Optimal Feed Forward MLPArchitecture for Off-Line Cursive Numeral Recognition

*Amit Choudhary**
Deptt. of Comp. Sc.
Maharaja Surajmal Institute,
New Delhi, India.
amit.choudhary69@gmail.com

Rahul Rishi
Deptt. of Comp. Sc.and Engg.
TITS, Bhiwani,
Haryana, India.
rahulrishi@rediffmail.com

Savita Ahlawat
Deptt. of Comp. Sc.and Engg.
Maharaja Surajmal Institute of Technology,
New Delhi, India.
savita.ahlawat@gmail.com

Vijaypal Singh Dhaka
Deptt. of Comp. Sc.
IMS, Noida,
UP, India.
vijaypal.dhaka@gmail.com

***Abstract*—The purpose of this work is to analyze the performance of back-propagation feed-forward algorithm using various different activation functions for the neurons of hidden and output layer and varying the number of neurons in the hidden layer. For sample creation, 250 numerals were gathered form 35 people of different ages including male and female. After binarization, these numerals were clubbed together to form training patterns for the neural network. Network was trained to learn its behavior by adjusting the connection strengths at every iteration. The conjugate gradient descent of each presented training pattern was calculated to identify the minima on the error surface for each training pattern. Experiments were performed by selecting different combinations of two activation functions out of the three activation functions logsig, tansig and purelin for the neurons of the hidden and output layers and the results revealed that as the number of neurons in the hidden layer is increased, the network gets trained in small number of epochs and the percentage recognition accuracy of the neural network was observed to increase up to certain level and then it starts decreasing when number of hidden neurons exceeds a certain level.**

Keywords- Numeral Recognition, MLP, Hidden Layers, Backpropagation, Conjugate Gradient Descent, Activation Functions.

I. INTRODUCTION

The process of offline numerals recognition involves the automatic conversion of numeral's gray scale/ binary images obtained from paper documents, photographs etc. into digit codes that are interpreted by the computer and other text-processing applications. The OCR technology has importance in various fields like Banking System, Postal Department etc to recognize courtesy amount on bank checks [2] and to recognize postal pin codes written on post cards / letters. The automated processing of handwritten material optimizes the processing speed as compared to human processing and shows high recognition accuracy and achieves high benefits in monetary terms.

Recognition of handwritten numerals is a very complex problem. The digits could be written in different dimension, thickness or slant [3]. These will give unlimited variations. The capability of neural network to generalize and its robust nature would be very beneficial in recognizing handwritten digits.

The artificial neural network structure involved in our experiment of handwritten numeral recognition is a multi layered perceptron (MLP) with one hidden layer which is a supervised learning network in nature. A MLP is a feed-forward neural network that uses error back-propagation algorithm for its training [5]. It is a modification of the standard linear perceptron in that it uses one or more hidden layers of neurons with non-linear activation functions and is more powerful than the perceptron in that it can classify data that is not linearly separable or separable by a hyper plane.

The experiments conducted in this paper have shown the effect of the number of neurons in the hidden layer and the transfer function used in the hidden layer and output layer on the learning and recognition accuracy of the neural network. The criterion to judge the performance of the network will be the number of training epochs, the MSE (Mean Square Error) value and the percentage recognition accuracy. All other experimental conditions such Learning Rate, Momentum Constant (α), Maximum Epochs Allowed, Acceptable Error Level and Termination Condition were kept same for all the experiments.

The rest of the paper is organized as follows: Section II deals with the overall system design and the various steps involved in the OCR System [4]. Neural Network Architecture is presented in detail in section III. Section IV provides various experimental conditions for all the experiments conducted under this work. Functioning of the proposed system is explained in Section V. Discussion of Results and

* Corresponding Author

interpretations are described in section VI. Section VII presents the conclusion and also gives the future path for continual work in this field.

