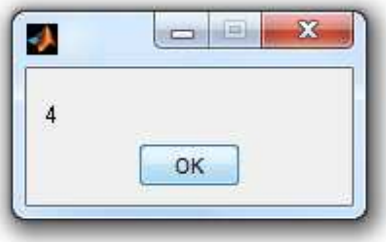


Fig. 4.7: Testing Output of 0, 1, 2 of Gurmukhi Numerals

```
>> final_DA_test

O =

    3.0825

>> final_DA_test

O =
```




```
>> final_DA_test

O =

    4.2345

>> final_DA_test
```



```
>> final_DA_test

O =

    5.0284

>> final_DA_test
```

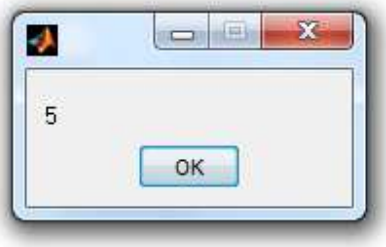
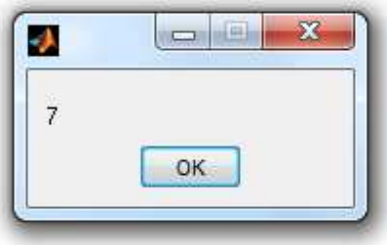


Fig. 4.8: Testing Output of 3, 4, 5 of Gurmukhi Numerals

```
>> final_DA_test

O =

    6.0270

>>
```


```
>> final_DA_test

O =

    7.1287

>> final_DA_test
```




```
>> final_DA_test

O =

    8.2565

>> final_DA_test
```

Fig. 4.9: Testing Output of 6, 7, 8, 9 of Gurmukhi Numerals

The figures given above reflect the output of 0 - 9 Gurmukhi numerals.

4.3 Performance Parameters

The performance parameters show the accuracy of the system in terms of its parameters like gradient, validation, learning rate and training.

