

# Neural Network Based Handwritten Digits Recognition- An Experiment and Analysis

M.J. Islam, Q. M.J. Wu, M. Ahmadi, M.A. Sid-Ahmed

**Abstract**— Handwritten digit recognition has become very useful in endeavors of human/computer interaction. Reliable, fast, and flexible recognition methodologies have elevated the utility. This paper presents an experiment and analysis of the Neural Network classifier to recognize handwritten digits based on a standard database. The experimental setup implemented in Matlab determines the ability of a Multi-Layer Neural Network to identify handwritten digit samples 5-9. This network is the representative for recognition of remaining digits 0-4. We consider not only accurate recognition rate, but also training time, recognition time as well as the complexity of the networks. The Multi-Layer Perceptron Network (MLPN) was trained by back propagation algorithm. Network structures vary with the hidden units, learning rates, the number of iterations that seem necessary for the network to converge. Different network structures and their corresponding recognition rates are compared in this paper to find the optimal parameters of the Neural Network for this application. Using the optimal parameters, the network performs with an overall recognition rate 94%.

**IndexTerms**— Handwritten Digits, Multi-Layer-Perceptron Neural Network, Network Architecture.

## I. INTRODUCTION

Speech is a sign system that is more natural than writing to humans. However, writing is considered to have made possible much of culture and civilization. The written form of language is contained in printed documents, such as newspapers, magazines and books, and in handwritten matter, such as found in notebooks and personal letters. Given the importance of written language in human transactions, its automatic recognition has practical significance. Great strides have been achieved in pattern recognition specifically in the area of handwritten digit recognition in recent years. This rapid progress has resulted from a combination of number of developments including the proliferation of powerful, inexpensive computers, the invention of new algorithms that take advantage of these computers, and the ability of large database of characters that can be used for training and testing [1]. Though not a significant enough

solution in the area of pattern recognition, one realm in which we think our system would be useful is mail sorting such as automatic zip code reading on letters, which is currently in use by the U.S. Postal Service [2].

The practical application of handwritten digit recognition and our interest motivated us to develop a neural network capable of identifying the digits 5-9. Our main intention is to implement a MLPN and to test the level of accuracy we could achieve with a single hidden layer of neurons. We assumed that back propagation training and a single hidden layer structure would yield high accuracy once we were able to provide the network with inputs containing the maximum amount of information about the digits 5-9. Since the success of a neural network depends on the different parameters like hidden units, learning rates, the number of iterations that seem necessary for the network to converge, optimizing the network structure was the main focus of this work.

The remainder of this paper is organized as follows: Section II explains about the experimental handwritten data set, section III discusses about network architecture and training mechanism. Matlab simulation and performance of the results are discussed in section IV and V respectively and finally, section VI concludes the paper.

## II. EXPERIMENTAL DATASET

The handwritten data set was obtained from the university of Wisconsin- Madison central repository [3]. The data file, called digits5-9.dat, is an ASCII file containing 100 examples, one per line. Each example is a comma-separated (without any white space) list of 65 integer values, the first 64 specifying the input and the last value specifying the digit which is the desired output. The input values are integers in the range [0..16]. Some examples of handwritten digits are shown in figure 1. We normalize the input values by converting them to real number in the range [0..1] by dividing every value by 16.0. This is useful because the derivative of the sigmoid function is often very close to 0, which can cause the network to converge very slowly to a good set of weights. The desired output digit is converted to a target output vector for the five output units. For example, if the digit is a "5" then the target vector [0.9 0.1 0.1 0.1 0.1]. This set of target output values is preferred because the sigmoid function cannot produce the exact output values of 0 and 1 using finite weights, and so the weight values may get very large and causing overflow problems. Final numbers are 64 pixels of size 8x8 with 16 gray levels per images. Even in the limited set of examples shown, the variety of written numbers is apparent. This variety is the root of one of the challenges of automatic handwriting recognition [4].

Manuscript received on September 26, 2008.

M.J. Islam is a doctoral student of Electrical and Computer Engineering Department, University of Windsor, Windsor, ON N9B3P4, Canada.

Q.M.J. Wu is the Professor of Electrical and Computer Engineering Department, University of Windsor, Windsor, ON, N9B3P4, Canada

M. Ahmadi is the University Professor of Electrical and Computer Engineering Department, University of Windsor, Windsor, ON, N9B3P4, Canada.