II. Overall System Design

A typical handwritten numeral recognition system is characterized by a number of steps, which include (1) Digitization / Image Acquisition, (2) Preprocessing, (3) Image Binarization (4) Network Training (5) Classification / Recognition and (6) Testing. Fig. 1 illustrates various steps involved in a handwritten digit recognition system.

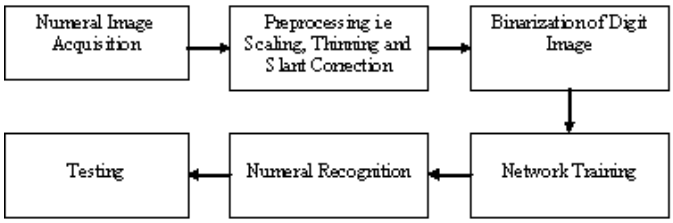

Figure 1. Typical Off-Line Handwritten Numeral Recognition System

A. Preprocessing

Preprocessing aims at eliminating the variability that is inherent in cursive and hand-printed numerals. Scaling, Noise Elimination, Slant Estimation and Correction and Contour Smoothing are some preprocessing techniques that have been employed in an attempt to increase the performance of the recognition process. Scaling sometimes may be necessary to produce numeral images of equal size. Noise (small dots or blobs) may be introduced easily into an image during image acquisition; therefore, these small foreground components are usually removed. S. Srihari, V. Govindaraju and R. Srihari also analyzed the size and shape of connected components in a numeral image and compared them to a threshold to remove the noise [1]. Slant estimation and correction is an integral part of any numeral image preprocessing. The slope can be estimated through analysis of the slanted vertical projections at various angles. Contour Smoothing [7] is a technique to remove contour noise that is introduced in the form of bumps and holes due to the process of slant correction.

B. Binarization of Numeral Images

All hand printed numerals are scanned into gray scale images. Each numeral image is traced vertically after converting the gray scale image into binary matrix. This binarization is done using logical operation on gray intensity level as followes:

If (I >= Threshold)

I = 0

Else

I = 1

where 0 < Threshold <= 1

This threshold parameter is based on the gray-scale intensity of the text in the document. More intensity leads to the more threshold value. This parameter is decided as shown in Table 1:

TABLE I. Intensity / Threshold Comparison Table

Gray-Scale Intensity of the Text (I)	Threshold
0 – 0.25	0.20
0.26 – 0.50	0.45
0.51 – 0.75	0.65
0.76 – 1.0	0.85

The first column of Table-1 represents the intensity of handwritten digit present in the document. This intensity is a gray-scale value when the document is saved after scanning in gray scale image format and the second column of this table represents the corresponding value of threshold level for binarization process [6]. The threshold parameter along with the grayscale image is made an input to the binarization program designed in MATLAB. The output is a binary image shown in Fig. 2(a).

C. Reshape and Resizing of Numerals for Sample Creation

Each digit is first converted into a binary form and then resized to 10x10 matrix M using nearest neighborhood interpolation and reshaped to a 100x1 logical vector so that it can be presented as input to the network for learning. The matrix is reshaped [8] using following algorithm:

Algorithm: ReshapeCharacter(M,100,1)

1. Find the transpose of Character's binary matrix M.

2. Read every element vertically i.e. column wise.

3. Save it in the new matrix according to its dimensions in row first manner.

4. Repeat step 2-3 until every element is read.

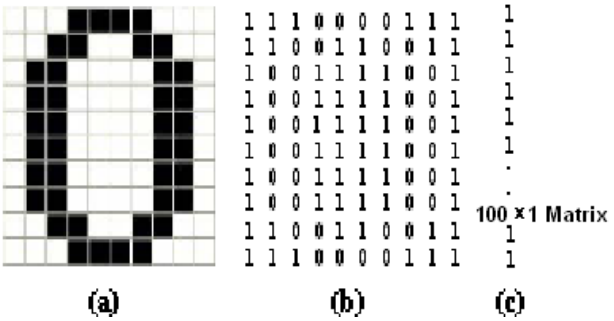

Figure 2. (a) Binary Representation of Digit '0'; (b) Binary Matrix representation and (c) Reshaped sample of Digit '0'.

