A Systemic Study of Pattern Recognition System Using Feedback Neural Networks

QASEM ABU AL-HAIJA
Tennessee State University
Computer and Information Systems Engineering
Nashville, TN
UNITED STATES

NOOR A. JEBRIL
King Faisal University
Computing Science Department
Eastern region, Ahsa
SAUDI ARABIA

Abstract: The evolution of artificial intelligence has led to the developments of various smart applications such as the pattern recognition models. Pattern recognition techniques has as widely applied in many real life applications such character recognition, speech recognition, and bio-metric authentication as well person identification. In this paper, we report on the detailed design of pattern recognition system using Hopfield Feedback Neural Network (HFNN) with the least possible recognition error. As a case study, we have applied the proposed HFNN model to recognize the decimal digits $0-9$ where each image digit comprises a 12×10 pixels. The developed HFNN model has been efficiently used in recognizing the patterns with 20% random bit noise at maximum recognition accuracy. However, to assure the the least possible recognition error, we have trained our HFNN through the digit patterns' perdition phase of 0% noisy patterns and the system was able to correctly predict all the patterns without any bit error. Finally, we have plotted all output patterns including the desired patterns, the training patterns, the 20% noisy patterns and recognized patterns, for comparison purposes and to gain more insights about the accuracy achieved by applying the proposed HFNN.

Key–Words: Neural Network, Pattern Recognition, Feedback Neural Networks, Associative Memory, Hopfield Network.

Received: November 26, 2019. Revised:March 31, 2020. Accepted: April 25, 2020. Published: May 5, 2020.

1 Introduction

Artificial intelligence (AI) can be interpreted naturally as the work of a machine that a human could have done using his intelligence. Currently, the artificial intelligence is gaining an appreciated amount of interest as almost all major IT companies are spending millions on the development and implementation of the artificial intelligence considering it critical to their future situation [1]. Artificial intelligence has been used efficiently to develop solutions for wide range of applications especially those based on human thinking to provide a proper decision such as the pattern recognition.

Pattern recognition or pattern matching [2] is the process of identifying and/or classifying data patterns using artificial learning and neural network algorithms based on knowledge already gained or on statistical information extracted from patterns and/or their representation. Pattern recognition has a wide range of applications such image processing and analysis, speech recognition, bio-metric identification, computer vision, seismic analysis,and radar signal classification and analysis. The general concept of pattern recognition can be illustrated in Figure 1 in which the noisy data are learned and matched at by the system memory to establish an accurate model to predict the proper pattern from the actual stored data with least possible recognition error.

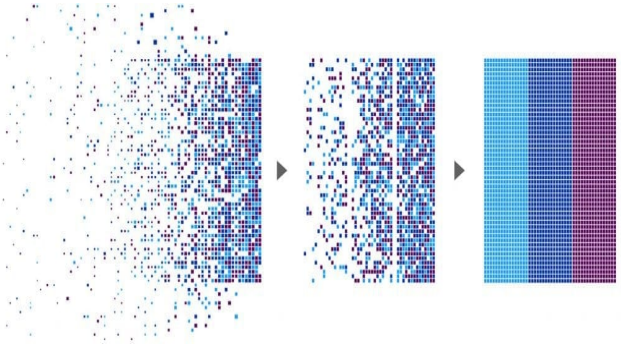

Figure 1: General idea of data pattern recognition

The pattern itself can be defined as a model of any recurring sequences of data over time that can be used to identify, classify or predict trends or features of several area of interests such as the image features to

identify objects, the frequent combinations for natural language processing, and the clusters specification of attacks/intrusions on a network. Figure 2 shows different objects that can be used as patterns in the recognition applications. Indeed, the pattern recognition became an essential integration for many overlapping areas of IT including big data analytic, bio-metric identification, security and privacy [2].