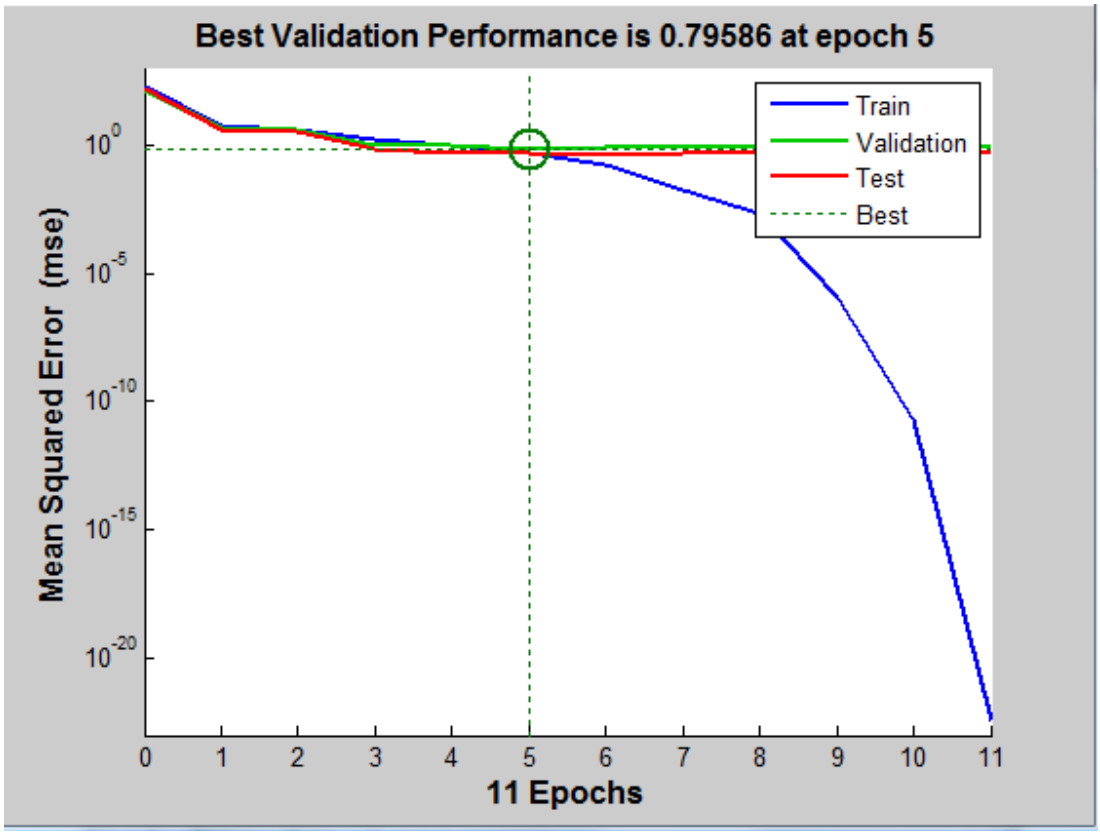


Fig. 4.10: Graph for Best Validation Performance at 5 Epochs

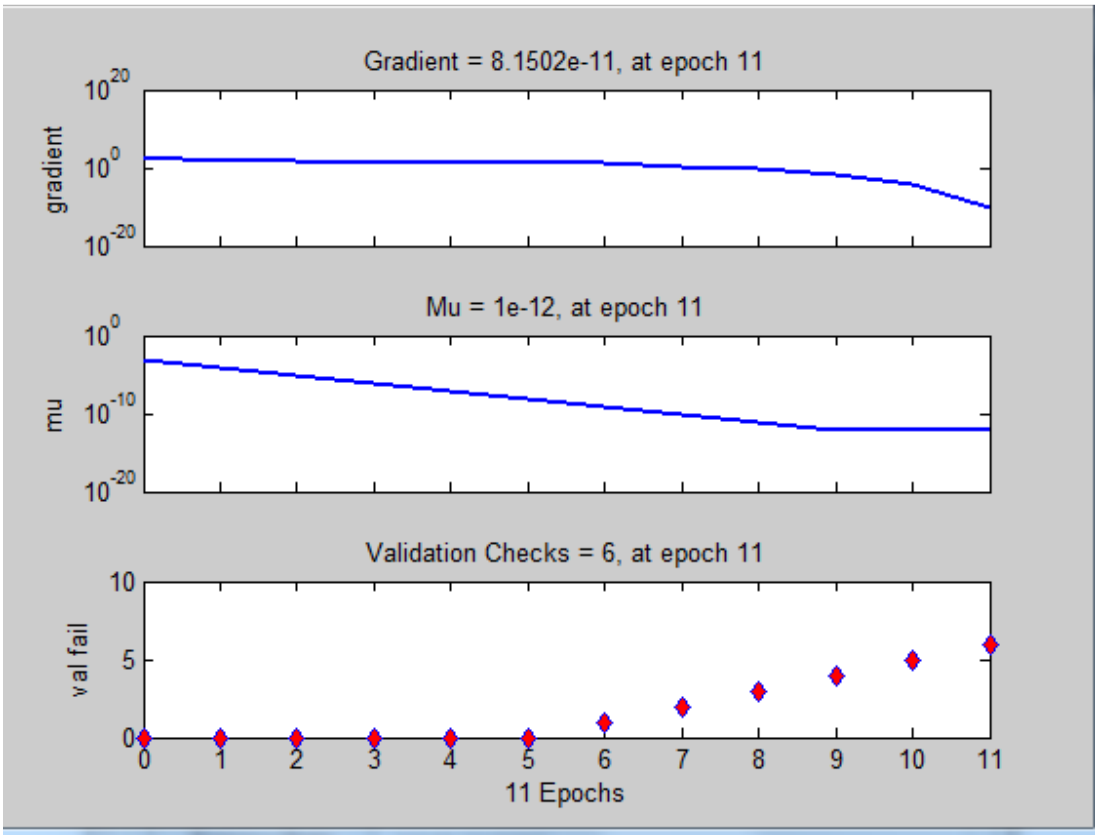


Fig. 4.11: Graphs Showing the Parameter Values

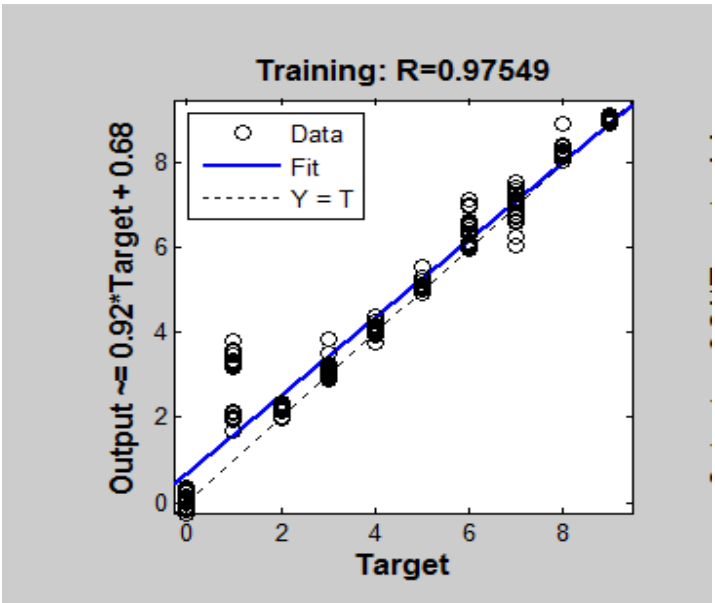


Fig. 4.12: Graph shows the Best Training Performance according to the target value

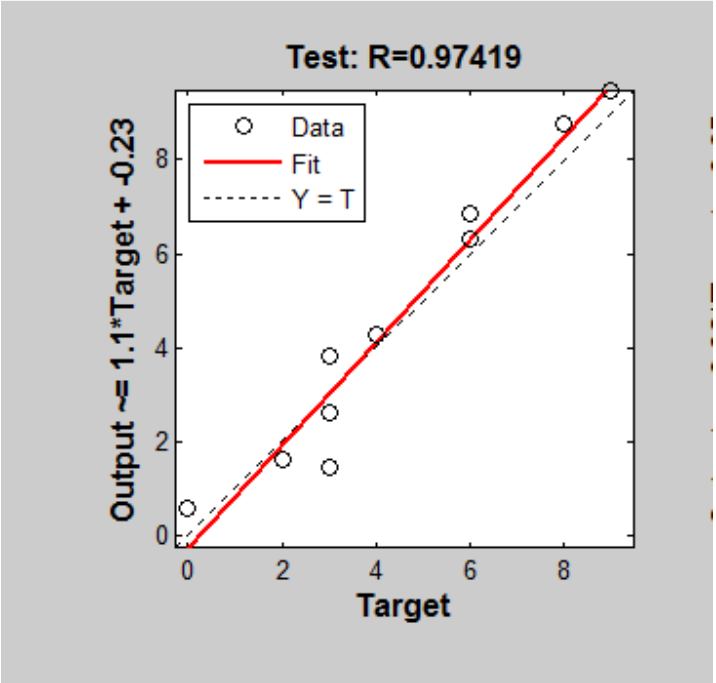


Fig. 4.13: Graph shows the Best Testing Performance according to the target value

4.4 Testing Results for Data of 100 Users

After training, the performance of network was tested by taking into account the data of 100 users to find the results which are exhibited in Table 4.5.

Table 4.5: Recognition Accuracy for Testing Data of Gurmukhi Numerals

S. No.	Size of training Data	Size of Testing Data	No. of Neurons in Hidden Layer	Recognition Accuracy (%)
1.	50	100	10	82.1
2.	50	100	20	85.0
3.	100	100	10	76.3
4.	100	100	20	82.6
5.	150	100	10	72.6
6.	150	100	20	88.1
7.	200	100	10	75.5
8.	200	100	20	89.6

Neural networks have here been used to solve the tasks which are otherwise difficult for the conventional computers or human beings. The recognition of Gurmukhi numerals has been done by using gradient descent based back propagation training algorithm. The main aim of using this algorithm is to reduce the error which is the difference between computed value of neural network and desired value. By using artificial neural network methodology, tested on 100 users' data of Gurmukhi numeral recognition, 89.6% accuracy has been achieved.

Chapter-5
Recognition of Special Characters using ANN

5.1 Special Characters Recognition

To recognize special characters (symbols), a database was organized which is shown in Table 5.1. Every special character has been given a reference number, *i.e.*, stroke-id. The following 20 symbols have been used in this work.

Table 5.1: Special Characters Database

S. No.	Stroke ID	Symbol	Description
1.	**20**		Hyphen
2.	**22**		Backslash
3.	**24**		Forward Slash
4.	**26**		Square Open Bracket
5.	**28**		Square Close Bracket
6.	**30**		Round Open Bracket
7.	**32**		Round Close Bracket
8.	**34**		Open Curly Bracket

Table 5.1 (Contd.)

9.	**36**		Close Curly Bracket
10.	**38**		Apostrophe
11.	**40**		At the Rate
12.	**42**		Comma
13.	**44**		Dot
14.	**46**		Less Than
15.	**48**		Greater Than
16.	**50**		Dollar Sign
17.	**52**		Tilde
18.	**54**		Caret
19.	**56**		Ampersand
20.	**58**		Question Mark

To carry out the recognition process, data has been collected from 20 users for these special characters. Each user contributed 20 samples of each special character. The samples have

been taken as *x* and *y* coordinates. We have taken 50, 100, 150 and 200 different users' samples for training the neural network. The output which appears in output datasheet is shown in Fig. 5.4.