The output of this algorithm is a binary matrix of size 100x1 which is made as an input to the neural network for learning and testing. Binary matrix representation of digit '0' can be defined as in Fig 2(b). The Resized digits were clubbed together in a matrix of size (100, 10) to form a **SAMPLE** as shown in Fig. 3.

So the final samples for network training/testing were made as follows:

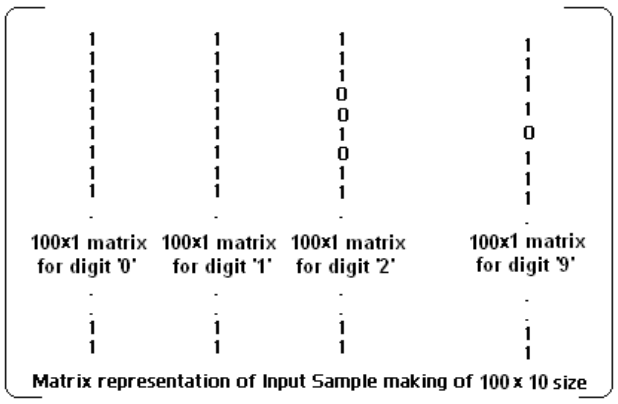

Figure 3. Input training / testing sample preparation

For sample creation, 250 numerals were gathered form 35 people of different ages including male and female. After the preprocessing is over, 5 samples were considered for training such that each sample was consisting of 10 digits (0-9) and 3 samples were considered for testing the recognition accuracy of the network.

III. Neural Network's Architecture Used in The Recognition Process

The architecture selected for the neural network is a Multilayer Perceptron (MLP) with one hidden layer. The network has 100 input neurons that are equivalent to the input numeral's size as we have resized every numeral into a binary matrix of size 10 X 10. The 10 output neurons correspond to 10 digits. The number of hidden neurons is directly proportional to the system resources. The bigger the number more the resources are required.

The processing nodes of input layer used the liner activation function and the nodes of hidden and output layers used the non-liner differentiable activation function as shown in Fig 4.

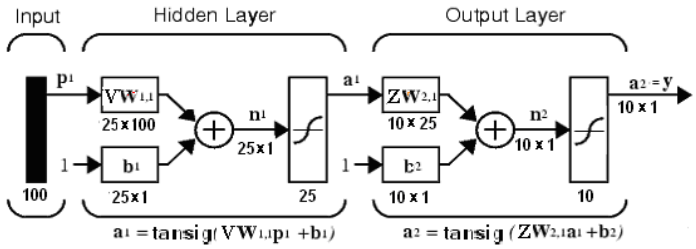

Figure 4. Architecture of the Neural Network used in the Recognition System.

The output of the network can be determined as:

$$y_k = f(\sum_{i=1}^{n} Z_i W_{ik}) \quad (3.1)$$

Where f is the output function,

Z_i is the output of hidden layer and

W_{ik} is the weight between hidden and output layer.

And , also for hidden layer's processing unit output ;

$$Z_i = f(\sum_{j=1}^{n} V_{ij} X_j) \quad (3.2)$$

Where X_j is the output of input layer and V_{ij} is the weight between input and hidden layer.

The MSE between the desired and actual output of the network is given by:

$$E = 0.5\sum_{k}[t_k - y_k]^2 \quad (3.3)$$

Where t_k is desired output

The error minimization can be shown as;

$$\partial E / \partial W_{ik} = [t_k - y_k] f'(y_k) z_i \quad (3.4)$$

Weights modifications on the hidden layer can be defined as;

$$\Delta V_{ij} = \partial E / \partial V_{ij} = \sum_{k} \partial k * (\partial y_k / \partial v_{ij})[t_k - y_k] f'(y_k) z_i \quad (3.5)$$

[Let, $\partial k = [t_k - y_k]\ f'(y_k)$]

and we have

$$\Delta V_{ij} = \sum_{k} \partial k * W_{ik} f'(z_i) \quad (3.6)$$

Thus the weight updates for output unit can be represented as;

$$W_{ik}(t+1) = W_{ik}(t) + \eta\, \Delta W_{ik}(t) + \alpha\, \Delta W_{ik}(t-1) \quad (3.7)$$

Where

$W_{ik}(t)$ is the state of weight matrix at iteration t

$W_{ik}(t+1)$ is the state of weight matrix at next iteration

$W_{ik}(t-1)$ is the state of weight matrix at previous iteration.