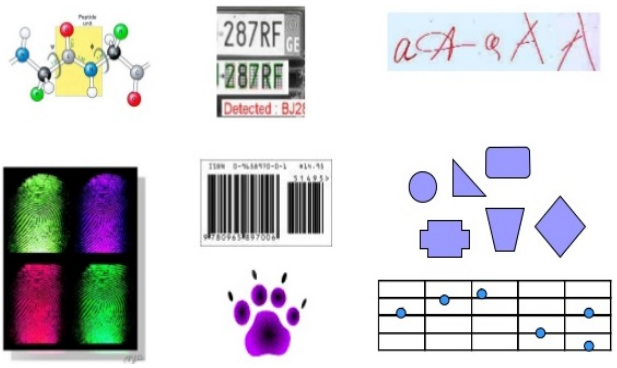

Figure 2: Examples of various patterns

In this paper, we are going to employ the Hopfield feedback neural network model to generate, predict, and recognize 10 different patterns for the decimal digits $0-9$ using binary representation imaging system. The target data-patterns will be prepared as 12×10 matrix representation images to be stored into the HNN memory. To test the efficiency of the developed HNN in the image retrieval for the stored patterns, all images will be recreated with 20% corruptions of image pixels randomly in order to be applied to HNN recognition system. Finally, HNN Modeling, coding, results and analysis are reported in this paper. Indeed, this paper aims mainly to investigate a major artificial intelligence technique that can be used to store and recognize different data patterns such as the recognition of decimal digits, namely, the Hopfiled neural networks [3]. Specifically, the main contributions of this paper can be summarized as follows:

- Understating the neural network techniques for pattern recognition such as feedback neural networks (FBNN) of Hopfield memory network and how to model their data-sets for processing purposes, and thus, data preparation for Hopfield neural network training and recognition.

- Developing a computational MATLAB codes for data pattern generation, pattern training/prediction and pattern recognition modeling. This includes a $12\times$ 10 matrix representation form and operations (such as matrices addition, multiplication, and transpose.

- Employing and training a single-layer Hopfield Neural Network (HNN) model to predict and recognize the 10 decimal digits $0-9$ using optimal recognition accuracy (i.e., least possible modeling error).

- Providing an pattern testing technique to evaluate the recognition accuracy based on a random bit noisy technique to generate a partially corrupted patterns. For example, testing the HNN using 5%, 10%, and 20% noisy patterns.

The rest of this paper is organized as follows: Section II, theoretical background, describes the overall neural network features and specifications, the feedback neural network,and the Hopfield neural network. Section III, System Implementation, provides the comprehensive graphical modeling environment to visualize the structure and the implementation of the Pattern Recognition System using Hopfield Feedback Neural Networks. Section IV, results and Discussion, presents the complete results of stored, trained, and recognized (predicted) patterns for the original and noisy patterns (additional results are provided in the appendix). Finally, Section V, Conclusions and remarks, concludes the work in this paper.

2 Theoretical Background

Artificial neural networks (ANN) systems are computing systems inspired by the biological neural networks that constitute animal brains [4]. An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal from one artificial neuron to another. An artificial neuron is an information-processing unit that is fundamental to the operation of a neural network. Figure 2 shows a notation for a single neuron model corresponding to an input vector X and output O. For each neuron, the individual elements of the input vector X = $\{x_0, x_1, ..., x_n\}$ are multiplied by the corresponding weights in the weight vector W = $\{w_0, w_1, ..., w_n\}$ and the weighted values are fed to the junction of the summation function, in which the dot product $(W.X)$ of the weight vector and the input vector is generated. The threshold b (or θ) is added to the dot-product forming the net input which is the argument of the transfer function f:

