INCREASING THE CAPACITY OF A B-MATRIX NEURAL NETWORKS

By

PRERANA RANI LADDHA

Bachelor of Technology in Computer Science

Jawaharlal Nehru Technological University

Hyderabad, AP, India

2009

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 2011

INCREASING THE CAPACITY OF A B-MATRIX NEURAL NETWORKS

Thesis Approved:

Dr. Subhash Kak

Thesis Adviser

Dr. Johnson P. Thomas

Dr. Michel Toulouse

Dr. Mark E. Payton

Dean of the Graduate College

ACKNOWLEDGMENTS

It gives me immense pleasure in acknowledging all those people, without whom, this thesis would have been impossible.

I owe my deepest gratitude and respect to my Thesis Advisor Dr. Subhash Kak who has always been a guiding light throughout. It was his enthusiasm, inspiration and patience which has always prompted me into further exploring this topic. It was his timely advice and guidance that led me this far. I could not have imagined having a better advisor and mentor for my Masters degree.

I would also like to thank Dr. Michel Toulouse and Dr. Johnson P Thomas for encouraging me with their invaluable comments and thought provoking questions which led me to some conclusive research.

I would like to thank my friends Thrishukanth Dasari, Anusha Gadepalli, Sindhura Reddy, Sahithi Kandula, Swapnika Ratakonda, Vikas Shetkar, Rupa Katha and Venkata Sudhir for always being there for me.

I would like to thank my sisters Meghana Laddha Jain, Vandana Saboo and my brother-in-laws, Neelesh Jain and Nishant Saboo for always believing in me and giving me constant support in everything I intended to do and for being someone I could totally depend on for moral support.

Lastly and most importantly, I would like to dedicate my thesis to my mother Susheela Laddha, for believing in me at every point of my life and teaching me to be a better human being

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

Introduction

The brain consists of interconnection of neurons. In an effort to create computing structures that are as efficient as the brain at cognitive tasks, interconnected artificial neurons form the basis of much research in cognitive science and artificial intelligence [1]-[12]. Specifically, research has been done on anomalous abilities [1], [2] and generalization and learning of memories [3]-[16].

1.1 Neuron and its behavior

A neuron is the basic building block of the nervous system. Each neuron in the network exhibits high functionality as it is specialized to transmit and receive information throughout the body. A neuron is comprised of three main parts: cell body, axon and dendrites. Figure 1 depicts the architecture of a neuron connected to other neurons and cells. The neurons, in order to communicate with each other, make use of both electrical signals as well as chemical messengers. The communication between the neurons is termed as Synaptic Transmission. A synapse occurs between the pre-synaptic neuron, the neuron which sends the information and the post-synaptic neuron, the neuron that receives the information. The synaptic process between the neurons is a chemical process, whereas the flow of information in the brain is an electrical process

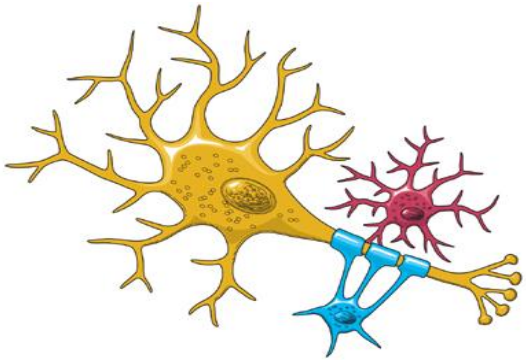
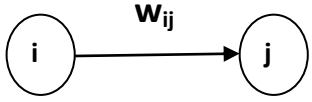


Figure 1: Architecture of a neuron [14]

A large number of these neurons connect with each other forming a neural network, which is responsible for the information processing and analysis in the brain. The connections between these neurons differentiate each individual in how we think, feel and act. In mathematical terms, if an i^{th} neuron sends a signal to the j^{th} neuron and the connection weight from neuron i to neuron j have value w_{ji}, then the total activity received by the j^{th} neuron is given as

$$A = \sum_i w_{ji} u_i$$



where u_i is the activity of the i^{th} neuron, being 1 if it is active and 0 if it is inactive. The connection weight value w_{ji} determines the magnitude of the signal reaching the j^{th} neuron from the i^{th} neuron. This elementary notion is used in the training and construction of models of both biological neural networks and artificial neural networks. The billions of neurons present in the brain are responsible for the myriad functions it performs. The networks of neurons at specific locations in the brain that receive inputs from areas such

as visual, auditory, olfactory etc have been identified. Based on this, the brain is widely classified into four lobes as shown in figure 2, where each lobe is associated with different functions. The neurons in the specific location are activated whenever the respective task is being performed by the body.

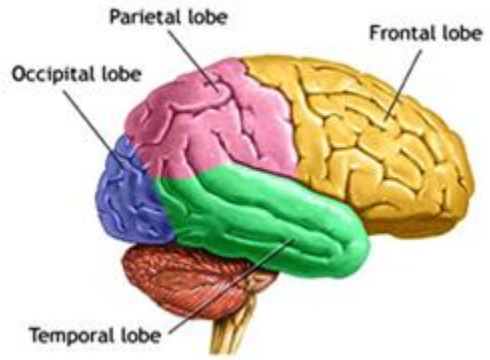


Figure 2: Brain with four lobes

When a body is involved in a series of processes, neurons from different parts of the brain have to be stimulated to participate in the completion of the process. Also, the memory associated with that task is stored in location specific neurons. Therefore, to accomplish a specific task, a series of interactions between the networks located at different parts in the brain should take place. For example, if an object has to be recognized, i.e. visual recognition memory, various factors are taken into the scenario. Visual recognition memory requires several judgments to be made such as familiarity, novelty, spatial arrangement, temporal order of the objects etc. Integration of multiple pieces of information takes place in order to recognize the object. Thus, while trying to recognize an individual, the memory not only identifies the person but also determines where and how that person is known to us.

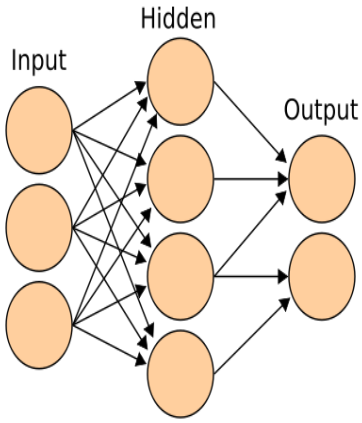
1.2 Memory Storage

It is generally accepted that to store permanent memories, the neurons in the brain reorganize themselves and the connections between them are strengthened. This perception can lead to an insight that, if the process of storing memories is a construction, then it is plausible that memories would be stored in the same neurons which originally constructed an experience and also the same neurons might be retrieved later to help remember it.

1.3 Artificial Neural Networks

Artificial neural networks are associative memories and adaptive networks. They are associative because when an input is fed into the network a corresponding output or identification is returned and they are adaptive as we can train the network using learning algorithms to produce the required outputs. Artificial neural networks to are inspired by the structure and functions of the biological neural networks. The fact that biological brain can perform highly complex tasks and yet consists of simple processing units called neurons motivated the research on artificial neural networks. Various architectures for neural networks have been proposed and analyzed by comparing and evaluating simulations on their adaptive efficiency and performance. In an artificial neural network, the capacity of neurons and the information processing capabilities of the network are limited. The input to an artificial neural network is an N-dimensional input vector and has one single output signal. The output signal is a non-linear function of the input vector and the weight vector [12]. Two kinds of networks are considered in the literature; the

feedback networks and the feedforward networks. Feed-forward neural networks allow the signals to be travelling in only one direction. In the feedback networks, on the other hand, signals can travel in any direction until the network reaches equilibrium. Based on the incoming input, the network, finds a new equilibrium point. The feedforward and the feedback networks are represented in figure 3.



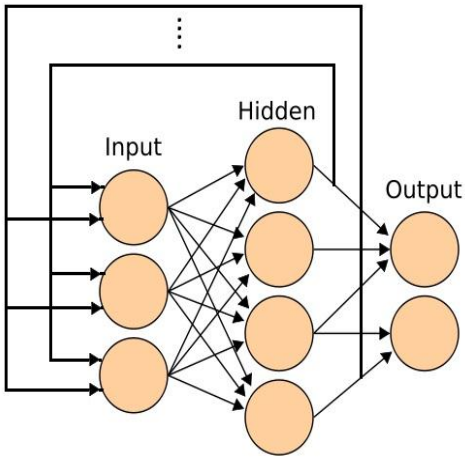


Figure 3: a) Feedforward Network b) Feedback Network

Feed-forward neural networks, also called as back-propagation neural networks are further classified into single-layer and multi-layer neural networks. As the name suggests, the single-layer feed forward neural network, has only one layer of neurons. The output signal from this layer is the output signal of the network. The multi-layer feed-forward neural network is comprised of multiple layers of neurons. Apart from the first layer the other layers receive inputs only from the previous layers of neurons. The output of the network is the output signal of the neurons in the last layer. The applications of feed-

forward neural networks are in the field of speech recognition, data reduction, sensor signal processing, balancing tasks and so on [12].