M.A. Sid-Ahmed is the Professor and Head of Electrical and Computer Engineering Department, University of Windsor, Windsor, ON, N9B3P4, Canada

### III. ARTIFICIAL NEURAL NETWORK

Artificial Neural Network (ANN) is a powerful classifier that represents complex input/ output relationships. It resembles human brain in acquiring knowledge through learning and storing knowledge within inter-neuron connection strengths. Some part of this section is adapted from [5]. Commonly, ANN's synaptic weights are adjusted or trained so that a particular input leads to a specific desired or target output. Figure 2 shows the block diagram for a supervised learning ANN, where the network is adjusted based on comparing neural network output to the desired output until the network output matches the desired output. Once the network is trained it can be used to test new input data using the weights provided from the training session.



Fig. 1: Sample Handwritten Digits

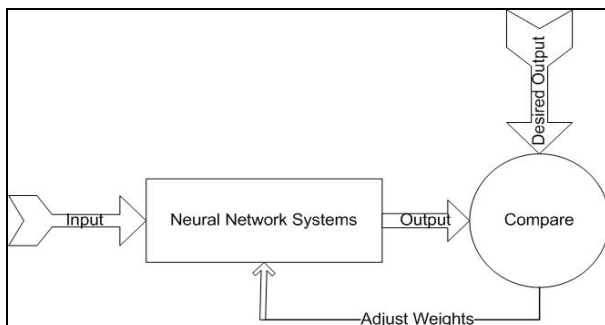


Fig. 2: Supervised Learning of ANN

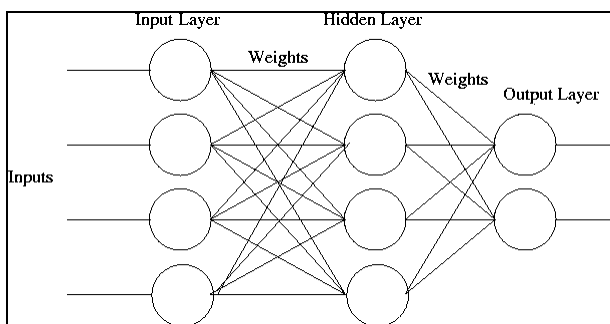


Fig. 3: Multi-Layer Feed-Forward Neural Network

#### F. The Multi-Layer Feed-Forward Network Model

Figure 3 represents the structure of a multi-layer feed-forward network. The neurons in this model are grouped in layers, which are connected to the direction of passing

signal. There are no lateral connections within each layer and no feed-backward connections within the network.

Multi-Layer perceptron (MLP) is the most common neural network model. It is a hierarchical structure of several perceptron that uses supervised training methods to train the network. The training of such a network with hidden layer is complicated. That's why when there exists an output error; it is hard to know how much error comes from the input nodes, other nodes and how to adjust the weights according to their contributions [5]. The problem can only be solved by finding the effect of all the weights in the network. This is solved by the back-propagation algorithm which is a generalization of the least-mean-square (LMS) algorithm.

MLPs contain three layers: the input layer, hidden layers and output layer that is obvious from the figure 3. The input nodes and the hidden nodes are connected via variable weights using feed-forward connections. The output of the hidden layer nodes is connected to the input of the output layer nodes via weights. Details of the back-propagation can be found [6, 7]. The calculated output is compared with the target output. The total mean square error (MSE) is computed using all training patterns of the calculated and target outputs are as follows:

$$MSE = \sum_{j=1}^m \frac{1}{2} \sum_{i=1}^5 (T_{ij} - O_{ij})^2 \quad (1)$$

Where  $m$  is the number of examples in the training set,  $T_{ij}$  is the target output value (either 0.1 or 0.9) of the  $i^{th}$  output unit for the  $j^{th}$  training example, and  $O_{ij}$  is the actual real-valued output of the  $i^{th}$  output unit for the  $j^{th}$  training example.

### IV. MATLAB SIMULATION

To recognize the digits 5, 6, 7, 8 and 9 correctly the input image is provided of size 8x8, given as a row-major ordered sequence of pixel brightness values, which are integers in the range [0..16]. Thus the input layer will have 64 units. The output layer will have five units, one for each of the possible classifications of the input. The goal is to train the network so that if the input image contains a "5", for example, then the first output unit's activation should be near 1 and all the other output unit's activations should be close to 0. Similarly, if the input image contains a "6" then the second output unit's value should be near 1, etc. For example, hidden layer has 10 units. Each unit in the input layer is connected to every unit in the hidden layer, and each unit in the hidden layer is connected to every unit in the output layer that is shown in figure 3. So the entire network contains  $((64 + 1) * 10) + ((10 + 1) * 5) = 705$  weights. The extra "+1"s in this formula is because each unit in the hidden layer and in the output layer also has an associated bias, which is treated as an extra weight with constant input value -1. Each unit in the hidden layer and the output layer computes the sigmoid function. This function returns a real value in the range [0..1], not a binary value as the linear threshold unit does. Sigmoid function refers to the special case of logistic function and is defined by the equation 2 and is shown in figure 4.