$$net = W.X = w_0.x_0 + w_1.x_1 + ...w_n.x_n + b$$

$$\Rightarrow net = \sum_{i=1}^{n} w_i.x_i + b$$

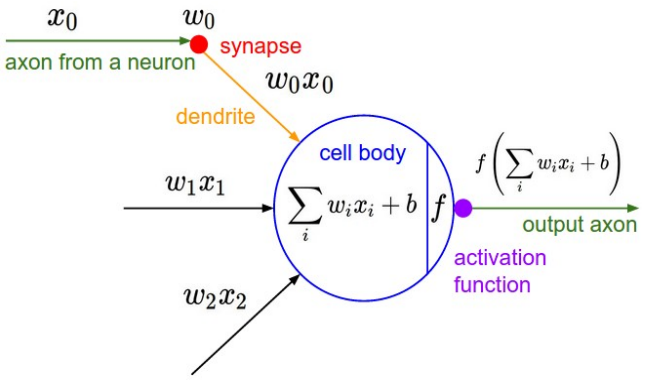

Figure 3: A Single Neuron Model

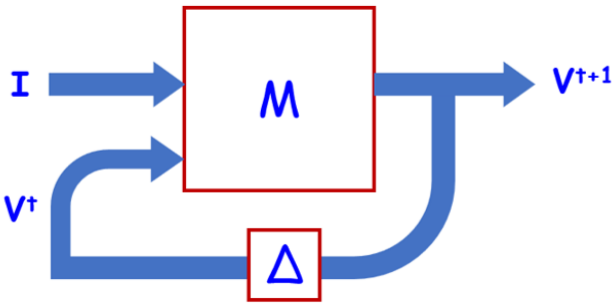

Figure 4: Block diagram of feedback auto associative network

The transfer (also called the activation) function f can be selected from a different type of functions based on the application itself [5]. Indeed, the activation function strongly influence the complexity and performance of the neural network. In the present study, Signed transfer function (also called hard limiter) was found to be appropriate for the problem investigated as the are commonly used for several NN applications. The Signed function can be expresses as follows:

$$f(net) = Sng(net) = \Gamma(net) = \begin{cases} +1, & \text{net} \geq 0 \\ -1, & \text{net} < 0 \end{cases}$$

2.1 Feedback Neural Network

Feedback Neural Network (FBNN), also called recurrent neural network (RNN) [6], is a layered neural network in which information flows from the input towards the output with feedback loops from output returned to the input and the neurons of the same layer are not connected to each other but they are connected with all the neurons of the previous and subsequent layer. There are several types of feedback neural network architecture, which has feedback connections from the output layer to the input layer or from the hidden layer to the input layer. These includes auto-associative neural networks and hetro-associative neural network.
In this paper, we are interested in the auto-associative networks in which the input vector is associated with itself in constructing the associative memory (i.e., the weighting matrix). Figure 4 depicts an illustration of a typical block diagram representation for feedback auto-associative neural network (recurrent auto-associative memory) in which the output vector V^{t+1} is fed back to the input layer V^t after processed in the M hidden layers of neurons.

2.2 Hopfield Neural Network

Hopfield neural network (HNN) [4] was invented by Dr. John J. Hopfield in 1982. It consists of a single layer which contains one or more fully connected recurrent neurons. The Hopfield network is commonly used for auto-association and optimization tasks [3]. A typical architecture diagram for the Hopfield network is shown in Figure 5.

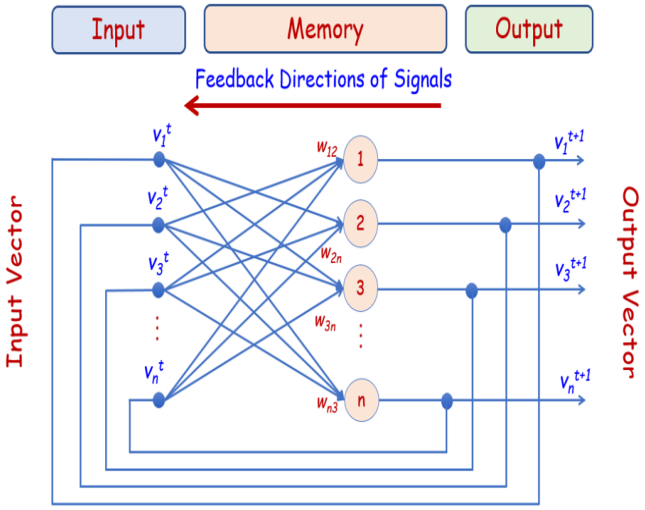

Figure 5: Typical Architecture of Hopfield Neural Network (HNN)

According to Figure 5, HNN consists of neurons with one inverting output where the output of each neuron should be the input of other neurons but not the input of self. Also, each input to the neuron has a weight strength represented by w_{ij} in which the weights are symmetrical with no self-connections i.e. $w_{ij} = w_{ji}$ and $w_{ii} = 0$.