The Recurrent neural networks were explored as a solution to encode temporal information using static neural networks. These neural networks have at least one feedback loop. A feedback network is equivalent to a large static structure. These networks range from fully connected networks to partially connected networks. In fully connected networks, each node has input from all other nodes and there is no distinct input layer. In a partially connected network, few nodes behave like a part of feed-forward network while the others receive feedback from other nodes. This kind of neural network is very similar to the biological structure of neural networks [17]. The very popular recurrent neural networks include BAM, Hopfield, Boltzmann machine etc. Some of the applications of recurrent neural networks are natural water inflow forecasting, wind turbine power estimation, financial prediction, and so on.

As mentioned earlier, artificial neural networks are developed based on the structure and functions of the biological neural network. Replicating a complex and high functionality structure like brain a challenging task and therefore, one is lead to several questions about the storage of memories in the network, the retrieval process of the memories, the processing of information within the structure, the activity of nodes, the indexing of memories etc. In this thesis we mainly concentrate on the retrieval process of the memories, the learning methodology of the memory vectors and the activity of the nodes in a fully connected network.

We know that brain performs zillions of tasks and has several sections within it to perform each task. At this point it becomes difficult to analyze the point from where the information processing starts and where it ends. In our work, we make a reasonable assumption that the neural activity might start at critical points and then spread through the network until equilibrium is reached. This idea comes from the B- Matrix approach proposed by Kak [6], [17]. This approach suggests a method for retrieving the memories from a network by the spreading activity, based on the proximity of the neuron. The B-Matrix approach uses the Hebbian rule of learning and shows how memories are retrieved from a single neuron in a network. After various simulations we found that the memory retrieval rate of the B-Matrix is not as high as expected. Hence, in this thesis we extend this approach by applying a learning algorithm to the B-Matrix to increase the memory retrieval rate, by making small changes to the weight matrix of the neurons.

CHAPTER II

Literature Review

2.1 Hebbian learning

The Hebbian model was proposed by Donald Hebb in the year 1949. Donald hebb wrote: *When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased [3]*.

In this model, it is assumed that the synaptic strength between two neurons is changed based on the pre-synaptic neuron's persistent stimulation of the post-synaptic neuron. In a neural network that is trained by Hebbian learning and the activation of a neuron depends on the cumulative sum of the weighted activations that are involved in the network [15]. The learning rule of these weights depends on the strength of the simultaneous activation of the sending and receiving neurons. The Hebbian learning rule is given by

$$W_{i,j} = x_i . x_j'$$

where x is the input vector and W is the weight matrix, which is an n × n matrix and n is the number of neurons in the network, containing the weights of successively firing neurons. A memory is successfully stored in the network if

$$x = sgn(W.x)$$

where *sgn* is the signum function, which is defined as follows:

$$sgn(x) = \begin{cases} 1 & if \quad x \geq 0 \\ -1 & if \quad x < 0 \end{cases}$$

2.2 B-Matrix Approach

The Hopfield model of neural networks can be viewed as a generalization of the idea of storage in terms of eigenvectors for a matrix [4]. This is more efficient as a model of storage and not as a model of recall by index. In the Hopfield model, we can check if a particular memory vector is stored or not but we cannot recall the memory by index, as it is not localized. The B-Matrix, proposed by Kak, addresses this problem based on a method which was earlier proposed by him, where in the activity spreads locally [15]. The spreading activity from one neuron to another is based on the adjacency of neurons in the network [17], [7]. Feed-forward networks are generally associated with the spreading activity. The B-matrix approach extends this notion to the feedback networks as well [17].

In the B-Matrix approach, the neural network that is trained by the Hebbian learning is considered. In this type of network the connectivity of neurons that fire together strengthens and that of those that don't gets weakened. The interconnection matrix, *T*, is calculated as a summation of the outer product of the memories stored in the network.

These memories are represented in the form of vectors of *1's* and *-1's*. The interconnection matrix *T* is given as:

$$T = \sum_{i=0}^{n} x^{(i)}\ x^{t(i)}$$

The number of memory vectors is *n* and $x^{(i)}$ represents the memory in column vectors that are stored in the network and $x^{t(i)}$ is its transpose. The resulting matrix is the T-matrix. The diagonal elements in this matrix are forced to zero. The memory is said to be stored if

$$x^{(i)} = sgn(T\ x^{(i)})$$

Once the memories are stored in the T-matrix, the B-matrix is calculated in order to initiate the retrieval process of the memory vectors. The B-matrix is the lower triangular matrix of the T-matrix, given as:

$$T = B + B^t$$

The B-matrix approach is a generator model for memory retrieval. In this approach, the activity starts from one neuron and then spreads to the adjacent neurons to increase the fragment length. With each update the fragment enlarges by one neuron and it is fed back into the circuit. This process is continued recursively until the entire memory vector is generated.

The generator model of the B-Matrix approach is shown in figure 4.

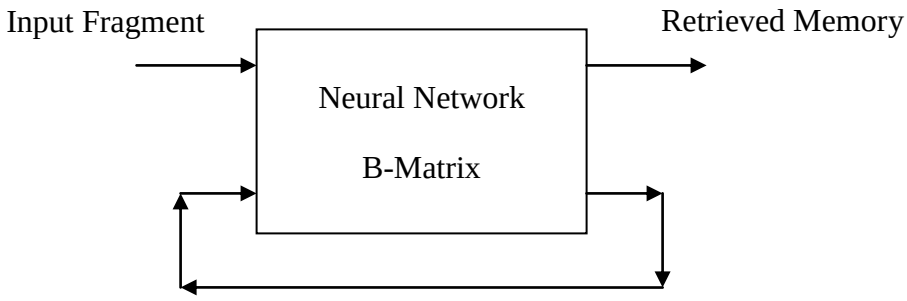


Figure 4: Generator model in B-Matrix

A fragment of the memory vector, say *1* or *-1*, is clamped onto the B-matrix which is structured according to the indexing of a specific neuron in the network. The organization of the B-matrix depends upon the proximity matrix that is determined by the adjacencies of the neurons in the neural network. The resultant fragment is fed back into the network to generate a relatively larger fragment and this process is continued until a complete memory vector is obtained. The updating process proceeds as follows:

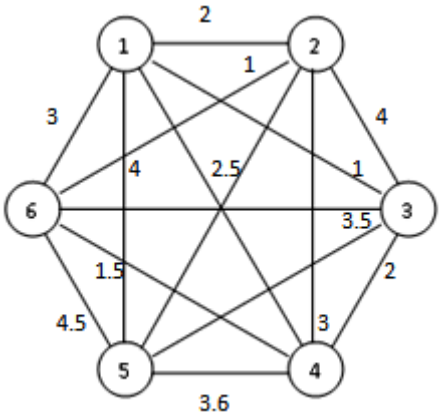
$$f^i = sgn\left(Bf^{(i-1)}\right)$$

Where f^i is the i^{th} iteration of the generator model. Notice that the i^{th} iteration of the generator model produces only the value of the i^{th} binary index of the memory vector, but does not alter *i-1* values that are already present.

The proximity matrix of the network plays a substantial role in the B-Matrix approach. It consists of the data related to distance measurements between the neurons. The neural

network is organized in a way such that the separation between the neurons takes different values. Also, the proximity of neurons does not follow the Cartesian constraints because the neural pathways may be twisted or coiled in the network [17]. Figure 5 shows the proximity matrix of a weighted network.