$$f(x) = \frac{1}{1 + e^{-x}} \quad (2)$$

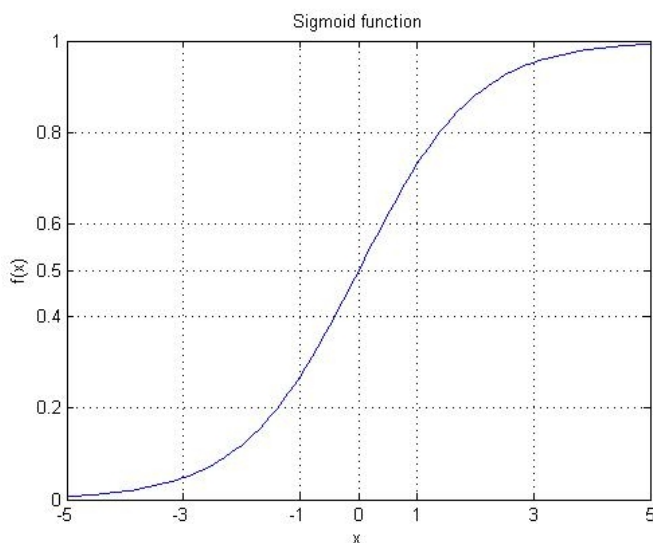


Fig. 4: Sigmoid function

To satisfy the requirements, a 5-fold cross-validation experiment is implemented to evaluate the performance of the network. To do this, the input file is divided into five parts, each containing  $100/5=20$  examples. For each of the five runs, 4 parts are used as the training set and the 5th part is used as the test set. The training of the network is continued at least for 200 epochs of the training set or until the MSE is 0.2 or less than that [8].

Once the network is trained using the training set of 80 examples in each run, the testing phase begins its recognition. In this phase a set of 20 examples are used to obtain the correct percentage of outputs. To compare the performance and to obtain the optimal set of parameters, the training and testing steps are repeated for different values of hidden units and learning rates. In each case the percentage correct output, training time, testing time, number of iterations is recorded and finally we obtain the optimal set of parameters for this application using the dataset.

## V. RESULTS AND PERFORMANCE ANALYSIS

To fulfill the objectives of this work rigorous methodological steps are followed. The first requirement is to implement the back-propagation algorithm to recognize the handwritten digits like 5, 6, 7, 8 and 9. To do this all the weights are initialized in the network to random values in the range [-1.0, 1.0]. The learning rate  $\alpha = 0.2$  is selected and the number of hidden units is 10. So the overall structure of the network is shown in Table I.

Table I: Neural Network structure

Input units, $X_i$	64
Hidden units, $Z_i$	10
Output units, $Y_i$	5
Learning rate, $\alpha$	0.2

The above network is trained using 4 parts of the data (total 80 examples) and the 5th part is used for testing. The total training and testing is continued for 5 times. So, total number

of training dataset becomes 400 and testing dataset becomes 100. The MSE curve corresponding to the epoch in each run is shown in figure 5. The percentage correct output, training time, testing time, and the number of iterations required to converge to  $MSE=0.2$  in each run and the average percentage correct output, average training time, average testing time, and average iterations required is shown in Table II from the experiment.

Now, to obtain the optimal set of parameters different network structures are evaluated varying the number of hidden units and learning rate. For example, by keeping the learning rate  $\alpha = 0.2$  unchanged, the hidden units are varied and the network is evaluated for the values of 5, 10, 15, 20, 25, and 50. The result for percentage correct output and iterations are shown in Table III and IV respectively.

From the Table III and IV, it is clear that in terms of percentage correct output, and iterations required to converge to  $MSE=0.2$ , the optimal result is found for hidden units 15. Based on that optimal value of hidden units 15, next step is to select the learning rate for which best recognition rate can be obtained. To do this hidden units are kept unchanged at  $h=15$  and learning rates  $\alpha$  are varied in the range of [0.05-0.30], incrementing each time by 0.05. The network is evaluated for each structure. The result for percentage correct output and iterations are shown in table V and VI respectively for the above structure varying the learning rate ( $\alpha$ ). It was found that if we decrease the learning rate, the network becomes too slow, although in our case it does not show any significant improvement in recognizing correct output. So eventually the training time and the number of iterations increased. At the same time, if the learning rate is increased the network is trained too fast, but there is a chance to skip the global minima.