5.2 Steps for Reorganization of Special Characters

The system automatically uses the intelligence of layered neural network to recognize the special characters. The neural network applied in the system utilizes the steepest descent algorithm with adaptive learning rate as the learning rule, which is used for the training of the neural network. The basic flow of the system is displayed below in Fig. 5.1.

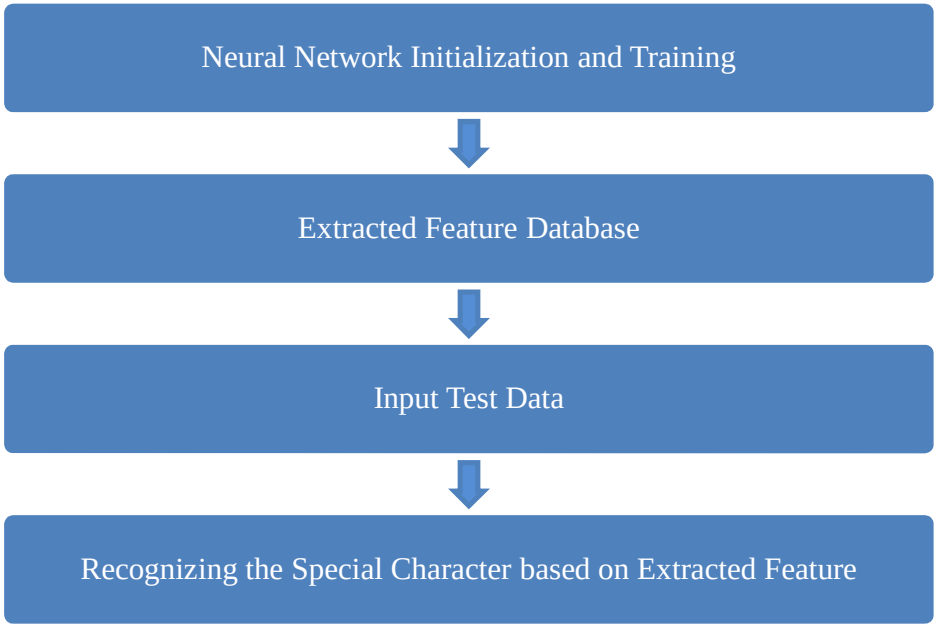


Fig. 5.1: Steps to Recognize a Handwritten Special Character

5.3 Neural Network Architecture

The objective of the neural network training was to find the connection between neurons which determine the global minimum error function. In this work, Levenberg-marquardt algorithm has been used to train the classifier for the classification of Human activity data from the accelerometer sensor. For the activity recognition, a fast training was supposed to be used. Therefore, the Levenberg-marquardt algorithm is used for training the neural

network. Finding the global minimum also requires selection of some parameters. MATLAB provides the default values for these parameters. The values of parameters used in this design are presented in Table 5.2 with a brief description of each parameter.

Table 5.2: Parameters Used for Neural Network Training for Special Character Recognition

Parameters	Value Description	Default Value Description
Input	64×64	Input data
Output	1	Output data
Epochs	1000	Maximum No. of epochs
Time	00:00:00	Minimum time
Performance	0.01	Minimum performance gradient
Hidden layer size	10, 20	No. of hidden layers
Mu	0.0001	Initial Mu
Validation checks	0	0

As mentioned earlier, the neurons are arranged into various layers: Input layer, Hidden layer and Output layer. A general architecture of an artificial neural network is shown in Figures 5.2 and 5.3. The neurons in each layer are fully interconnected by connection strengths called as weights. Also, each hidden and output layer neuron consists of a bias term associated with it. The designing of neural network involves the decision in number of hidden layers and number of neurons in each layer along with the transfer functions. In this work, MATLAB neural network has been used for implementation. The network architecture for implementation has been done with 10 and 20 neurons in the hidden layer. The performance of the neural network has been evaluated with the increasing number of hidden layers as well as each hidden layer neurons for the training as well as testing of data. Here, the database for training has been divided into two types. Initially, we trained the database with 10 neurons and then with 20 neurons.

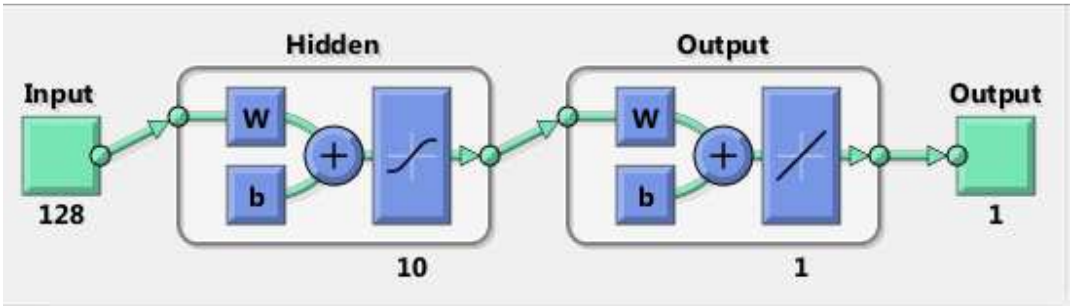


Fig. 5.2: Network Architecture for Implementation of Special Characters by Using 10 Neurons of ANN

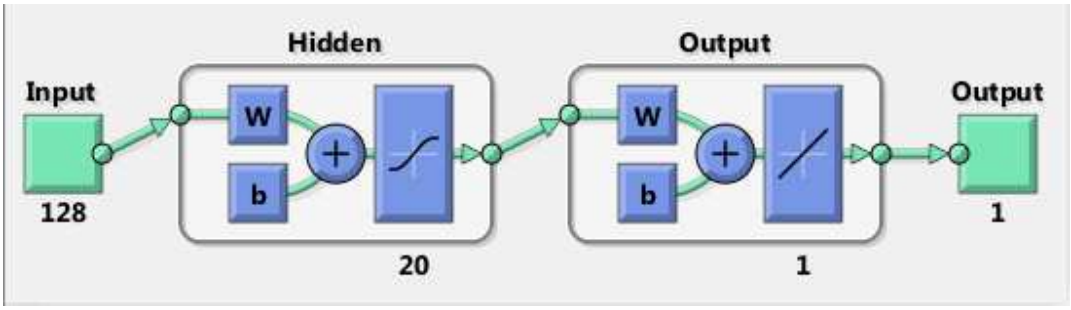


Fig. 5.3: Network Architecture for Implementation of Special Characters by Using 20 Neurons of ANN

5.4 Feature Extraction

Feature extraction is carried out to extract relevant features for recognition of special characters. Firstly, features are computed and extracted; and then most relevant features are selected to construct feature vector which is used for recognition (Gharde *et al.*, 2013). The computation of features is based on statistical, structural, directional, moment, and transformation like approaches. Feature extraction is extracting information from raw data which is more relevant for the classification purpose and that minimizes the variation within a class and maximizes the variations between classes.