$\Delta W_{ik}(t)$ is current change/ modification in weight matrix and

α is standard classical momentum variable to accelerate learning process and depends on the learning rate of the network.

The neural network was exposed to 5 different samples as achieved in section III, each sample being presented 100 times. Actual output of the network was obtained by "COMPET" function. This is a competitive transfer function which works as follows:

Function Compet (net N)

Start:

[S, Q] = Size (N)

(N being network simulated vector)

[Maxn, indn] = max (N, [ ], 1)

Result Matrix A = Sparse (indn, 1: Q, Ones (1, Q), S, Q)

End Function.

This "COMPET" function puts 1 at the output neuron in which the maximum trust is shown and rest neuron's result into '0' status. The output matrix is a binary matrix of size (10, 10). The output is of the size (10×10) because each digit has 10×1 output vector. First 10×1 column stores the first digit's recognition output, the following column will be for next digit and so on for 10 digits. For each digit the 10×1 vector will contain value '1' at only one place. For example digit '0' if correctly recognized, will result in [1, 0, 0, 0 …all …0].

The difference between the desired and actual output is calculated for each cycle and the weights are adjusted by the equation (3.7). This process continues till the network converges to the allowable error or till the maximum number of training epochs is reached.

IV. EXPERIMENTAL CONDITIONS

The various parameters and their respective values used in the learning process of all the experiments with various activation functions of the neurons in hidden and output layers and having different number of neurons in the hidden layer are shown in Table 2.

TABLE II. EXPERIMENTAL CONDITIONS OF THE NEURAL NETWORK

Parameters	Value
Input Layer	
No. of Input neurons	100
Transfer / Activation Function	Linear
Hidden Layer	
No. of neurons	Between 5 and 95
Transfer / Activation Function	LogSig or Tansig or Purelin
Learning Rule	Momentum
Output Layer	
No. of Output neurons	10
Transfer / Activation Function	LogSig or Tansig or Purelin
Learning Rule	Momentum
Learning Constant	0.01
Acceptable Error (MSE)	0.001
Momentum Term (α)	0.90
Maximum Epochs	1000
Termination Conditions	Based on minimum Mean Square Error or maximum number of epochs allowed
Initial Weights and biased term values	Randomly generated values between 0 and 1
Number of Hidden Layers	1

V. FUNCTIONAL DETAILS

The neural network structure is determined by the number of neurons in the input, hidden and output layers. The network performance is also influenced by the activation functions of the hidden and output layer neurons. The number of neurons in the input and output layer of the network depends on the type of the problem. In the current situation, the number of neurons in the input and output layers are fixed at 100 and 10 respectively. The number of neurons in the hidden layer and the activation functions of the neurons in the hidden and output layers are to be decided.

It is very difficult to determine the optimal number of hidden neurons. Too few hidden neurons will result in underfitting and there will be high training error and statistical error due to the lack of enough adjustable parameters to map the input-output relationship. Too many hidden neurons will result in overfitting and high variance. The network will tend to memorise the input-output relations and normally fail to generalize. Testing data or unseen data could not be mapped properly.