-



$$\begin{bmatrix} 0 & 2 & 1 & 2.5 & 1.5 & 3 \\ 2 & 0 & 4 & 3 & 2.5 & 1 \\ 1 & 4 & 0 & 2 & 3.5 & 3.5 \\ 2.5 & 3 & 2 & 0 & 3.6 & 1.5 \\ 1.5 & 2.5 & 3.5 & 3.6 & 0 & 4.5 \\ 3 & 1 & 3.5 & 1.5 & 4.5 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 3 & 4 & 6 & 2 & 5 \\ 2 & 6 & 4 & 3 & 1 & 5 \\ 3 & 1 & 2 & 6 & 4 & 5 \\ 4 & 6 & 1 & 3 & 5 & 2 \\ 5 & 1 & 3 & 4 & 6 & 2 \\ 6 & 2 & 1 & 3 & 4 & 5 \end{bmatrix}$$

Figure 5: Proximity Matrix

Let us assume without loss of generality that this network is already trained with a set of memories. The spreading activity takes different routes from different starting points depending upon their proximity. Assume that the activity starts at the third neuron and spreads from there on. The synaptic order given by the proximity matrix is [3 1 2 6 4 5], and the memory retrieval proceeds as shown in the figure.

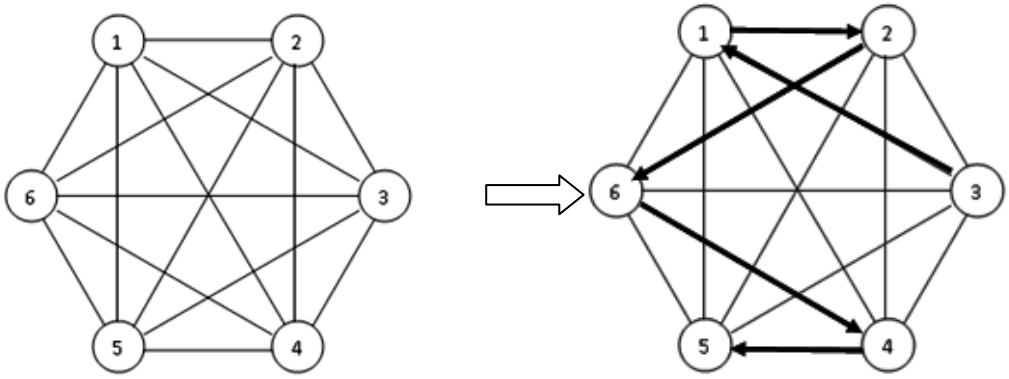


Figure 6: Graph showing the Neural Network and Activity Spread from Neuron 3.

In this approach, in order to retrieve the memories from a network, a small input fragment is clamped on to the right neuron which in turn triggers another neuron depending upon the proximity matrix. The memory retrieval process in the B-matrix approach appears to be analogous to what happens in biological systems. To illustrate this with an example, consider a person listening to a song. The next time he hears the song, he need not listen to the whole song to remember that this was the song he heard. All he needs is a small fragment of the song to help him recollect and maybe even sing the song.

As the result of several simulations we have observed that the memory retrieval rate from the B-matrix is not as high as expected. Hence, in this thesis we propose a learning algorithm to increase the memory retrieval rate in the B-Matrix approach. Because of very close similarities of the B-Matrix to the biological neural network, we chose this approach as the basis of our work and applied the Widrow-Hoff learning rule to train the network in such a way that it can retrieve more number of memories.

2.3 Widrow-Hoff learning rule

The Widrow-Hoff learning rule was proposed to increase the memory storage capacity of the Hebbian network [18]. In simple words, Widrow-Hoff learning rule also called as Delta Rule is like taking small steps towards an optimal solution. In neural networks, the elements in the weight matrix are changed iteratively by a small value in order to retrieve more number of memories.

The Widrow-Hoff model proceeds by first calculating the error associated with the retrieval of the memory from the network. Based on the error matrix obtained, the weights are so adjusted that the error for that specific memory is minimized. This procedure is repeated iteratively until all the memories of the network have been stored with no error, or with a permissible threshold.

Summarizing,

$$W_{n+1} = W_n + \Delta(\mathrm{W_n})$$

Where $\Delta(\mathrm{W_n}) = \eta(\mathrm{x_i} - \mathrm{W_n x_i})$, W is the weight matrix, x_i is the present input, and η is a small positive constant.

This way of learning represents *batch learning* as opposed to *single stimulus learning* of Hebbian Learning. It has been shown that batch learning does converge faster to the correct solution than single stimulus learning. Our use of the Widrow-Hoff learning rule to modify the elements in the B-Matrix to increase that the memory retrieval rate is somewhat similar to the approach used in [16], the Active Sites Implementation for the B-Matrix neural Networks.

2.4 Delta Rule

In [19][20], the author proposes a learning rule in which the network learns incrementally. It works in the same way as Widrow-Hoff learning rule, where, for every iteration, the weights of the network are adjusted to favor successful retrieval of the memory, with the exception that the retrieval algorithm used, is the B-Matrix approach instead of the Hebbian approach. Once a memory is retrieved by the network the process is taken further to retrieve another memory by making changes in the weights of the network, while the network is still capable of retrieving the previously retrieved memories. Active Sites are also defined, which are identified by the network during training and these sites are dependent on the input vectors given to the network This increases the retrieval rate of the network but results in high complexity because, as the size of the network increases the probability of retaining the previously retrieved memories is reduced.

In our work the inactive neurons in the network are chosen to retrieve the memories that have not been retrieved by the network. Changes are made in the proximity of these inactive neurons such that they retrieve the desired memory when triggered by an input fragment. This reduces the probability of losing the memories that have been retrieved earlier and reduces the complexity of the learning algorithm. The next section gives a detailed explanation of our proposed algorithm and also steps taken to reduce the complexity of the algorithm while maintaining high retrieval rate.

CHAPTER III

Proposed Approach

The values of the weights control the response of the output at each stage in the neural network. This forms the starting point in the development of our method. The weight values in the neural network have to be modified such that maximum numbers of memories are retrieved. The network *learns* by changing the weights between the edges of the neurons in order to get the desired output. *Learning* in a network is a repetitive incremental process by which the system produces desired outputs in response to a specified input. Biologically, the learning mechanism used in our work can be compared to the Synaptic Plasticity in the brain. Synaptic Plasticity is modifying the strength or efficacy of synaptic transmissions between the neurons at the pre-existing synapses [21] and this is also called neuroplasticity. In our work, we attempt to extend this concept to artificial neural networks in order to retrieve memories by applying the B-Matrix and Widrow-Hoff learning rule.

3.1 Algorithm

In the B-matrix approach, the memory vector can be retrieved by triggering a neuron with a small input fragment. Forming this as the basis of our work, we propose a method to increase the memory retrieval rate from. We achieve this by subjecting the network to a.

learning methodology where in small changes are made to the weights of the network resulting in increased retrieval rate.

Consider a network represented as a quadruple, $T = \{n, E, W, r\}$, where n is the number of nodes, E is the number of edges, W is an $n \times n$ weight matrix and r is the memory retrieval rate calculated by the B-Matrix approach. In order to improve the retrieval rate of the network, the following algorithm is proposed

Step 1: Identify the Inactive nodes, $\{n_1, n_2 \ldots n_i\}$ in a Network, where $i \leq n$

Step 2: Determine the memories $\{m_1, m_2 \ldots m_j\}$ to be retrieved

Step 3: Select a node n_k, where $1 \leq k \leq i$, to be triggered

Step 4: Apply the Widrow-Hoff Learning Rule to the weight matrix W

Step 5: Apply B-Matrix to the changed network

Step 6: If Retrieval rate > r, go to Step 2

Else go to Step 3

Step 7: Repeat Step 6 until Retrieval rate =100 or Threshold =t

This algorithm gives a brief conception of the learning methodology. We will discuss each of the above steps in detail and will also show graphically the improved efficiency in the retrieval rates of different networks.

3.2 Inactive Nodes

We consider a network that is trained by the Hebbian rule and the memories $\{m_1, m_2 \ldots m_l\}$ are stored in the network. Once the memories have been stored in the network, we calculate the T-Matrix and B-Matrix for the network. A small input fragment of 1 or -1 is applied to the B-matrix of each neuron in the network. The B-Matrix of each neuron is ordered by the proximity of the respective neuron. The resultant fragment is fed back into the network to generate a relatively larger fragment and this process is continued until a complete vector is obtained. The memory vectors obtained from each of the *n* neurons is $\{v_1, v_2 \ldots v_n\}$. $\forall$ v_k such that $1 \leq k \leq n$, compare it to each memory vector m_j such that $1 \leq j \leq l$. If any of the vector v_k is equal to either m_j or $- m_j$ then that memory is retrieved and the neuron that retrieves the memory is an active neuron. The neurons whose vectors are not same as any of the memories, such type of neurons are the *inactive* neurons.