5.4.1 Handwritten Input Datasheet for Special Characters' Recognition

The data has been inputed to ANN using the excel file. A portion of this file is given in Fig. 5.4. The values in this file are the features in the form of *x* and *y* coordinates of a stroke.

We have taken 50, 100, 150 and 200 different users' samples to train the neural network as shown in the Figure given below:

	FZ	GA	GB	GC	GD	GE	GF	GG	GH	GI	GJ	GK	GL	GM	GN	GO	GP	GQ	GR	GS
1	196	268	245	269	227	300	228	300	240	0	0	0	0	0	0	0	0	32	0	
2	300	278	300	280	269	250	259	252	214	20	10	20	19	21	10	0	0	0	39	
3	182	252	229	253	212	284	213	284	227	10	11	7	10	9	11	13	11	23	13	
4	294	276	298	280	268	250	262	257	211	16	9	15	18	18	10	0	2	3	35	
5	168	236	213	237	197	268	198	268	215	19	21	16	22	21	20	24	21	14	26	
6	288	276	298	280	268	250	267	262	209	14	9	10	18	17	10	0	6	5	34	
7	154	220	197	221	182	252	183	252	202	26	28	25	31	32	34	34	32	5	38	
8	283	275	298	280	268	254	271	268	206	10	9	8	17	14	10	0	9	9	31	
9	141	206	181	206	167	236	168	240	189	36	40	33	42	42	43	45	45	5	49	
10	277	271	290	280	268	258	276	271	205	8	10	7	17	11	10	0	9	11	29	
11	127	190	165	189	152	230	161	224	176	43	50	40	52	50	54	57	58	19	60	
12	272	268	285	280	268	260	279	270	203	6	9	0	13	9	10	0	8	18	28	
13	113	174	149	172	137	214	146	208	163	51	59	50	62	62	63	67	68	32	73	
14	269	268	279	280	267	264	278	270	195	0	8	0	11	10	10	0	8	22	26	
15	109	158	133	155	123	198	131	192	150	61	66	58	73	72	75	77	77	44	82	
16	255	267	277	280	258	267	278	270	188	0	8	0	8	8	10	0	8	22	23	
17	96	142	117	138	108	182	117	176	137	71	77	68	86	72	86	88	90	55	93	
18	241	260	272	280	257	268	273	270	189	0	8	0	8	0	10	0	10	22	20	
19	82	128	101	121	94	167	103	161	125	81	85	78	95	83	97	99	98	69	105	
20	228	245	258	279	250	260	268	267	176	0	5	0	8	0	10	0	1	22	20	
21	71	112	86	104	80	151	89	145	112	91	93	90	104	94	108	110	111	81	118	
22	215	240	253	271	245	259	257	261	163	0	0	0	8	0	10	0	0	22	19	
23	66	97	76	88	70	135	81	129	100	101	103	101	116	106	118	121	125	92	128	
24	205	225	238	267	232	255	250	260	151	0	0	0	9	0	9	0	0	21	17	
25	62	85	66	81	63	120	70	114	93	111	113	113	122	117	130	132	137	103	138	
26	197	210	233	258	217	250	235	251	139	0	0	0	1	0	10	0	0	20	10	

Fig. 5.4: Datasheet of Special Characters as Input

5.4.2 Output Datasheet of Special Characters' Recognition

The output corresponding to the data given in Fig. 5.4 is given in Fig. 5.5. These are the stroke ID assigned by us to different strokes.

	GA	GB	GC	GD	GE	GF	GG	GH	GI	GJ	GK	GL	GM	GN	GO	GP	GQ	GR	GS
1	56	56	56	56	56	56	56	56	58	58	58	58	58	58	58	58	58	58	
2																			
3																			
4																			
5																			
6																			

Fig. 5.5: Datasheet of Special Characters as Input

5.5 Neural Network Training

After obtaining the entire data, it was divided into two data sets. The training data is used for making the network learn from the data, and hence, setting the network weights and

biases using a training algorithm. The testing data is used to check the performance of the neural network. Classification of testing data indicates that how well the network generalizes the classification of new data. The complete data is divided into 3-ratios as 90% data for training, 5% data each for validation and testing. Out of the three folds, two were used as training data sets and the rest as the testing data. The training and testing process is repeated for all the possible combinations of the three fields. The training performance as calculated from the performance command is the best performance.

5.5.1 Training Results Pertaining to Special Characters' Recognizing

Table 5. 3: Recognition Accuracy for Training Data of Special Characters

S. No.	Size of Training Data	No. of Neurons in Hidden Layer	Recognition Accuracy (%)
1.	50	10	91.1
2.	50	20	93.0
3.	100	10	90.3
4.	100	20	96.0
5.	150	10	94.2
6.	150	20	97.1
7.	200	10	90.5
8.	200	20	95.6

The best accuracy of 94.2% has been achieved with 10 neurons of ANN, while it is 97.1% with 20 neurons.

5.5.2 Testing Process and Results Pertaining to Special Characters' Recognizing

The testing has been done on database which is given to trained neural network. The basic flow of testing the system is shown below in Fig.5.6.