Hopfield Training Algorithm: During training of Hopfield neural network, weights will be updated. Since Hopfield network operates using input and output patterns vector, which can be either binary (0,1) or bipolar (+1, -1) in nature. Hence, in both the cases, weight updates can be done with the following relations:

Case I (Binary input patterns): For a set of binary patterns $s(p) = s_1(p), s_2(p), ..., s_n(p)$, the weight matrix is calculated as follows:

$$\sum_{p=1}^{n}[2s_i(p)-1][2s_j(p)-1] \quad for \quad i \neq j.$$

Case II (Bipolar input patterns): For a set of Bipolar binary patterns $s(p) = s_1(p), s_2(p), ..., s_n(p)$, the weight matrix is calculated as follows:

$$\sum_{p=1}^{n}[2s_i(p)][2s_j(p)] \quad for \quad i \neq j.$$

Hopfield Testing Algorithm; To test the Hopfield neural network, we apply the following steps:

Step (1): Initialize the weights, which are obtained from training algorithm.

Step (2): Make initial activation of the network equal to the external input vector X as follows: $y_i = x_i$ for $i = 1$ to n.

Step (3): For each unit Yi, perform the following:

- Calculate the net input of the network as follows:

$$y_{inti} = x_i + \sum_j y_i.w_{ji}$$

- Apply the activation over the net input to calculate the output as follwos:

$$y_i = \begin{cases} 1, & if \quad y_{inti} > \theta_i \\ y_i, & if \quad y_{inti} = \theta_i \\ 0, & if \quad y_{inti} < \theta_i \end{cases}$$

  where θ_i is the threshold.
- Broadcast this output y_i to all other units.
- Test the network for conjunction.

Energy Function Evaluation: An energy function is defined as a function that is bonded and non-increasing function of the state of the system. Energy function E_f, also called **Lyapunov function** determines the stability of discrete Hopfield network, and is characterized as follows:

$$E_f = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j.w_{ij} - \sum_{i=1}^{n} x_i y_i + \sum_{i=1}^{n}\theta_i y_i$$

Condition - In a stable network, whenever the state of node changes, the above energy function will decrease. Suppose when node i has changed state from $y_i^{(k)}$ to $y_i^{(k+1)}$, then the Energy change ΔE_f is given by the following relation:

$$\Delta E_f = E_f\left[y_i^{(k+1)}\right] - E_f\left[y_i^{(k)}\right]$$

$$\Rightarrow \Delta E_f = -\left(\sum_{j=1}^{n} w_{ij}y_i^{(k)} + x_i - \theta_i\right)\left(y_i^{(k+1)} - y_i^{(k)}\right)$$

$$\Rightarrow \Delta E_f = -(net_i)\Delta y_i$$

Where: $\Delta y_i = y_i^{(k+1)} - y_i^{(k)}$. The change in energy depends on the fact that only one unit can update its activation at a time.

3 System Implementation

In this work, we created a Hopfield neural network to model and recognize ten digits '0', '1', ..., '9'. It is trained with standard data sets of sizes 12×10 and it is tested with the recreated noisy data sets with 20% of random corruptions for image pixels. The developed model has been implemented using MATLAB platform. However, the implementation phases of this work can be described as follows:

Phase (1): Creating Original Data Patterns for the Neural Network: This phase concerns with creating and generating the patterns for the desired output numbers 0 - 9 using image matrices of size 12 $\times$ 10. In this phase, the original data patterns was created as a unipolar matrices and then converted to bipolar using the signed function. Figure 6 shows a sample pattern matrix that has been used to create the image for digit 'ZERO', we provide both the unipolar matrix with its corresponding bipolar version after passing it through the sign function. Note that, binary bit **'1'** in the matrix fill will the grid pixel with **'white'**

color whereas the binary bit **'0'** in the matrix will fill the grid pixel with **'black'**.

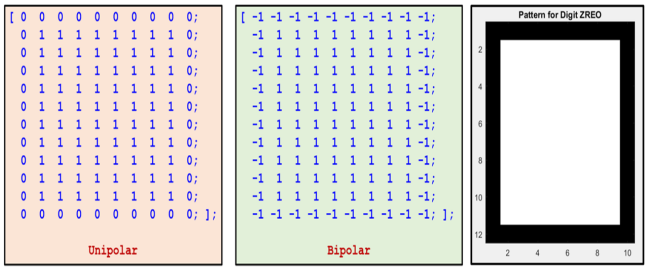

Figure 6: Creating Pattern for Digit Zero **'0'**

Phase (2): Visualizing the desired output patterns: This phase is concern with plotting the desired output numbers of bipolar matrices data which is to be used by HNN training and storage algorithm to verify the bit distribution of each matrix pattern. Figure 7 shows the plot of this phase which illustrates all the desired output digit patterns using black-white images with 12 $\times$ 10 pixels. Indeed, the black and white color scheme is not the only allowed scheme, any two different colors can be used to express the patterns such as green and yellow.