The inactive neurons are chosen to retrieve the memories that have not been retrieved by the network. Changes are made in the proximity of these inactive neurons such that they retrieve the desired memory when triggered by an input fragment. In some cases we do not have any inactive neurons i.e. all the neurons retrieve a memory vector that has been stored in the network, yet all the memories are not retrieved. In this case, the neuron that has generated the most frequently retrieved memory vector is selected as the inactive neuron. This is chosen because, when there are changes made in the B-matrix there is a probability that the active neurons in the network might get inactive. By choosing this type of neuron, we reduce the chances of losing the memory vector.

3.3 Memory to be retrieved

If $\{m_1, m_2 \ldots m_l\}$ are the memories stored in the network, then on triggering the neurons with a small fragment, not all memory vectors are retrieved. The memory vectors that have not been retrieved are given as $\{m_1, m_2 \ldots m_j\}$ where $0 \leq j \leq l$. In order to select a memory that has to be retrieved at the chosen inactive node, we calculate the hamming distance between the vector that was originally retrieved at this node and $m_{k,}$ where $1 \leq j \leq l$. The memory vector with the least hamming distance is chosen and if two or more have equal values, one of the memories is chosen randomly. This reduces the number of changes that have to be made to the weight matrix and also reduces the complexity of the process.

3.4 Widrow-Hoff Learning Rule

Once the inactive node and a suitable memory vector have been chosen, we apply the Widrow- Hoff learning rule to the B-Matrix. The B-Matrix is rearranged according to the selected nodes proximity. The error vector, B-Matrix and a threshold of *0.1* are the three parameters sent to the Widrow-Hoff function. To reduce the complexity of the process only the part of error vector and B-matrix in which the changes are to be made, is sent to the Widrow-Hoff function. This is a preeminent step by which the complexity of the process reduces significantly. The memories are retrieved from the modified B-Matrix to analyze the memory retrieval rate and also the transformation of inactive nodes to active nodes.

If the retrieval rate of the modified network is higher than the original network, the process is continued by choosing the next inactive in order to retrieve more memories.

Else, the modifications made to the weighted matrix are retraced and the memory with the penultimate hamming distance is chosen to be retrieved at that inactive node. This process continues till we have tried to retrieve all the memories at each inactive node or a retrieval rate of 100 percent is achieved.

Example

Consider a neural network with eight neurons, in which all the neurons are connected with each other. Figure x depicts such a network. Consider that the network is weighted with certain values and depending upon the proximity matrix the indexing of each neuron is as given in table x.

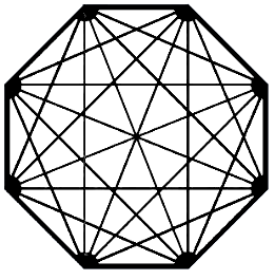


Figure 7: An 8 node connected network

Neuron	Index
Neuron1	[1 2 3 4 5 6 7 8]
Neuron2	[2 8 1 3 5 7 4 6]
Neuron3	[3 5 8 6 4 7 1 2]
Neuron4	[4 3 5 6 7 8 1 2]
Neuron5	[5 1 2 7 5 4 8 3]
Neuron6	[6 1 2 7 5 4 8 3]
Neuron7	[7 3 2 1 8 6 5 4]
Neuron8	[8 7 5 2 3 1 4 6]

Table 1: Indexes for each neuron in the network based on the proximity matrix

The five memory vectors that are stored in the network are as follows.

Memory1 = [-1 1 1 1 1 -1 1 -1]

Memory2 = [-1 1 1 -1 1 1 1 1]

Memory3 = [1 1 -1 -1 1 -1 -1 -1]

Memory4 = [1 1 1 -1 -1 1 -1 -1]

Memory5 = [1 -1 1 1 1 -1 1 1]

The matrix given below is the B-Matrix obtained from the memory vectors.

$$\begin{bmatrix} 0 & & & & & & & \\ -1 & 0 & & & & & & \\ -1 & 1 & 0 & & & & & \\ -1 & -3 & 1 & 0 & & & & \\ -1 & 1 & 1 & 1 & 0 & & & \\ -1 & 1 & 1 & -3 & -3 & 0 & & \\ -3 & -1 & 3 & 3 & 3 & -1 & 0 & \\ -1 & -3 & 1 & 1 & 1 & 1 & 3 & 0 \end{bmatrix}$$

A fragment of memory, 1 or -1 is clamped onto the B-matrix in order to retrieve the memories that are stored in the network. Table 2 shows the memories that were retrieved from different neurons in the network. From the table we observe that retrieval rate is 40%. The inactive neurons are chosen in order to retrieve the remaining 60% of the memories. The weight matrix of the network is modified using the Widrow-Hoff learning function and the necessary changes are made to the matrix. The B-matrix after the modifications is shown as follows:

Neuron	1	-1
Neuron 1	Memory 2	Memory 2
Neuron 2	-	Memory 1
Neuron 3	-	-
Neuron 4	-	-
Neuron 5	-	-
Neuron 6	Memory 2	Memory 2
Neuron 7	-	-
Neuron 8	-	-

Table 2(a): Memories retrieved from the B-matrix approach

Neuron	1	-1
Neuron 1	-	-
Neuron 2	Memory 1	-
Neuron 3	Memory 4	Memory 4
Neuron 4	-	-
Neuron 5	Memory5	Memory 5
Neuron 6	Memory 2	Memory 2
Neuron 7	-	Memory 3
Neuron8	-	-

Table 2(b) Memories retrieved after applying the learning algorithm

The final modified B-Matrix generated by the Widrow-Hoff function is as follows:

$$\begin{bmatrix} 0 & & & & & & & \\ -1.4 & 0 & & & & & & \\ -1 & 1 & 0 & & & & & \\ -1 & -3 & 1 & 0 & & & & \\ 0.2 & 1.2 & -0.2 & 1 & 0 & & & \\ -1 & 1 & 1 & -3 & -3 & 0 & & \\ -3 & -1 & 3 & 3 & 3 & -1 & 0 & \\ -1 & -3 & 0.6 & 1 & 0.8 & 1.0 & 3.0 & 0 \end{bmatrix}$$

This weighted matrix is again clamped with the memory fragments, 1 and -1 and the memory vectors obtained are listed in table 2(b). The memory retrieval rate increases from *40%* to *100%*. Also, the number of active neurons increases from *37.5* % to *62.5* %. After the various simulations, the threshold value of *0.1* proved to be efficient for the Widrow-Hoff constant, as the changes made to the matrix are very minute.

Proximity of the matrix

As mentioned earlier, the proximity matrix in the B-matrix approach is determined by the spreading activity of the neurons, which is based on the adjacency of neurons in the network. Here we carried out a few experiments by changing the manner in which the synaptic order of the neuron is generated. In this approach we select a specific node and generate the synaptic order by calculating the ascending order of the proximity of that neuron with every other neuron in the network. This pattern is depicted in the figure 8 below

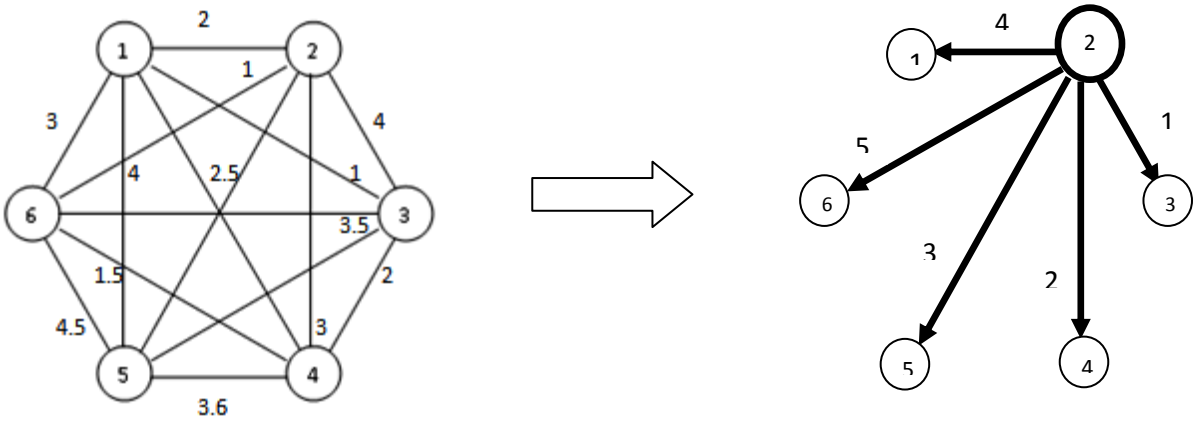


Figure 8: The Synaptic order of node 2 in a weighted network

In the above figure we observe that node 2 has a different spreading activity when compared to the one used in B-Matrix approach. In this method node 2 initially triggers node 3 and then it moves to node 4. This order depends on the weights of the edges between the nodes. Node 2 continues by triggering node 5, node 2 and finally node 6, which has the least weighted edge. Therefore, the synaptic order of node 2 is [2 3 4 5 1

6]. We performed experiments of networks of different sizes using the above mentioned procedure to calculate the proximity of the network. It was observed that the results of obtained after applying the learning algorithm were not very different compared to the results that were obtained from the regular B-Matrix proximity approach. Hence we can make a reasonable assumption that the proximity of matrix does not affect the memory retrieval rate to a large extent.