Each neuron in the neural network has a transformation function. To produce an output, the neuron performs the transformation function on the weighted sum of its inputs. Various transfer functions used in neural networks are compet, hardlim, logsig, poslin, purelin, radbas, satlin, softmax, tansig and tribas etc. The three transfer functions normally used in MLP are logsig, tansig and purelin.

logsig transfer function is also known as Logistic Sigmoid:

$$\log sig(x) = \frac{1}{1+e^{-x}},$$

tansig transfer function is also known as hyperbolic tangent:

$$\tan sig(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

The hyperbolic tangent and logistic sigmoid are related by:

$$\frac{\tan sig(x) + 1}{2} = \frac{1}{1+e^{-2x}}$$

We get:

$$\tan sig(x) = \frac{2}{1+e^{-2x}} - 1$$

purelin transfer function is also known as simple linear transfer function and produces the same output as its input.

$$purelin(x) = x$$

These transfer functions are commonly used as they are easy to use mathematically and while saturating, they are close to linear near the origin.

There is not any rule that gives the calculation for the ideal parameter setting for a neural network. In our problem, the structure analysis has to be carried out to find the optimum number of neurons in the hidden layer and the best activation functions for the neurons in the hidden and output layers.

For this analysis, the activation function of the neurons in the hidden layer was selected as logsig and the activation function of the neurons in the output layer was changed and the process is repeated by selecting different number of neurons in the hidden layer as shown in Table 1. Similarly, as shown in Table 2 and Table 3, the activation function of the hidden layer neurons was selected as tansig and purelin respectively and the activation function is changed for output layer neurons with various different numbers of neurons in the hidden layer.

Mean Square Error (MSE) is an accepted measure of the performance index often used in MLP networks. The lower value of MSE indicates that the network is capable of mapping the input and output accurately. The accepted error level is set to 0.001 and the training will stop when the final value of MSE reaches at 0.001 or below this level.

The number of epochs required to train a network also indicates the network performance. The adjustable parameters of the network will not converge properly if the number of training epochs is insufficient and the network will not be well trained. On the other hand, the network will take unnecessary long training time if there are excessive training epochs. The number of training epochs should be sufficient enough so as to meet the aim of training.

If there is a saturated or horizontal straight line at the end of the MSE plot, the further training epochs are no longer beneficial and the training should be stopped by introducing the stopping criterion such as maximum number of training epochs allowed in addition to the stopping criterion of maximum acceptable error as specified in the training algorithm. This type of MSE plot is observed when there is a very complicated problem or insufficient training algorithm or the network have very limited resources.

VI. Results and Discussion

Three types of transfer functions: ‘logsig’, ’tansig’ and ’purelin’ are used to simulate the neural network used in our problem of numeral recognition. As seen in Table 3, the activation function for the output layer was changed to logsig, tansig and purelin one by one by fixing the activation function of hidden neurons to ‘logsig’. The number of training epochs and MSE values of the network are indicated corresponding to the number of neurons in the hidden layer. Similarly in Table 4 and Table 5, the activation function of the hidden neurons was fixed as ‘tansig’ and ‘purelin’ respectively.

If we ignore the type of activation function used in hidden layer and output layer, it is clear from Table 3, Table 4 and Table 5, that, as the number of neurons in the hidden layer is increased, the number of epochs required for the training of the network decreases. Insufficient number of hidden neurons results in underfitting and the MSE does not fall below the acceptable error level and the training is stopped by the stopping criterion of maximum allowed number of training epochs. Overfitting is observed when the number of hidden neurons exceeds a certain count and the percentage recognition accuracy gradually falls with the increase in the number of hidden neurons. The network will tend to memorise the input-output relations and normally fail to generalize. However the training of the network becomes faster as indicated by the decrease in number of training epochs as the number of hidden neurons increases.

The optimal number of hidden neurons and the best combination of activation functions for the hidden and output neurons can be decided by considering the numeral recognition accuracy of the network. By detailed analysis of Table3, Table 4 and Table 5, it is clear that the good recognition accuracy is observed when the selected combination of activation function for hidden and output neurons were logsig-tansig, logsig-pureline, tansig-tansig and tansig-pureline.