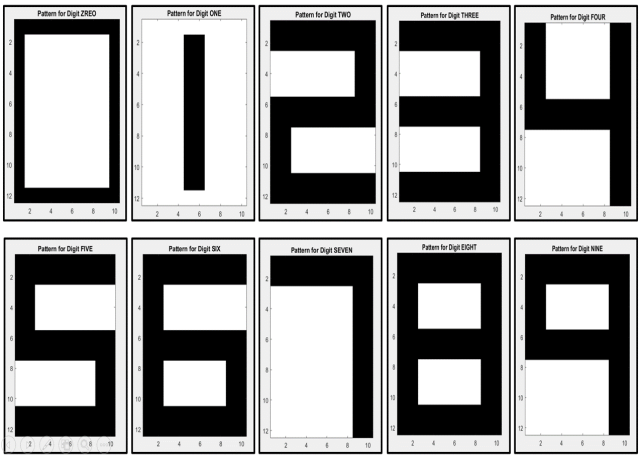

Figure 7: The desired output patterns for all numbers

Phase (3): Implementing Hopfield Neural Network (HNN): This phase is about developing the complete model and coding for the feedback Hopfield auto-associative memory network. Since patterns are stored in 12 $\times$ 10 matrices, the network structure is composed of 120 inputs to the network and thus 120 neurons are used to form the input layer as depicted in Figure 8 which shows the architecture of the proposed HFNN in which all outputs are feedback to the inputs in a fully connected network with the neurons except for the output of the same neuron is not feedback to its input.

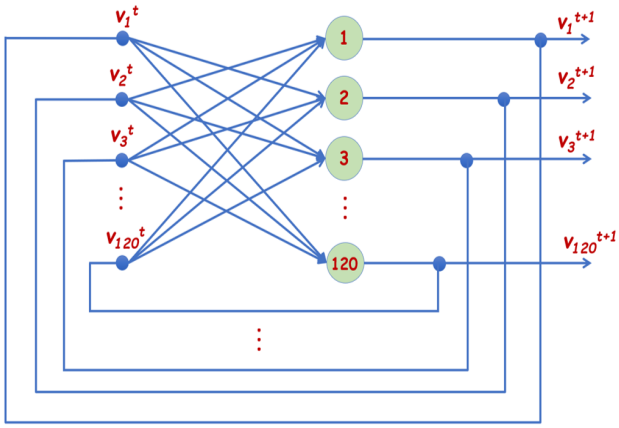

Figure 8: Schematic design of the Hopfield NN

Phase (4): Constructing the Hopfeild Weight Matrix: This phase concerns of preparing the weight matrix to store all vector patterns as follows:

- Get all training data vectors X_i, $i = 1, 2, 3, ..., L$.
- Compute the weight matrix $W = \sum X_i X_i^T$ over L vectors.
- Set $w_{ii} = 0$, $\forall\ i$, where w_{ii} is the i^{th} diagonal element of W.
- Assign the j^{th} row vector of W to the j^{th} neuron as its initial weights.

Phase (5): Training the HNN with The Ten Constructed Images: This phase is about training the developed HNN with all input patterns. To train the network to recognize the aforementioned numbers, we applied the corresponding 12 $\times$ 10 grids in the form of 1 $\times$ 120 vectors to the input of the network.This phase is important to assure the efficiency of HNN in image reterival for the 10 grid patterns with the least possible training error. This phase will be discussed and illustrated in the next section of this paper, results and discussions.

Phase (6): Testing HNN with Partially Corrupted Data Patterns: This phase is about investigating the robustness of the developed neural network, we added a 20% random noise bits to the input vectors. The resultant corrupted pattern are then used to test the ability of the proposed model in recognizing the partially corrupted patterns. However, the random bit noise of the 120 bit of each pattern has been developed by multiplying the 120 by 0.20 (20% noise) and then randomized by the $rand()$ function of the Matlab. The results of testing the patterns created at this phase will be discussed and illustrated in the next section of this paper, results and discussions.