From Table V and VI, it is obvious that in terms of percentage correct output, and number of iterations required to get the  $MSE=0.2$  or lower, the learning rate  $\alpha = 0.25$  provides the best result. Finally the network is evaluated using the best values for  $h=15$  and  $\alpha = 0.25$  and the corresponding training plot (Epoch vs. MSE) is shown in figure 6 and the outputs are shown in Table VII.

Finally the comparative statement of our proposed optimal network to the other methods like linear classifiers and neural networks [2] with different hidden units are shown in Table VIII.

Table VIII: Performance of the Proposed Network Structure

Classifier	Hidden Units	Error Rate (%)
Linear Classifier [2]	-	12.0
Neural Network [2]	300	4.7
Neural Network [2]	1000	4.5
<b>NN (Proposed)</b>	<b>15</b>	<b>6.0</b>

From table VIII, the performance of the proposed optimal network is obvious. Linear classifier has huge error rate. Although the neural network has the error rate 4.7% and 4.5%, but the number of hidden units are used 300 and 1000 respectively that definitely increases the recognition time. In

our proposed optimal network only 15 hidden units are used which is very low compare to other neural network methods and the correct recognition rate is 94%.

## VI. CONCLUSIONS

Network performance depends on many factors including high accuracy, low run time, low memory requirements, and reasonable training time. In this paper, in addition to accuracy, we look at measures the training time, testing time and number of iterations required to converge to a specific MSE like 0.2 in our case. In this paper, back-propagation learning algorithm was successfully applied to a large, real world task. Our results appear to be at the state of the art in handwritten digit recognition. Back-propagation was implemented for the 64 input units, different hidden units and 5 output units to recognize 5 handwritten digits like 5, 6, 7, 8 and 9. This algorithm can be easily implemented for recognizing 10 digits (0-9) by doing some minor modifications. The network parameters hidden units, learning rate plays an important role for obtaining high percentage of correct output. In this experiment hidden units  $h=15$  and learning rate  $\alpha =.25$  is found to be the optimal.

**M.J. Islam** was born in Comilla, Bangladesh on June 1, 1975. He received the BSc. (Hon's) and MSc. in Electronics and Computer Science in 1995 and 1996 respectively from Shahjalal University of Science and Technology, Sylhet, Bangladesh. He received MAsC in Electrical and Computer Engineering from Ryerson University, Toronto, ON, Canada, in 2003. Currently he is doing PhD in Electrical and Computer Engineering at University of Windsor, Windsor, ON, Canada. The author is the IEEE student member and the member of International Association of Engineers (IAENG). His research interests include but not limited to machine learning and computer vision, image processing and document image analysis.

**Q.M.J. Wu** received the Ph.D. degree in electrical engineering from the University of Wales, Swansea, U.K., in 1990. In 1995, he joined the National Research Council of Canada where he became a Senior Research Officer and Group Leader. He is currently a Professor in the Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON, Canada. He was recently named as the Canada Research Chair (CRC) in Automotive Sensors and Sensing Systems. He has published over 100 scientific papers in areas of computer vision, neural networks, fuzzy systems, robotics, micro-sensors and actuators, and integrated micro-systems. His current research interests include 3-D image analysis, active video object tracking and extraction, vision-guided robotics, sensor analysis and fusion, wireless sensor networks, and integrated micro-systems.

**M. Ahmadi** received the B.Sc. degree in electrical engineering from Arya Mehr University, Tehran, Iran, in 1971 and the Ph.D. degree in electrical

Finally, the optimal parameters are applied to the same set of data and 94% recognition rate is achieved. This result is found to be the superior in terms of complexity of the network, training time and testing time.