By taking into consideration all the performance measure parameters such as MSE, overfitting and underfitting, training time and the available system resources, it is evident from Table 2 that the neural network with 25 neurons in the hidden layer and with tansig-tansig activation function combination for the hidden and output neurons provides the best sample recognition accuracy of 99.1% with least MSE of 0.000216485 and sufficient number of training epochs. For this network, the profile of MSE plot for the training epochs is drawn in Fig 5

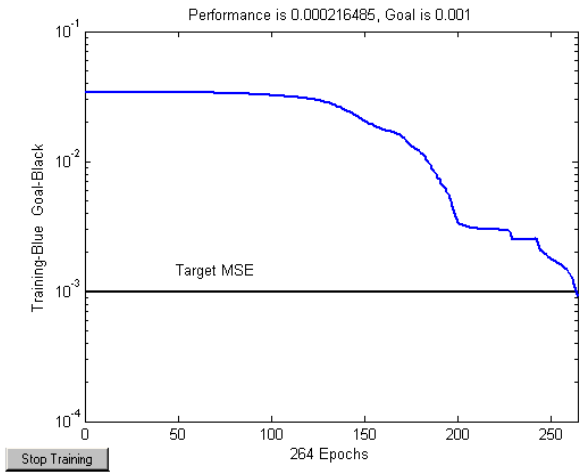

Figure 5. The variation of MSE with the training epochs

VII. Conclusion And Future Scope

The MLP used in the proposed method for the handwritten numeral recognition employing the back propagation algorithm performed exceptionally well with 25 neurons in the hidden layer and tansig as the activation function for both hidden and output layer neurons. In this optimal MLP structure, the number of neurons in the input and output layers were already fixed to 100 and 10 respectively.

Thus it can be concluded that if the accuracy of the results is a critical factor for an application, then the network having the optimal structure should be used but if training time is a critical factor then the network having a large number of hidden neurons should be used. Training speed can be achieved at the cost of system resources because the number of training epochs decreases with the increase in the number of hidden neurons.

More work need to be done especially on the test for more complex handwritten numerals. The proposed work can be carried out to recognize numeral strings after proper segmentation of the digits into isolated digit images.

REFERENCES

[1] S Srihari, V Govindraju, R Srihari. "Handwritten text recognition. Proc 4th International Workshop on frontiers in handwriting recognition", Taiwan, 265-275, 1994.

[2] Neves, R.F.P., et al. 2008. A New Technique to Threshold the Courtesy Amount of Brazilian Bank Checks. In Proceedings of the 15th IWSSIP, (Bratislava, Slovak Republic, June 2008), IEEE Press, in press.

[3] S. Ouchtati, B. Mouldi, A. Lachouri,"Segmentation and Recognition of Handwritten Numeric Chains". In Journal of Computer Science 3 (4) ISSN 1549--3636, 242—248, 1997.

[4] N. Arica and F. Yarman-Vural, "An Overview of Character Recognition Focused on Offline Handwriting", IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol. 31, No.2, pp. 216--233, 2001.

[5] B. K. Verma, "Handwritten Hindi Character Recognition Using Multilayer Perceptron and Radial Basis Function Neural Network", IEEE International Conference on Neural Network, vol. 4, pp. 2111--2115, 1995.

[6] R. Ilin; R. Kozma; P. J. Werbos. Beyond feedforward models trained by backpropagation: A practical training tool for a more efficient universal approximator. *IEEE Transactions on Neural Networks*, 19(6):929--937, June 2008.

[7] An OCR System for Telugu, Proceedings of the Sixth International Conference on Document Analysis and Recognition, p.1110, September 10-13, 2001

[8] N. P. Banashree, R. Vasanta,"OCR for Script Identification of Hindi (Devnagari) Numerals using Feature Sub Selection by Means of End-Point with Neuro-Memetic Model. In Proceedings of World Academy of Science, Engineering and Technology, vol. 22 ISSN 1307-6884, 78—82, 2007.