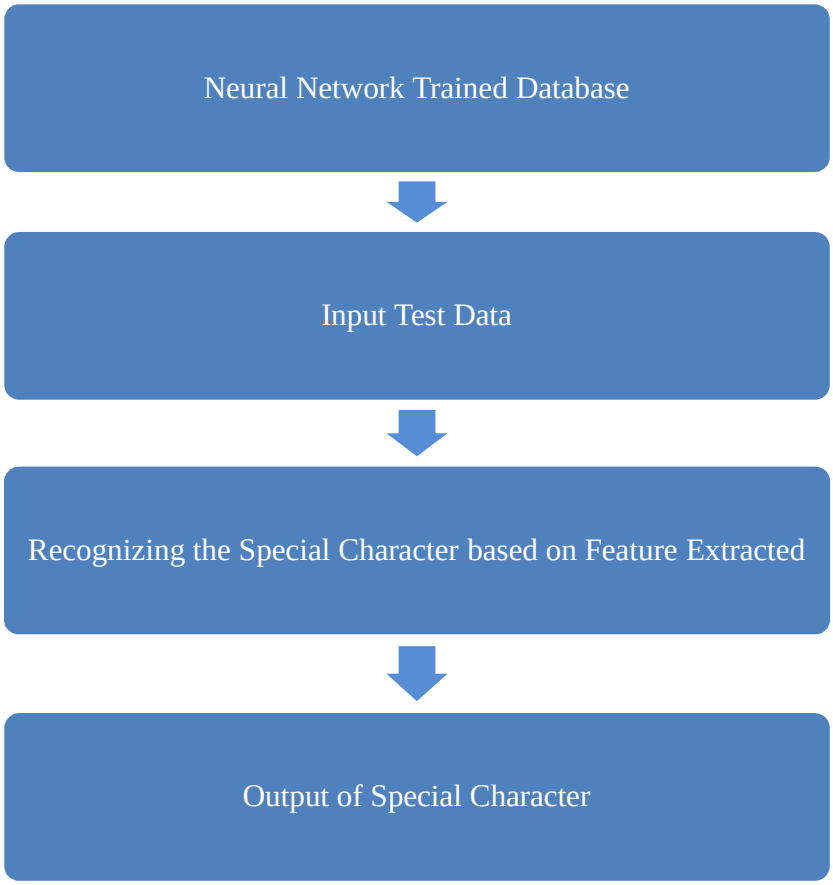
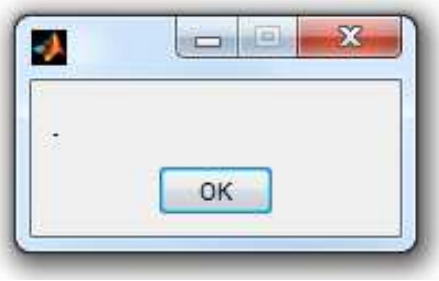


Fig. 5.6: Steps for Testing to Recognize a Special Character

The testing results pertaining to the recognition of Special Characters are produced below:

```
>> specialchartest

O =

   20.0000

>>
```



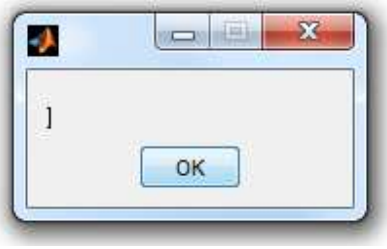
```
>> specialchartest

O =

    22

>>
```


```
>> specialchartest

O =

   24.0000

>>
```

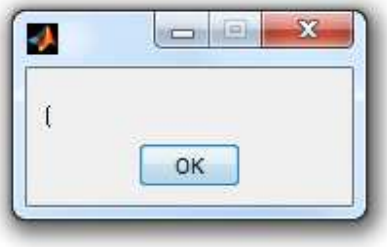
Fig. 5.7: Testing Output of -, \, / Symbols

```
>> specialchartest

O =

   26.0000

>>
```


```
>> specialchartest

O =

    28

>>
```




```
>> specialchartest

O =

   30.0000

>>
```



```
>> specialchartest

O =

    32

>>
```

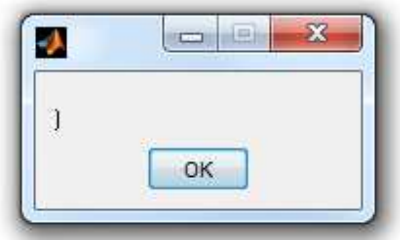
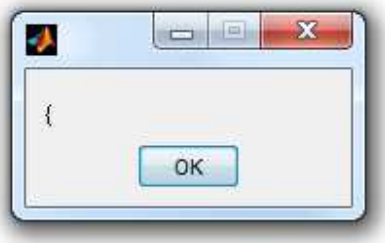


Fig. 5.8: Testing Output of [,], (,) Symbols

```
>> specialchartest

O =

   34.0000

>>
```



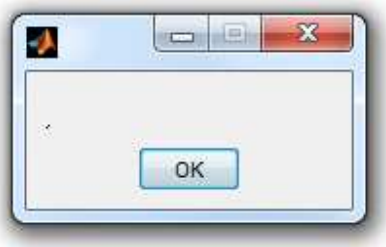

```
>> specialchartest

O =

   36.0000

>>
```


```
>> specialchartest

O =

   38.0000

>>
```

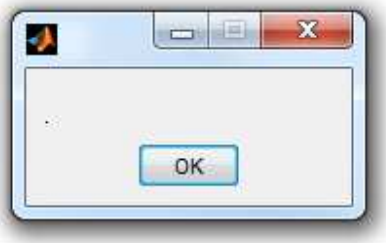

```
>> specialchartest

O =

   40.0137

>>
```

Fig. 5.9: Testing Output of {,}, Comma, @ Symbols

```
>> specialchartest

O =

   42.4936

>>
```



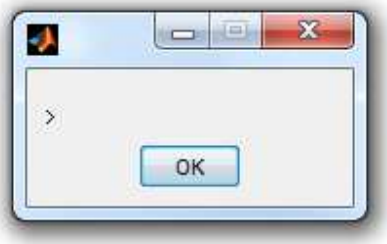

```
>> specialchartest

O =

   44.0356

>>
```




```
>> specialchartest

O =

   45.9427

>>
```

Fig. 5.10: Testing Output of ‘, ., < Symbols

```
>> specialchartest

O =

   48.0084

>>
```




```
>> specialchartest

O =

   49.9886

>>
```

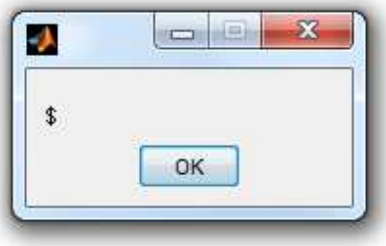


Fig. 5.11: Testing Output of >, $, ~ Symbols

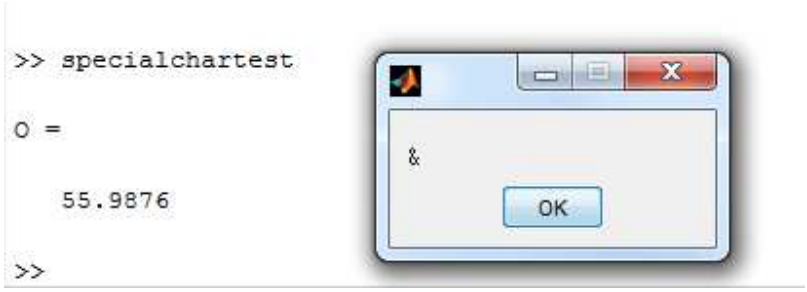


Fig. 5.12: Testing Output of ^, &, ? Symbols

The above given figures shows the output of the special symbols after testing

5.6 Performance Parameters

The performance parameters show the accuracy of the system in terms of its parameters like gradient, validation, learning rate and training.