## REFERENCES

- [1] Alpaydin E., An Introduction to Machine Learning, The MIT press, Cambridge, Massachusetts, London, England, 2004.
- [2] LeCun Y. et al., Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 1999, pp. 2278–2324.
- [3] Dataset [online] Available: <http://pages.cs.wisc.edu/dyer/cs540/hw/hw4/digits5-9.dat>.
- [4] Duda R. et. al. Pattern Classification. Wiley Inter-science, 2nd ed. 2000.
- [5] Hamze F., Technical report, Stanford University, March 2000.
- [6] Alginahi Y. 2004, Computer analysis of composite documents with non-uniform background. PhD Thesis, Electrical and Computer Engineering Department, University of Windsor, Windsor, ON, Canada.
- [7] Sid-Ahmed M.A., Image Processing: Theory, Algorithm and Architectures. McGraw-Hill, 1995.
- [8] Wu Q.M.J., Class Notes- Machine Learning and Computer Vision. University of Windsor, Windsor, ON, Canada, 2007.

engineering from Imperial College of London University, London, UK, in 1977.

He has been with the Department of Electrical and Computer Engineering, University of Windsor, Windsor, Ontario, Canada, since 1980, currently as University Professor and Director of Research Center for Integrated Microsystems. His research interests include digital signal processing, machine vision, pattern recognition, neural network architectures, applications, and VLSI implementation, computer arithmetic, and MEMS. He has co-authored the book Digital Filtering in 1 and 2 dimensions; Design and Applications (New York: Plenum, 1989) and has published over 400 articles in these areas. He is the Regional Editor for the Journal of Circuits, Systems and Computers and Associate Editor for the Journal of Pattern Recognition, and the International; Journal of Computers and Electrical Engineering.

**M.A. Sid-Ahmed** was born in Alexandria, Egypt, in March 1945. He received the BAsC. degree in Electrical Engineering from the University of Alexandria, Egypt, in 1968 and the Ph.D. degree in Electrical Engineering from the University of Windsor, Windsor, Ontario, Canada, in 1974. After graduation, he worked at the Alberta Government telephones and the University of Alexandria, Egypt. He joined University of Windsor in 1978 as a faculty member and is presently the department head for the ECE program. Dr. Sid-Ahmed holds four US patents, published a book on Image processing with McGraw-Hill and authored and co-authored over 100 papers. His areas of interest include image processing, robotic vision, pattern recognition and Improved Definition Television

TABLE II: OUTPUT OF THE NETWORK. H = 10 AND  $a = 0.2$

Output	Run:1	Run:2	Run:3	Run:4	Run:5	Average
Iterations	380	410	290	340	300	344
Correct output (%)	90	100	90	95	90	93
Training time (sec)	92.157	97.390	70.828	83.203	73.828	83.48
Testing time (sec)	0.0310	0.0320	0.0310	0.0310	0.0150	0.0280

TABLE III: CORRECT OUTPUT (%) VS. HIDDEN UNITS, H.  $a = 0.2$

Hidden units (h)	Run:1	Run:2	Run:3	Run:4	Run:5	Average Output (%)
5	95	100	90	100	90	95
10	90	95	90	95	95	93
<b>15</b>	<b>95</b>	<b>100</b>	<b>90</b>	<b>95</b>	<b>95</b>	<b>95</b>
20	95	100	90	100	90	92
25	85	100	90	95	90	92
50	80	95	90	100	95	92

TABLE IV: ITERATIONS VS. H TO CONVERGE TO MSE = 0.2.  $a = 0.2$

Hidden units (h)	Run:1	Run:2	Run:3	Run:4	Run:5	Average
5	200	540	1330	600	1290	792
10	480	360	290	340	250	344
<b>15</b>	<b>310</b>	<b>380</b>	<b>320</b>	<b>380</b>	<b>260</b>	<b>330</b>
20	320	400	280	440	370	362
25	320	360	390	400	330	360
50	280	350	260	330	320	308