[9] B. Burton, R.G. Harley, G. Diana, and J.R. Rodgerson, "Reducing the Computational Demands of Continually Online- Trained Artificial Neural Networks for System Identification and Control of Fast Processes" IEEE transaction on Industry Applications,Vol 34. no.3, May/June 1998, pp. 589-596.

TABLE III. BEHAVIOR OF MLP WITH LOGSIG FUNCTION IN HIDDEN LAYER

Number of Hidden Units	Hidden Layer- Output Layer Logsig- Logsig			Hidden Layer- Output Layer Logsig -Tansig			Hidden Layer- Output Layer Logsig -Pureline		
	MSE	Epochs	Recognition Accuracy	MSE	Epochs	Recognition Accuracy	MSE	Epochs	Recognition Accuracy
5	0.008928986	1000	78.2 %	0.000850345	590	95.5 %	0.008828986	1000	82.0 %
10	0.009418455	1000	81.7 %	0.000888548	579	96.9 %	0.009495754	1000	85.4 %
15	0.003887096	1000	77.7 %	0.000614491	490	96.6 %	0.007185188	1000	80.0 %
20	0.000552677	850	91.2 %	0.000534798	403	98.0 %	0.00614034	1000	79.4 %
25	0.00081461	831	92.9 %	0.000224105	367	98.6 %	0.005683699	1000	77.5 %
30	0.000587808	859	93.5 %	0.000319418	380	97.6 %	0.000965726	773	90.4 %
35	0.000517400	770	92.3 %	0.000341597	352	98.0 %	0.000675745	672	93.1 %
40	0.000562353	690	91.9 %	0.000449759	368	98.3 %	0.0007857	699	93.4 %
45	0.000557294	600	91.4 %	0.000540665	292	97.3 %	0.000763599	580	91.1 %
50	0.000777322	539	92.0 %	0.000739418	205	97.0 %	0.000876958	521	93.7 %
55	0.000802615	550	91.4 %	0.000614805	246	96.3 %	0.000776376	511	92.6 %
60	0.000528738	622	89.9 %	0.000630529	230	95.7 %	0.000576766	556	90.9 %
65	0.000623401	503	90.5 %	0.000657159	258	94.6 %	0.000678987	509	90.3 %
70	0.000550557	467	90.8 %	0.000795910	240	95.0 %	0.00062571	460	91.2 %
75	0.000685151	459	91.1 %	0.000688535	187	94.3 %	0.000869587	373	89.7 %
80	0.000851809	404	89.2 %	0.000899568	134	94.3 %	0.000785595	373	89.0 %
85	0.00073169	442	88.7 %	0.000825945	101	92.8 %	0.000962565	340	90.0 %
90	0.000803686	373	87.9 %	0.000740504	167	93.6 %	0.000878538	322	87.2 %
95	0.000807035	391	86.6 %	0.000934694	159	93.1 %	0.000957101	343	87.3 %