Few steps have been taken to reduce the complexity and increase the efficiency of the algorithm:

1. Error Matrix:

   In order to reduce the complexity of the algorithm, only a part of the weight matrix is called the error matrix is sent to the Widrow-Hoff function. The difference between the memory vector originally retrieved at the node and the memory vector to be retrieved at that node is taken as the error matrix. The column of the weighted matrix corresponding to the error matrix is sent as a parameter to the Widrow-Hoff function. This reduces the necessity to parse through the entire matrix each time a change has to occur.

2. Hamming Distance:

   Each time a memory is selected to be retrieved at an inactive node, the hamming distance between the memory vector originally retrieved at the node and the memory vectors is calculated. This reduces the number of changes that have to be made to the weighted matrix in order to retrieve the memory vector.

3. Loss of Memory:

Retrieving the memories at inactive nodes and the nodes that retrieve the most highly retrieved memories are chosen to retrieve the memories. This step reduces the probability of loss of memories that have already been retrieved from the network due to changes made in the weighted matrix.

CHAPTER IV

Experimental Design

PROBLEM STATEMENT:

How does the memory retrieval rate in a neural network change, when a learning algorithm is applied to it?

WORKING HYPOTHESIS:

If small changes are made to the weights of the network, then the retrieval rate of the network increases.

EXPERIMENTATION UNITS:

Network size of 8, 12, 16, 20, 28, 32, 64 nodes

CONTROL:

The simulation is based on a comparison between the memory retrieval rate in the B-matrix approach and the learning algorithm.

INDEPENDENT VARIABLES:

- Number of memories stored in the network
- Number of Nodes in the network
- Proximity Matrix

REPLICATION:

The experimental units have been simulated several times to improve the efficiency of the results. An average value of all the trials is taken to plot the graph.

LEVELS AND REPEATED TRIALS:

LEVELS (SIZE OF THE NEWTORK, N) (FOR 1: N MEMORIES)	8	12	16	20	28	32	64
REPEATED TRIALS	50	50	50	50	50	30	30

DEPENDENT VARIABLES:

- Memory retrieval rate, measured as a percentage of memories retrieved from the network

- Number of Active nodes, measure as the number of nodes retrieving a memory that has been stored in the network.

CONSTANTS:

- Learning rate in Widrow-Hoff learning algorithm is taken as 0.1

STEPS IN CONDUCTING THE SIMULATION:

1. Calculate the memory retrieval rate, r of the network using the B- Matrix approach
2. If the retrieval rate is less than 100, go to Step 2 else go to Step 10
3. Identify the Inactive nodes, $\{n_1, n_2 \ldots n_i\}$ in a Network, where $i \leq n$
4. Determine the memories $\{m_1, m_2 \ldots m_j\}$ to be retrieved
5. Select a node n_k, where $1 \leq k \leq i$, to be triggered
6. Apply the Widrow-Hoff Learning Rule to the weight matrix
7. Apply B-Matrix to the changed network
8. If Retrieval rate > r, select the next memory that has to be retrieved, else select the next inactive node to retrieve the memory
9. Repeat this procedure until maximum number of memories have been retrieved
10. Calculate the memory retrieval rate and the increase in the active neurons in the network

The simulation was done in Matlab. The program takes into consideration the proximity of each neuron and calculates the memory retrieval using the simple B-Matrix approach. It determines all the inactive nodes and applies the learning algorithm to the network in order to retrieve the maximum number of memories. It also calculates the change in the number of active and inactive neurons on applying the algorithm. The following table shows the results of the learning algorithm on networks of different sizes.

Number of nodes	% of Memories retrieved in B Matrix	% of Memories retrieved in Learning Algorithm	% of Active Nodes in B Matrix	% of Active Nodes in Learning Algorithm
8	71.31	92.27	12.38	13.34
12	55.48	86	11.72	15.37
16	42.58	78.52	10.22	16.63
20	32.46	70.10	8.07	16.06
28	25.84	63.79	5.01	9.41
32	26.65	59.64	6.3	11.61
64	14.41	44.27	6.45	13.55

Table 3: The average number of memories retrieved and Active nodes in a network in B- Matrix approach and the learning algorithm.

The following graphs show the results that have been obtained on different sized networks. The X- axis shows the number of neurons in the network and the Y-axis is the retrieval rate.

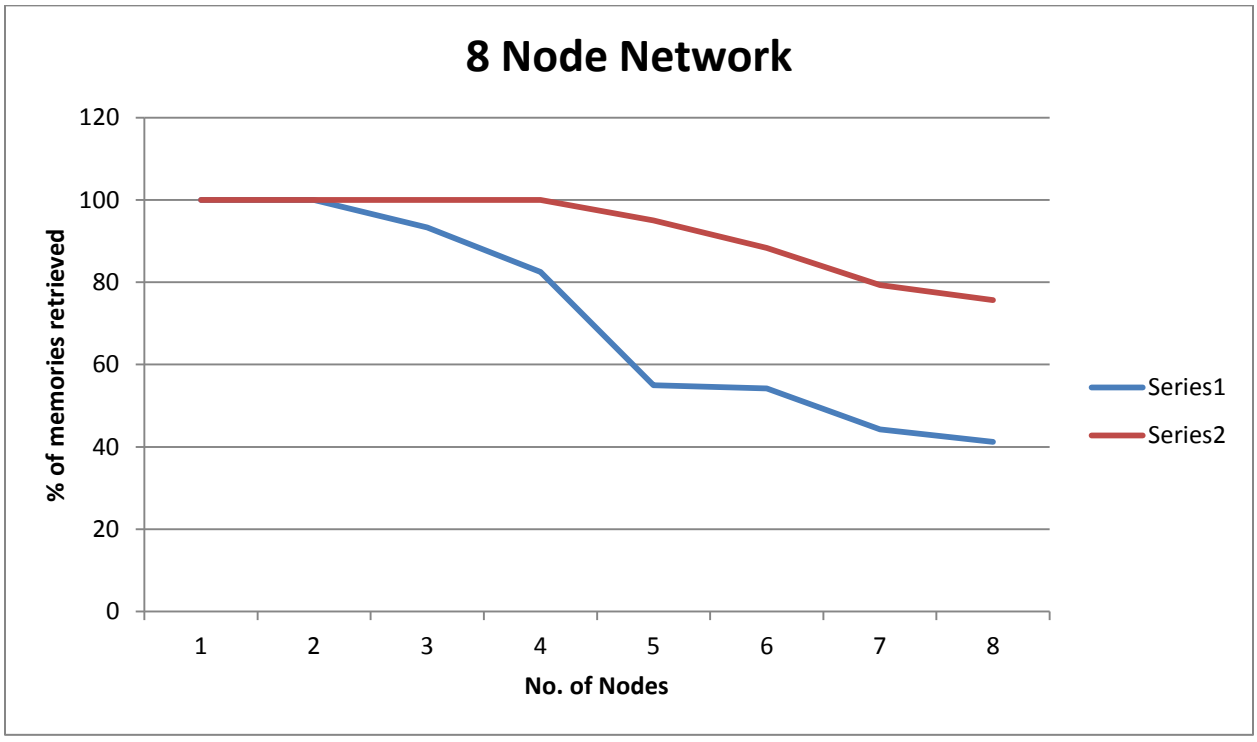


Figure 9: Graph depicting the retrieval rate in B-Matrix and learning algorithm in an 8 Node Network.