4 Results and Discussions

The pattern recognition of numerical digits has always been a subject of interest as it can be related closely with many real time models involving pattern matching such as authentication techniques. In this paper, the recognition of decimal digits patterns (0-9) has been efficiently implemented and investigated using the feedback neural network of the Hopfield auto-associative memory algorithm. Therefore, in this section, we provide the results of the implementation phase for the proposed HNN Modeling using MATLAB simulation platform. In order to train the network to recognize the ten aforementioned numbers (0-9), we have applied the corresponding 12×10 patterns input vectors to the network to assure that the network can retrieve all the stored digit images accurately. The result of this training is illustrated in Figure 9. According to the figure, the system successfully recognized all the trained exemplars with 100% of accuracy. In this figure, we have changed the color map in the code to green (+1) and blue (0 or -1) instead of black (+1) and white (0 or -1) for better readability especially in the printed version of this paper. Indeed, in Matlab, you can use the *colormap* function to change the coloring scheme in for any figure.

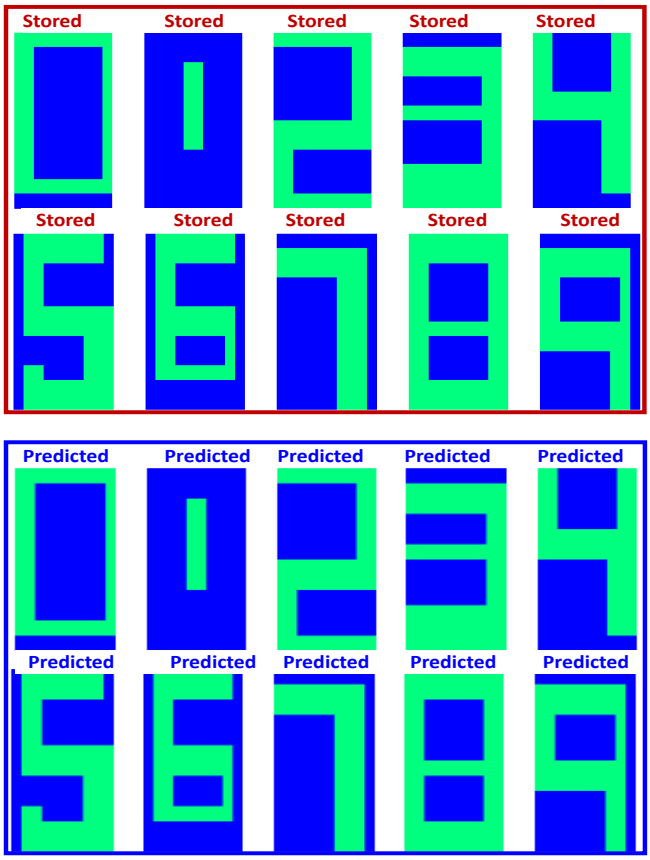

Figure 9: Stored Digits vs. Model Predicted Digits

Regarding the recognition results, in order to determine if error detection is performed correctly, we have created new partially corrupted patterns for 0-9 digits with 20% random bit noise and then we have tried to recognize/test the untrained corrupted patterns with the same trained weights. Figure 10 below shows each noisy digit and its correct recognition by our implemented HNN network. We also, provide some test results with small recognition errors in Appendix A. Moreover, we have test the network with even larger bit errors (e.g. 30%) and we get very accurate results. It can be clearly seen that, the implemented Hopfield network is robust with high convergence rate and also it is seen that it has very high success rate even if in the case of large bit errors.