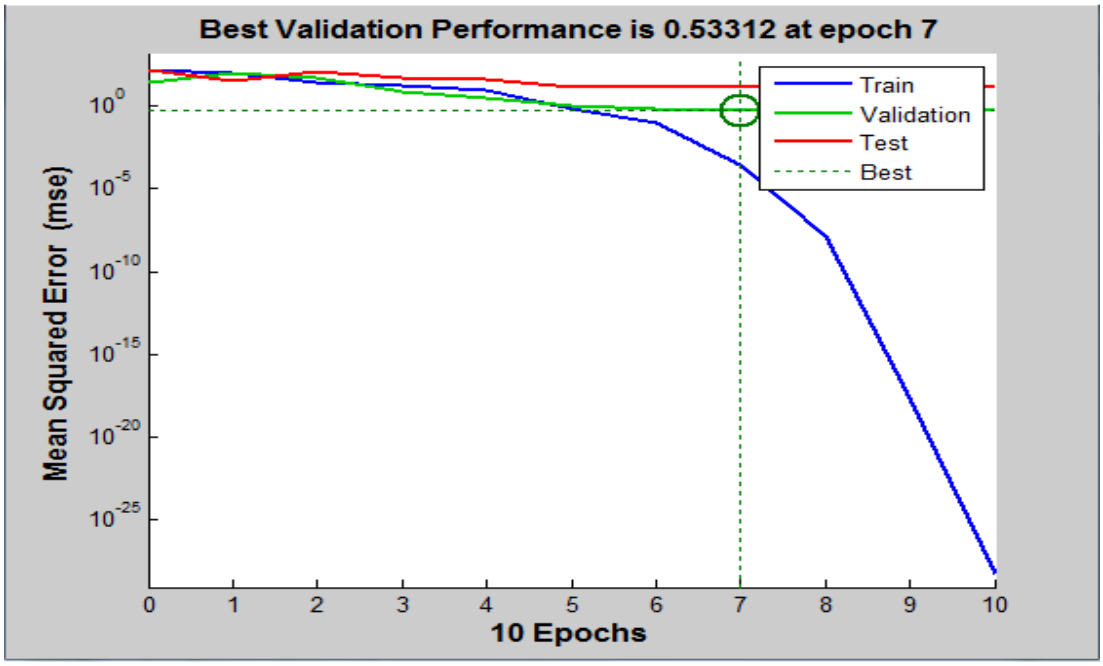


Fig. 5.13: Graph for Best Validation Performance at 7 epochs

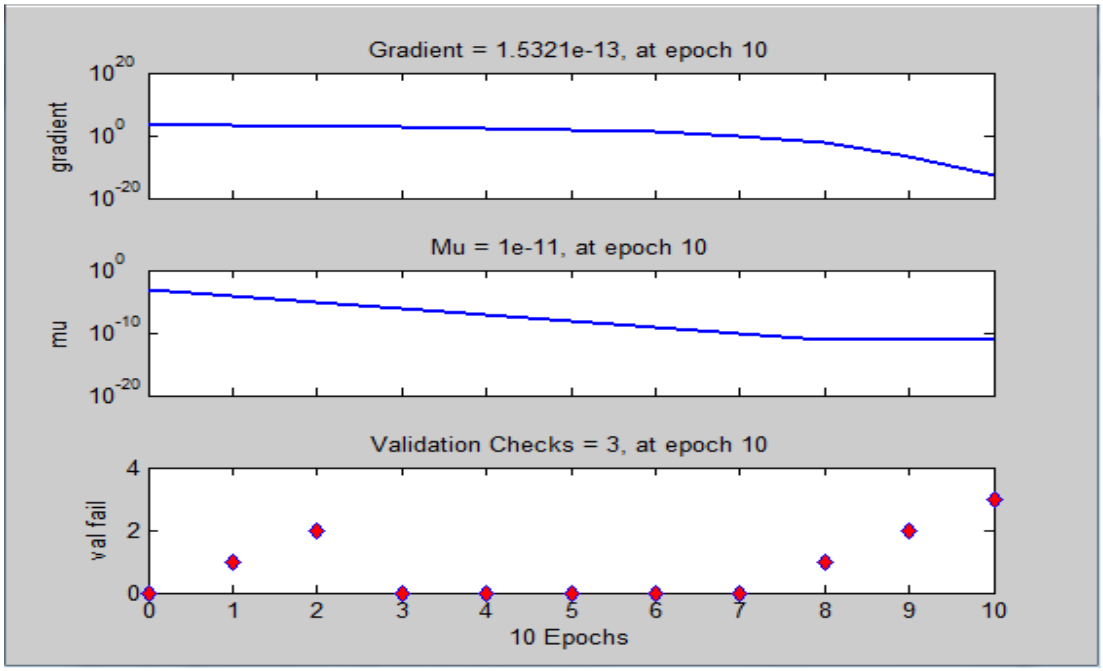


Fig. 5.14: Graphs Showing the Parameter Values

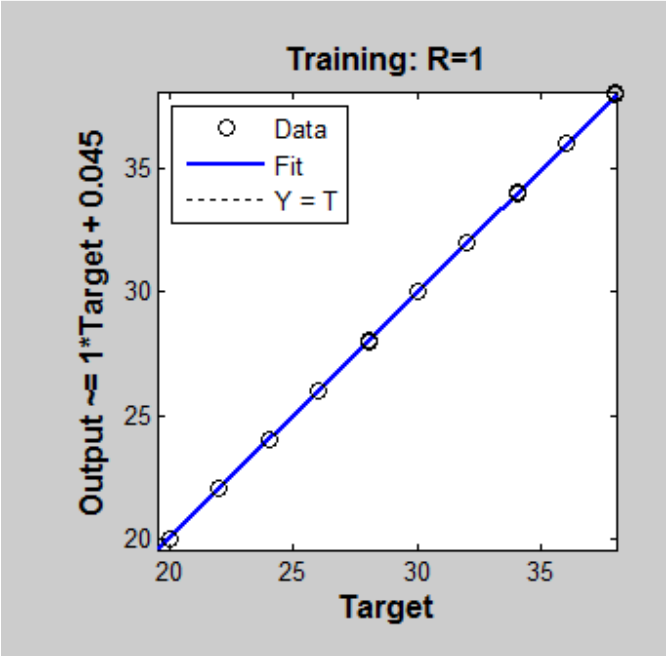


Fig. 5.15: Graph Shows the Best Training Performance according to Target Value

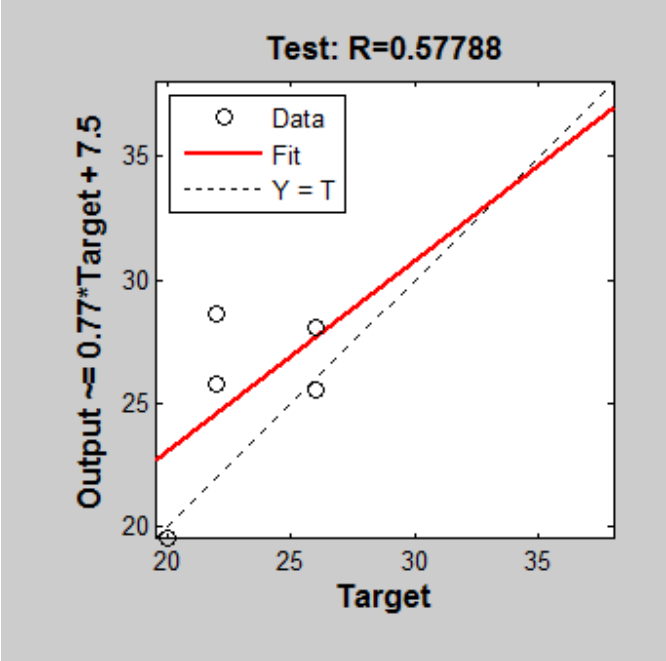


Fig. 5.16: Graph Shows the Best Testing Performance according to Target Value

5.7 Testing Results for Data by 100 Users

After training, the performance of whole database was tested with the help of data collected from 100 users. The results obtained after testing are presented in Table 5.4.

Table 5.4: Recognition Accuracy for Testing Data of Special Characters

S. No.	Size of Training Data	Size of Testing Data	No. of Neurons in Hidden Layer	Recognition Accuracy (%)
1.	50	100	10	90.0
2.	50	100	20	92.0
3.	100	100	10	81.3
4.	100	100	20	89.5
5.	150	100	10	87.4
6.	150	100	20	89.6
7.	200	100	10	81.6
8.	200	100	20	91.3

The recognition of handwritten special characters has been made by using gradient descent based back propagation training algorithm. The main aim of using this algorithm is to reduce the error which is the difference between computed value of neural network and desired value. The system developed in this work has achieved 92.0% accuracy in the recognition of special characters.

Chapter – 6
Conclusion and Scope for Further Research

6.1 Conclusion

No doubt, a lot of research work has been carried out on the subject under study, but most of it is available in the languages such as English, Japanese, Chinese, Arabic, Urdu, Devnagari and Tamil. However, similar work available in Punjabi language is not sufficient, and it requires further investigation. The present work is a modest attempt to develop a computerized system for the recognition of handwritten special characters and Gurmukhi numerals by using artificial neural networks in MATLAB. Neural networks are used to solve the tasks which are difficult for conventional computers or human beings to solve. The gradient descent based back propagation training algorithm has been used in the present system to recognize the handwritten Gurmukhi numerals.. The main aim of using this algorithm is to reduce the error which is the difference between computed value of neural network and desired value. By the use of artificial neural network methodology for the recognition of Gurumukhi numerals, 96.4% accuracy has been achieved; and it is 97.1% in the case of special characters' recognition. The data used for the databases of Gurmukhi numerals' recognition has been collected from 30 different people. Each writer contributed 10 samples of each numeral from zero to nine. Each writer contributed to write 20 samples of each special character. Each of these numerals and special characters has been given a unique ID. These samples have been transformed into *x*-coordinate and *y*-coordinate values. Among these samples, some distortions and irregularities are also incorporated. Then it is trained by neural network by using MATLAB. The whole data so obtained is divided into training and test data sets. The training data was used for making the network learn from the data statics, and hence, setting the network weights and biases by using a training algorithm. The classification of testing data gives an indication of how well the network generalizes the classification for new data. The whole data was divided into 3-ratios as 90% data for training, 5% data each for validation and testing. Out of the three folds, two were used as training data sets and the rest as the test data. The training performance as