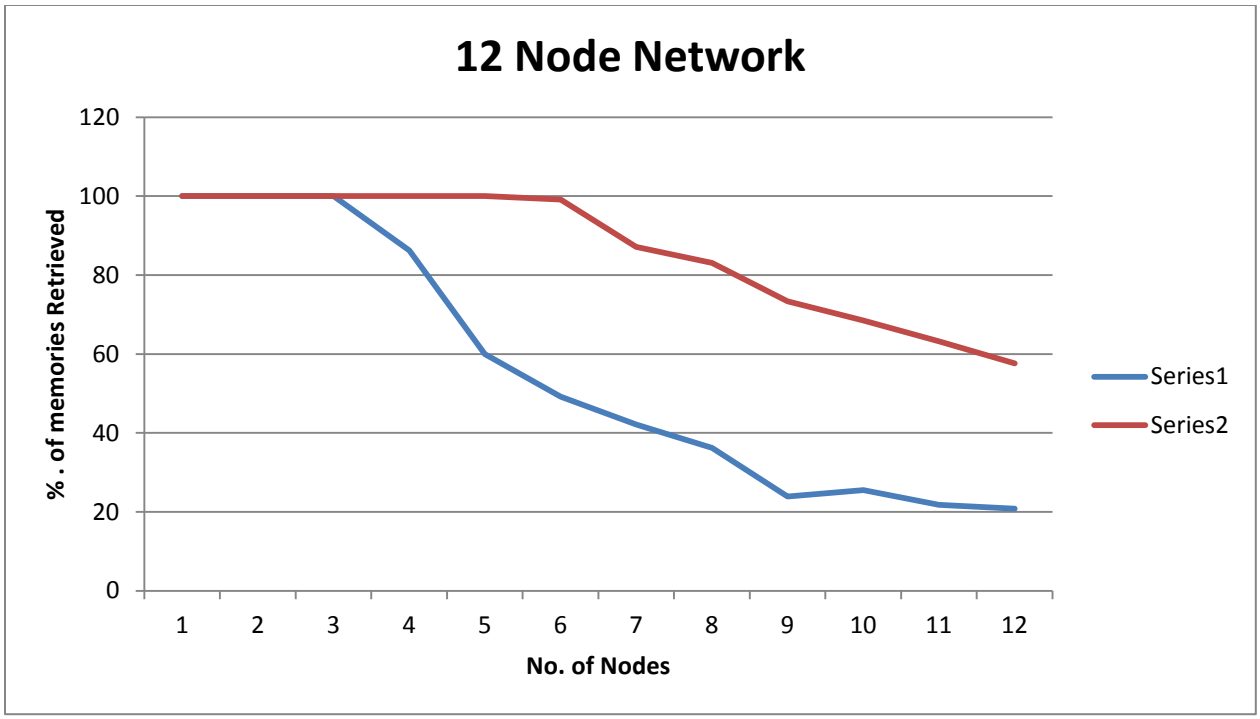


Figure 10: Graph depicting the retrieval rate in B-Matrix and learning algorithm in a 12 Node Network.

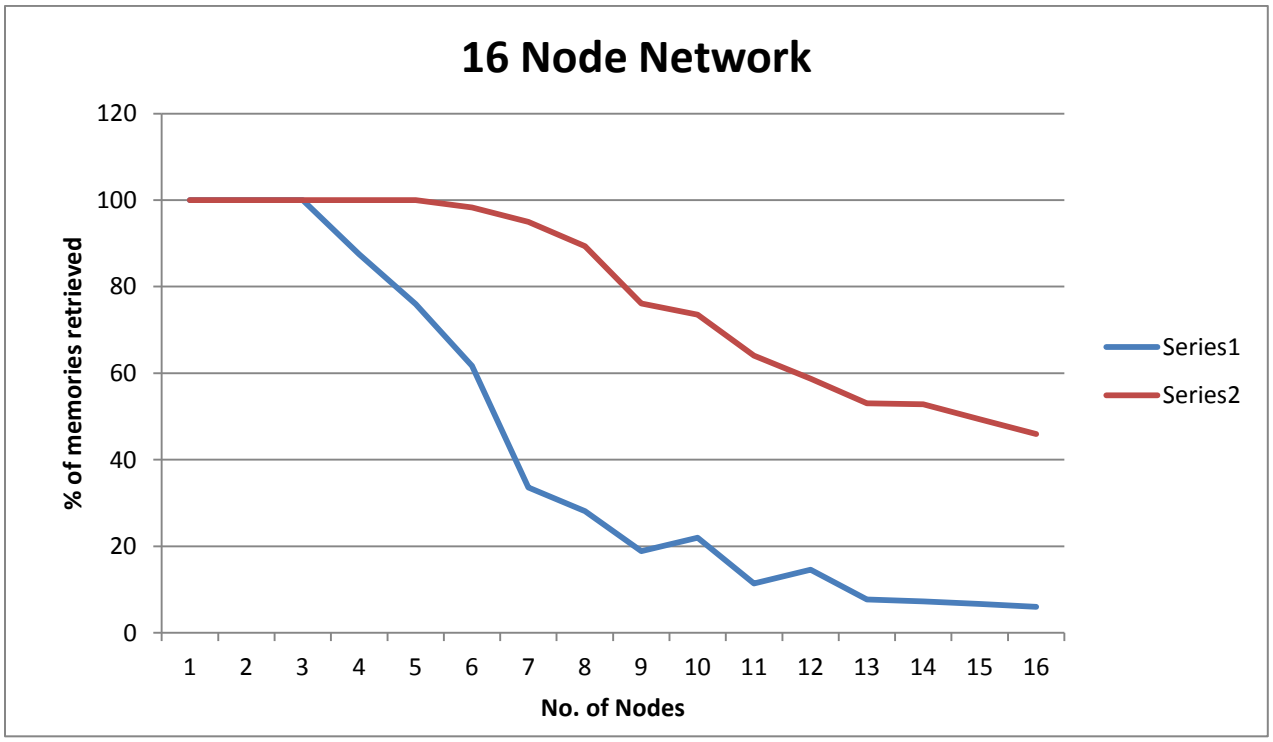


Figure 11: Graph depicting the retrieval rate in B-Matrix and learning algorithm in a 16 Node Network.

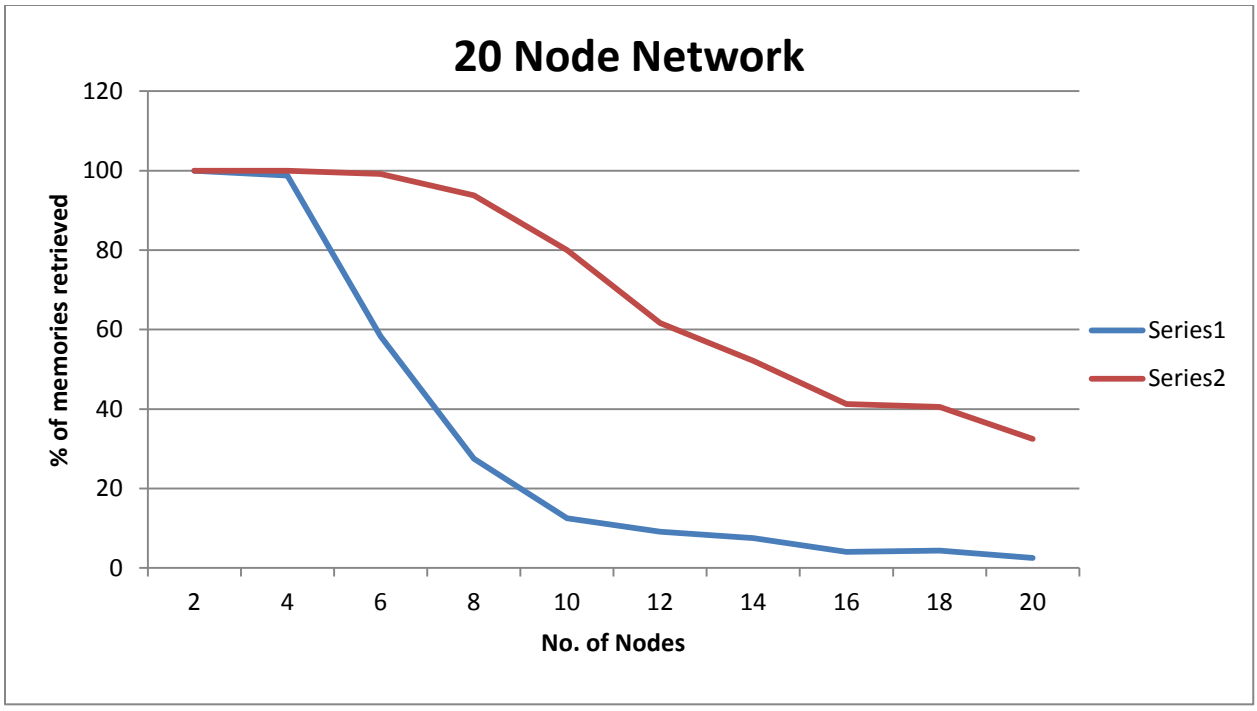


Figure 12: Graph depicting the retrieval rate in B-Matrix and learning algorithm in a 20 Node Network.