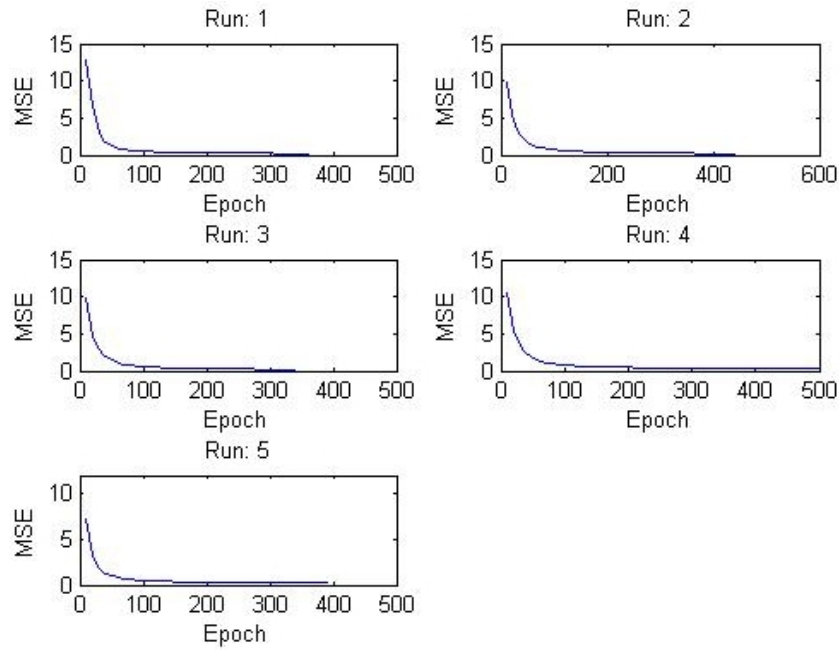


Fig. 5 : Epoch vs. MSE. Hidden units,  $h= 10$  and  $a = 0.2$

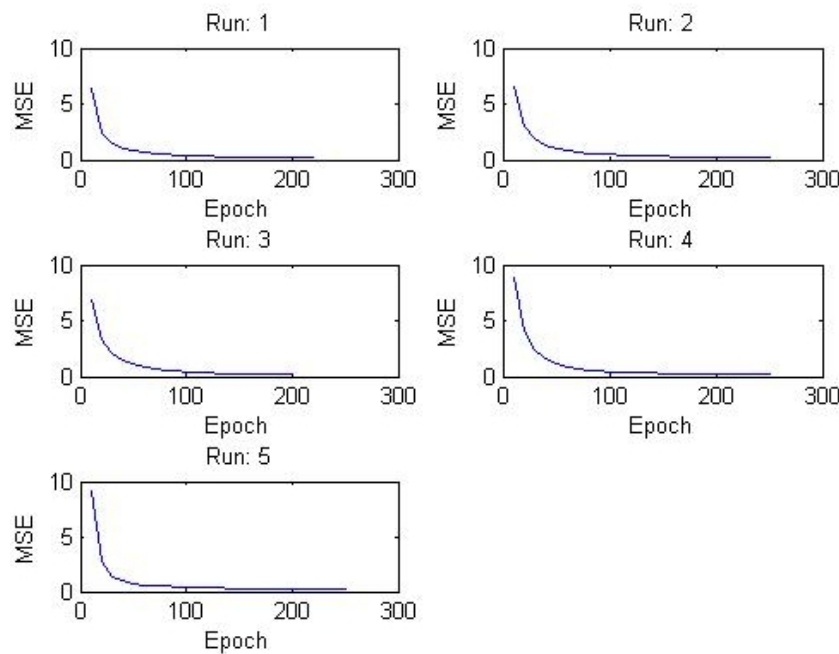


Fig. 6: Epoch vs. MSE. Optimal network for  $h = 15$  and  $a = 0.25$

TABLE V: CORRECT OUTPUT (%) vs  $a$  .  $h = 15$

Learning Rate ( $a$ )	Run:1	Run:2	Run:3	Run:4	Run:5	Average Output (%)
0.05	85	100	95	100	95	95
0.10	95	100	90	100	95	96
0.15	90	95	90	100	95	94
0.20	85	100	95	95	95	94

<b>0.25</b>	<b>95</b>	<b>100</b>	<b>90</b>	<b>100</b>	<b>95</b>	<b>96</b>
0.30	90	100	90	100	95	95

TABLE VI: ITERATIONS VS.  $a$  TO CONVERGE TO MSE = 0.2, H = 15

Learning Rate ( $a$ )	Run:1	Run:2	Run:3	Run:4	Run:5	Average Output (%)
0.05	1120	1300	1040	1270	1540	1254
0.10	600	650	790	450	660	630
0.15	380	440	470	420	460	434
0.20	290	300	360	380	320	330
<b>0.25</b>	<b>280</b>	<b>250</b>	<b>230</b>	<b>320</b>	<b>260</b>	<b>268</b>
0.30	220	200	230	220	260	226

TABLE VII: OUTPUT OF THE OPTIMAL NETWORK. H = 15 AND  $a = 0.25$

Output	Run:1	Run:2	Run:3	Run:4	Run:5	Average
Iterations	200	260	200	270	370	<b>260</b>
Correct Output (%)	85	100	90	100	95	<b>94</b>
Training Time (sec)	59.547	76.375	59.062	79.797	115.344	<b>78.025</b>
Testing Time (sec)	0.031	0.047	0.032	0.031	0.031	<b>0.0344</b>