calculated from the performance command is computed as 0.0573 which is the best performance for Gurumukhi numerals, while it is 0.6521 for special characters' recognition.

6.2 Scope for Further Research

- Every research work has its own limitations; and this study is also no exception in this regard. Limited scope, and time are some of the chief constraints of this study.
- In order to achieve a higher recognition rate, the size of user database should be increased. This can help in achieving better accuracy rate. Also, Gurmukhi alphabets can be included in the database for their recognition, thus, a Gurumukhi alphanumeric character recognizer can be developed.
- The work from stroke level recognition can be extended to character and word level recognition.
- The stroke database can be increased and examined for getting better recognition results.
- Some more features and pre-processing techniques can be introduced to further improve the recognition results.

References

[1] Ahranjany, S. S.; Razzazi, F.; and Ghassemian, M. H. (2010). "A Very High Accuracy Handwritten Character Recognition System for Farsi/Arabic Digits Using Convolutional Neural Networks", *Proceedings of IEEE International Conference on Neural Network*, Vol. 44, pp. 1585 - 1592.

[2] Arulmozhi, V. (2011), "Classification Task by Using MATLAB Neural Network Tool Box – A Beginner's View", *International Journal of Wisdom Based Computing*, Vol. 1, No. 2, pp. 59 – 60.

[3] Blumenstein, M.; Verma, B.; and Basli, H. (2003), "A Novel Feature Extraction Technique for the Recognition of Segmented Handwritten Characters", *International Journal of Computer Application on Neural Networks*, Vol. 1, pp. 137 – 141.

[4] Bontempi, B.; and Marcelli, A. (1994), "A Genetic Learning System for On-line Character Recognition", *Proceedings of 12th IAPR International Conference on Pattern Recognition*, Vol. 2, pp. 83 - 87.

[5] Dhir, R.; Singh, K.; and Rani, R. (2011), "Handwritten Gurmukhi Numeral Recognition Using Different Feature Sets", *International Journal of Computer Applications*, Vol. 28, No. 6, pp. 20 – 24.

[6] Dubey, R.; and Agrawal, D. K. (2012), "Comparative Analysis of Off-line Signature Recognition", *Proceedings of International Conference on Communication Information & Computing Technology*, India, pp. 1-6.

[7] Duneau, L.; and Dorizzi, B. (1994), "Online Cursive Script Recognition: A System that Adapts to an Unknown User", *Proceedings of* 12th *IAPR International Conference on Pattern Recognition*, Vol. 2, pp. 24 - 28.

[8] Eraqi, M.; and Sherif, A. (2011), "An On-Line Arabic Handwriting Recognition System", *Proceedings of IEEE International Conference on Document Analysis and Recognition*, pp. 409 - 413.