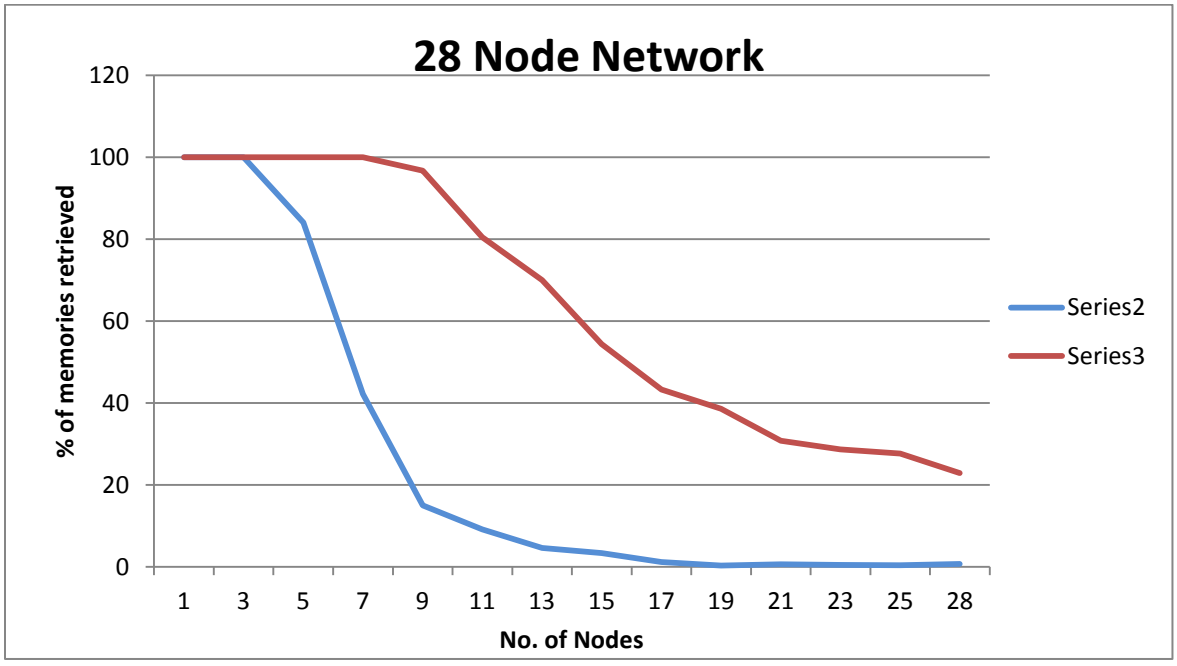


Figure 13: Graph depicting the retrieval rate in B-Matrix and learning algorithm in a 28 Node Network.

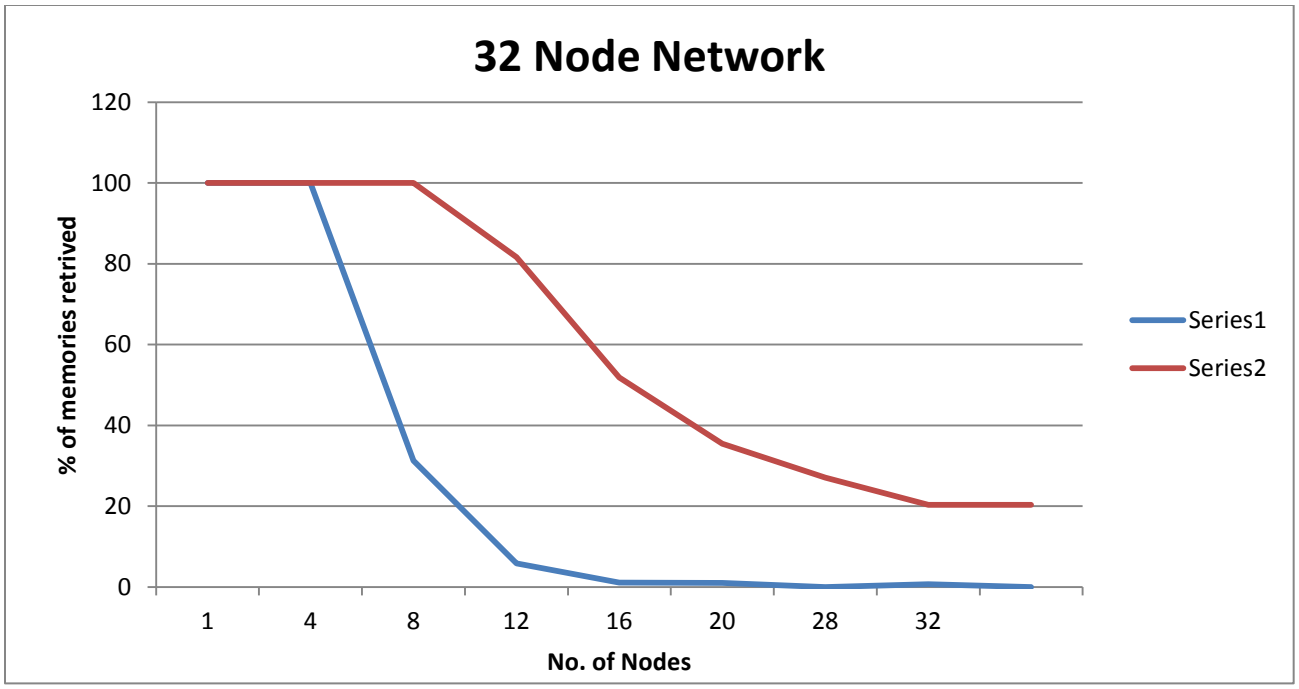


Figure 14: Graph depicting the retrieval rate in B-Matrix and learning algorithm in a 32 Node Network.

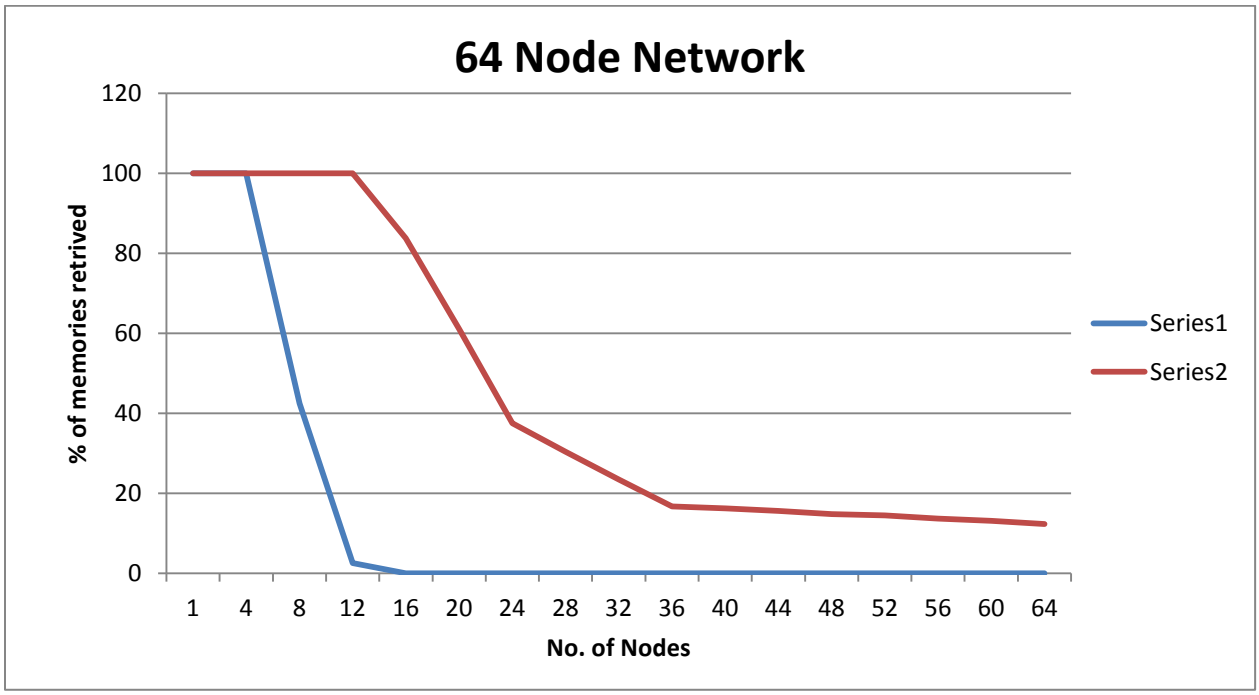


Figure 15: Graph depicting the retrieval rate in B-Matrix and learning algorithm in a 64 Node Network

ACTIVE NODES:

The following figure shows the number of active and inactive neurons in a network of 16 nodes which has been trained with 4 memories. Each color represents the each memory vector and the gray color indicates an inactive neuron.