TABLE IV. BEHAVIOR OF MLP WITH TANSIG FUNCTION IN HIDDEN LAYER

Number of Hidden Units	Hidden Layer- Output Layer Tansig - Logsig			Hidden Layer- Output Layer Tansig –Tansig			Hidden Layer- Output Layer Tansig -Pureline		
	MSE	Epochs	Recognition Accuracy	MSE	Epochs	Recognition Accuracy	MSE	Epochs	Recognition Accuracy
5	0.008659145	1000	77.7 %	0.000737171	441	96.5 %	0.009634384	1000	83.1 %
10	0.009659549	1000	79.8 %	0.000623718	366	97.1 %	0.008247478	1000	82.6 %
15	0.003457529	970	91.4 %	0.000416114	299	96.7 %	0.008746174	1000	83.0 %
20	0.000954238	850	92.3 %	0.000313748	310	97.9 %	0.000466938	883	94.3 %
25	0.000518571	831	94.0 %	0.000216485	264	99.1 %	0.000534352	847	93.9 %
30	0.000569569	859	92.5 %	0.00039264	270	98.7 %	0.000976004	861	94.2 %
35	0.000549511	770	93.0 %	0.000351614	251	98.1 %	0.00083439	733	94.1 %
40	0.000655709	690	92.8 %	0.000423734	222	98.2 %	0.000634753	699	93.7 %
45	0.000558188	600	91.1 %	0.000612344	194	97.7 %	0.000826441	592	93.1 %
50	0.000666341	539	92.6 %	0.000517227	209	96.8 %	0.000974747	533	92.6 %
55	0.000734692	550	90.1 %	0.000577998	188	97.4 %	0.000816411	501	92.9 %
60	0.00058623	622	89.4 %	0.000467775	160	97.3 %	0.00051246	573	91.7 %
65	0.000516414	503	89.5 %	0.000514644	139	95.7 %	0.000625785	555	91.0 %
70	0.000564327	467	89.0 %	0.000514081	145	96.8 %	0.000714237	461	90.8 %
75	0.000682111	459	87.8 %	0.000618745	129	96.3 %	0.000846144	477	91.6 %
80	0.000725695	404	87.3 %	0.000518473	119	94.6 %	0.000827494	390	90.5 %
85	0.000628575	442	88.7 %	0.000623715	134	94.9 %	0.00071676	360	90.2 %
90	0.00075955	373	87.9 %	0.00071415	102	95.1 %	0.000619455	297	89.9 %
95	0.000593765	391	87.1 %	0.000824999	117	94.4 %	0.000551288	338	90.5 %

TABLE V. BEHAVIOR OF MLP WITH PURELIN FUNCTION IN HIDDEN LAYER

Number of Hidden Units	Hidden Layer- Output Layer Pureline - Logsig			Hidden Layer- Output Layer Pureline –Tansig			Hidden Layer- Output Layer Pureline –Pureline		
	MSE	Epochs	Recognition Accuracy	MSE	Epochs	Recognition Accuracy	MSE	Epochs	Recognition Accuracy
5	0.008817411	1000	69.5 %	0.000850345	945	88.1 %	0.00757252	1000	65.3 %
10	0.008141479	1000	70.3 %	0.000888548	869	87.4 %	0.009255902	1000	66.7 %
15	0.009164144	1000	71.1 %	0.000614491	749	89.6 %	0.008477877	1000	71.9 %
20	0.00729585	1000	70.2 %	0.000534798	821	92.3 %	0.006137051	1000	67.3 %
25	0.004556234	1000	73.6 %	0.000597450	833	91.9 %	0.007147814	1000	71.8 %
30	0.000634022	901	88.8 %	0.000319418	738	92.0 %	0.006347055	1000	72.4 %
35	0.000694905	845	87.9 %	0.000341597	725	93.6 %	0.004985955	1000	69.6 %
40	0.000647446	765	86.7 %	0.000449759	620	92.7 %	0.003799632	1000	70.0 %
45	0.000737051	688	87.4 %	0.000540665	601	91.9 %	0.000710525	863	85.5 %
50	0.000782752	605	86.5 %	0.000739418	554	92.0 %	0.000915644	771	83.6 %
55	0.000723695	593	85.9 %	0.000614805	527	91.3 %	0.00082542	690	84.8 %
60	0.00054666	534	86.0 %	0.000630529	488	90.8 %	0.000786612	664	84.9 %
65	0.000512492	578	85.7 %	0.000657159	519	90.2 %	0.000715646	660	82.5 %
70	0.000491464	488	85.1 %	0.000795910	431	89.7 %	0.000514746	612	81.9 %
75	0.000359686	437	86.1 %	0.000688535	389	90.5 %	0.000772598	631	82.0 %
80	0.000712545	441	83.9 %	0.000499568	392	89.5 %	0.000824675	569	80.6 %
85	0.000925552	423	84.3 %	0.000825945	368	89.7 %	0.000575385	588	80.2 %
90	0.000625695	444	83.1 %	0.000740504	323	88.4 %	0.000715734	561	81.4 %
95	0.000358255	409	82.7 %	0.000734694	335	88.9 %	0.00053625	532	79.5 %