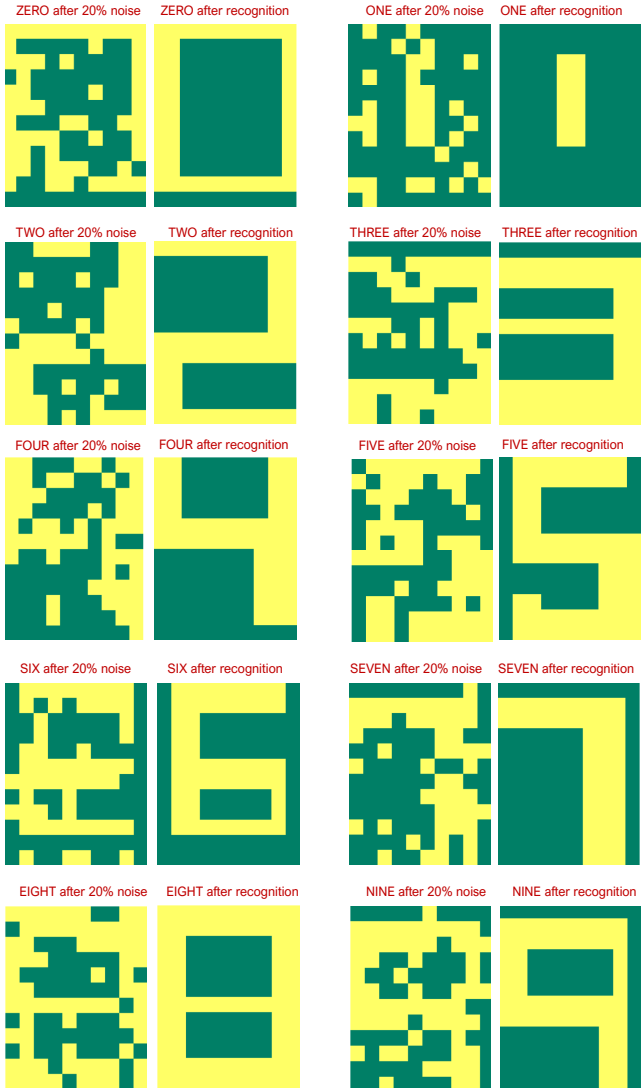

Figure 10: Recognition of *Digit Pattern* with 20% of Noise Applied on Actual Digit

From the results, it is evident that the network works fairly well even for 20% random bit noise. Also, it was noted that as the number of error bits increases from (i.e., more than 20% noise), the recognition error rate also slowly increases.

5 Conclusions

A Feed-Back Neural Network (FBNN) based Hopfield memory approach for modeling and recognizing the patterns of decimal digits with optimal accuracy levels has been developed and reported in this paper. The proposed model comprises two layers of neurons and it stored the desired patterns using 12×10 pixels images. Also, in order to test the ability of the developed HNN to recognize the partially corrupted patterns, we have developed a random bit noise technique that can generate noisy patterns randomly at each simulation run. Eventually, the simulation results showed that proposed HNN model has efficiently retrieved the desired correct patterns form a 20% noisy patterns.

6 References

1. Brunette E.S, Flemmer RC, Flemmer CL. *A review of artificial intelligence*. IEEE 4th International Conference on Autonomous Robots and Agents, pp: 385-392, 2009. Retrieved on-line from: https://ieeexplore.ieee.org/document/4804025/
2. C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006. Retrieved on-line from: http://users.isr.ist.utl.pt/ wurmd/Livros/school/Bishop
3. S. Haykin, *Neural Networks and Learning Machines*. 3^{rd} Edition, Pearson publications, ISBN-13: 978-0-13-147139-9, Retrieved on-line from: http://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf
4. A. P. Engelbrecht, *Computational Intelligence: An Introduction*. John Wiley & Sons Ltd, 2^nd edition, ISBN 978-0-470-03561-0, Apr, 2007. Retrieved on-line from: https://www.wiley.com/en-us/Computational+Intelligence
5. Karlik, B., Olgac, A.V. *Performance Analysis of Various Activation Functions in Generalized MLP Architectures of Neural Networks*. Int. J. Artif. Intell. Expert Syst. 2011, 1, 111–122. Retrieved on-line: http://www.cscjournals.org/library/
6. Williams, Ronald J.; Hinton, Geoffrey E.; Rumelhart, David E. *Learning representations by back-propagating errors*. Nature. 323 (6088): 533–536, October 1986. Retrieved on-line from: doi:10.1038/323533a0. ISSN 1476-4687.

7 Appendix: Samples of Erroneous Recognition

Here are samples of digit recognition with error, appeared through simulation runs.