[9] Funanda, A.; Muramatsu, D.; and Matsumoto, T. (2004), "The Reduction of Memory and the Improvement of Recognition Rate for HMM On-line Handwriting Recognition", *Proceedings of IEEE 9th International Workshop on Frontiers in Handwriting Recognition*, pp. 383 - 388.

[10] Gharde, S. S.; Nemade, V. A.; and Adhiya, K. P. (2013), **"**Design and Implementation of Special Symbol Recognition System Using Support Vector Machine", *International Journal of Innovative Technology and Exploring Engineering*, ISSN: 2278-3075, Vol. 2, No. 6, pp.145 – 148.

[11] Goh, W. L.; Mittal, D. P.; and Babri, H. A. (1997), "An Artificial Neural Networks Approach to Hand-writing Recognition", *Proceedings of First IEEE International Conference on Knowledge-based Intelligent Electronic Systems*, Vol. 1, pp.132 – 136.

[12] Gumah, M. E.; Schneider, E.; and Aburas, A. (2010), "Handwriting Recognition System Using Fast Wavelets Transform", *Proceedings of IEEE 10th International Conference on Document Analysis and Recognition*, Vol. 1, pp.1 – 6.

[13] Jaremsri, M. L.; and Imprasert, Y. (2011), "Thai Handwritten Character Recognition Using Heuristic Rules Hybrid with Neural Network", *Proceedings of* 8th *International Joint Conference on Computer Science and Software Engineering*, pp. 160 - 165.

[14] Jun, W.; Xiaofeng, L.; and Zhang, Y. (2005), *Advances in Neural Networks*, Published by Springer, pp. 15 - 18.

[15] Kaur, A.; and Arora, M. (2013), "Neural Network-based Numerical Digits Recognization Using NNT in Matlab", *International Journal of Computer Science & Engineering Survey*, ISSN : 0976-2760, Vol. 4, No. 5, pp. 42 - 58.

[16] Kumar, M.; Jindal, M. K.; and Sharma, R. K. (2011), "Classification of Characters and Grading Writers in Offline Handwritten Gurmukhi Script", *Proceedings of International Conference on Image Information Processing*, pp. 1 - 4.

[17] Liu, X. Y.; and Blumenstein, M. (2004), "Experimental Analysis of the Modified Direction Feature for Cursive Character Recognition", *Proceedings of IEEE 9th International Workshop on Frontiers in Handwriting Recognition*, pp. 353 – 358.

[18] Liwicki, M.; Scherz, M.; and Bunke, H. (2006), "Word Extraction from On-Line Handwritten Text Lines", *Proceedings of IEEE 18th International Conference on Pattern Recognition*, Vol. 2, pp. 929 – 933.

[19] Mahmud, J. (2005), "An Intelligent Feature Analyzer for Handwritten Character Recognition", *International Journal of Computer Applications on Intelligent Agents, Web Technologies and Internet Commerce*, Vol. 2, pp. 763 – 769.

[20] Maureen, C.; and Caudill, M. (1990), "Neural Network Primer, Part 1", A1 Report, December 1987, Hecht-Nielsen, *Neuro Computing*, Published by Addison-Wesley, pp. 46 – 52.

[21] Pal, U.; Wakabayashi, T.; and Kimura, F. (2009), **"**Comparative Study of Devnagari Handwritten Character Recognition using Different Features and Classifiers", *Proceedings of IEEE 10th International Conference on Document Analysis and Recognition*, ISSN : 1520-5363, pp. 1111 – 1115.

[22] Rajashekararadhya, S. V.; and Ranjan, P. V. (2010), "The Zone-based Projection Distance Feature Extraction Method for Handwritten Numeral/Mixed Numerals Recognition of Indian Scripts", *Proceedings of IEEE 12th International Conference* on *Frontiers in Handwriting Recognition*, pp. 617 - 622.

[23] Rajput, G. G.; and Mali, S. M. (2010), "Fourier Descriptor Based Isolated Marathi Handwritten Numeral Recognition," *International Journal of Computer Applications*, Vol. 3, No. 4, pp. 9 - 13.

[24] Ritu; Sangita; and Saroha, V. (2013), "Neural Networks", *International Journal of Current Engineering and Technology*, ISSN 2277– 4106, Vol.3, No.2, pp. 38 - 52.

[25] Sanmorino, A.; and Yazid, S. (2012), "A Survey for Handwritten Signature Verification", *Proceedings of IEEE International Conference on Uncertainty Reasoning and Knowledge Engineering*, pp. 54 - 57.

[26] Sharma, A.; Kumar, R.; and Sharma R. K. (2009), "Rearrangement of Recognized Strokes in Online Handwritten Gurmukhi Words Recognition", *IEEE 10th International Conference on Document Analysis and Recognition*, pp. 1241 - 1245.

[27] Sharma, A.; Young, D. L.; and Wan, Y. C. (2008), "High Accuracy Human Activity Monitoring using Neural network", *Proceedings of IEEE 3rd International Conference on Convergence and Hybrid Information Technology*, Vol.12, No.1, pp. 229 – 232.

[28] Shokoohi, Z.; Hormat, A. M.; Mahmoudi, F.; and Badalabadi, H. (2013), "Persian Handwritten Numeral Recognition Using Complex Neural Network and Non-linear Feature Extraction", *Proceedings of 1st Iranian Conference on Pattern Recognition and Image Analysis*, pp. 1 - 5.

[29] Tsuruoka, S.; Fadzil, Mohd. T.; Kawanaka, H. T.; and Yasuji, M. (2010), "Personal Dictionaries for Handwritten Character Recognition", *Proceedings of IEEE 12th International Conference* on *Frontiers in Handwriting Recognition Using Characters Written by a Similar Writer*, pp. 599 - 604.

[30] Verma, B. (1998), "A Feature Extraction Technique in Conjunction with Neural Network to Classify Cursive Segmented Handwritten Characters", *Proceedings of IEEE World Congress on Computational Intelligence*, Vol. 1, pp. 332 - 336.

[31] Werbos, P. J.; (1994), *The Roots of Back Propagation: From Ordered Derivatives to Neural Networks*, Published by John Wiley & Sons, Inc., pp. 256 – 317.

[32] Xue, D.; and Chen, Y. (2009), *Solving Applied Mathematical Problems with MATLAB*, Published by Chapman & Hall/CRC, pp. 114 - 128.

[33] Yusuf, Perwez (2012), "Recurrent Neural Network Method in Arabic Words Recognition System", *International Journal of Computer Science and Telecommunications*, Published by Sysbase Solution (Ltd.), UK, London, Vol. 3, No. 11, pp. 43 – 48.