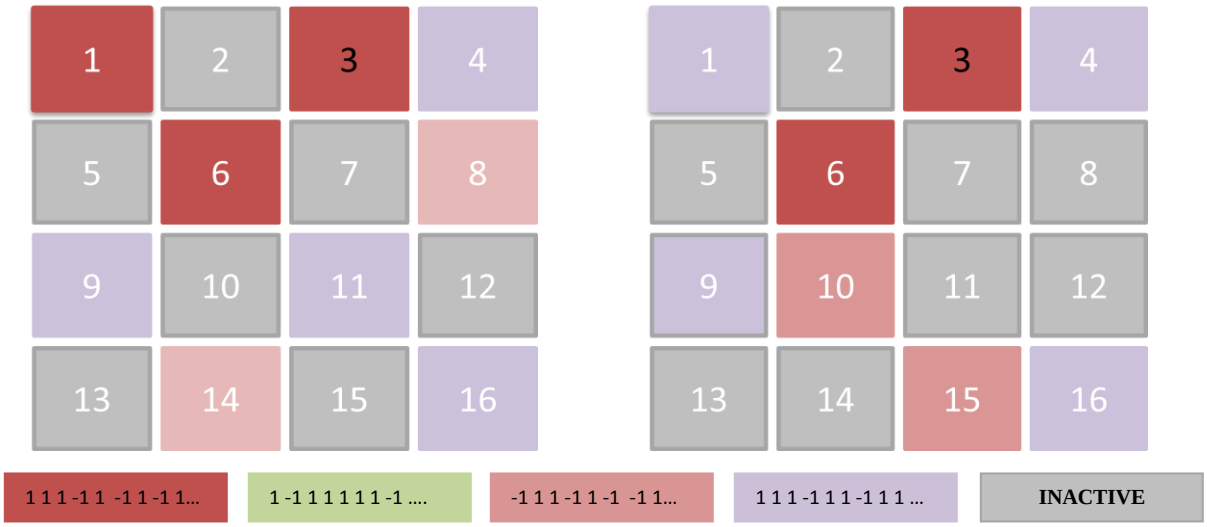


Figure 16(a), 16(b): Active and Inactive neurons when triggered with 1 and -1 respectively, before the learning algorithm

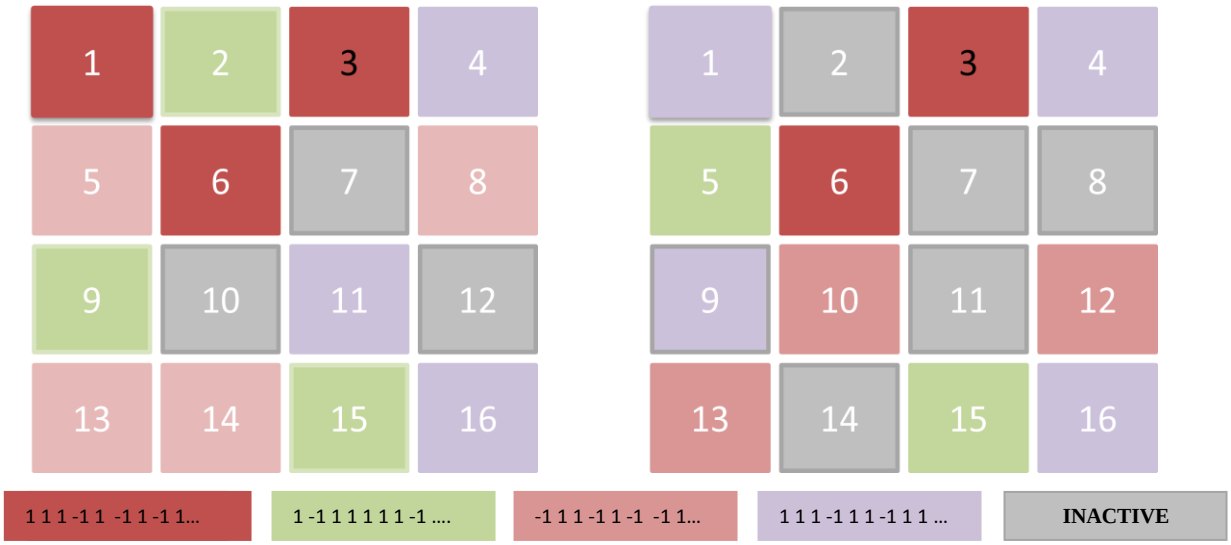


Figure 17(a), 17(b): Active and Inactive neurons when triggered with 1 and -1 respectively, before the learning algorithm

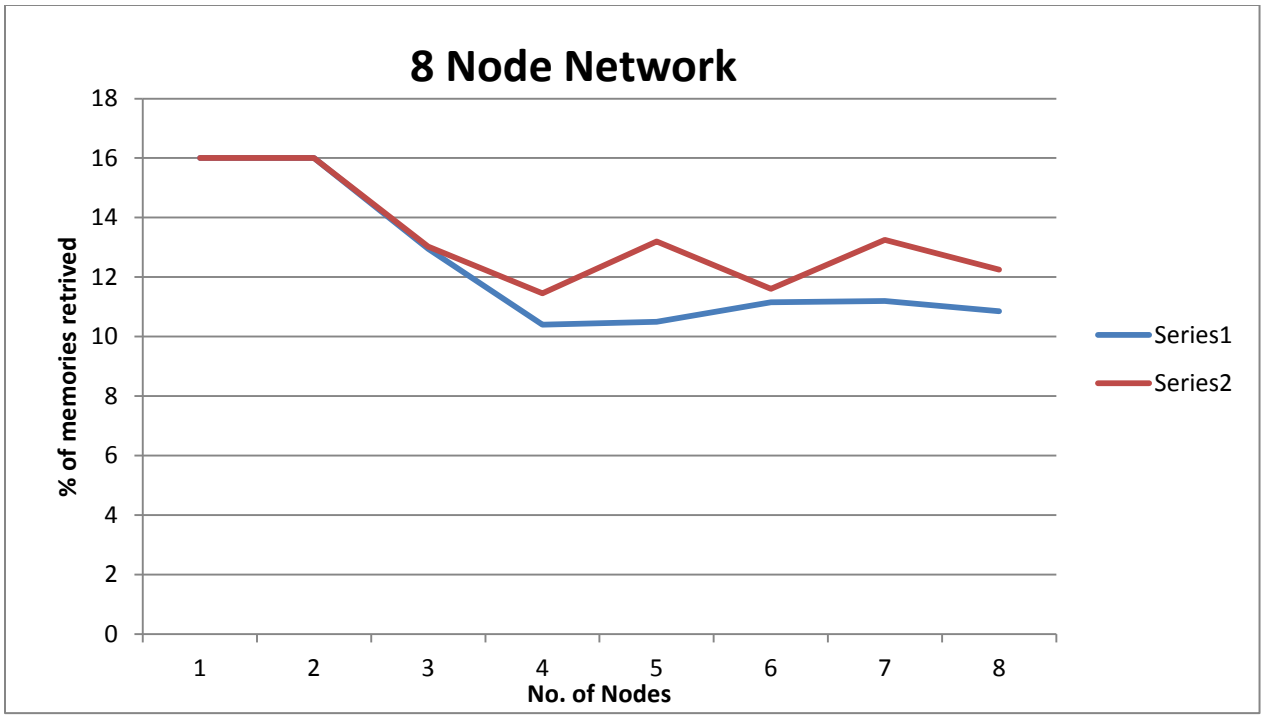


Figure 18: Graph depicting the Active nodes in B-Matrix and learning algorithm in an 8 Node Network.

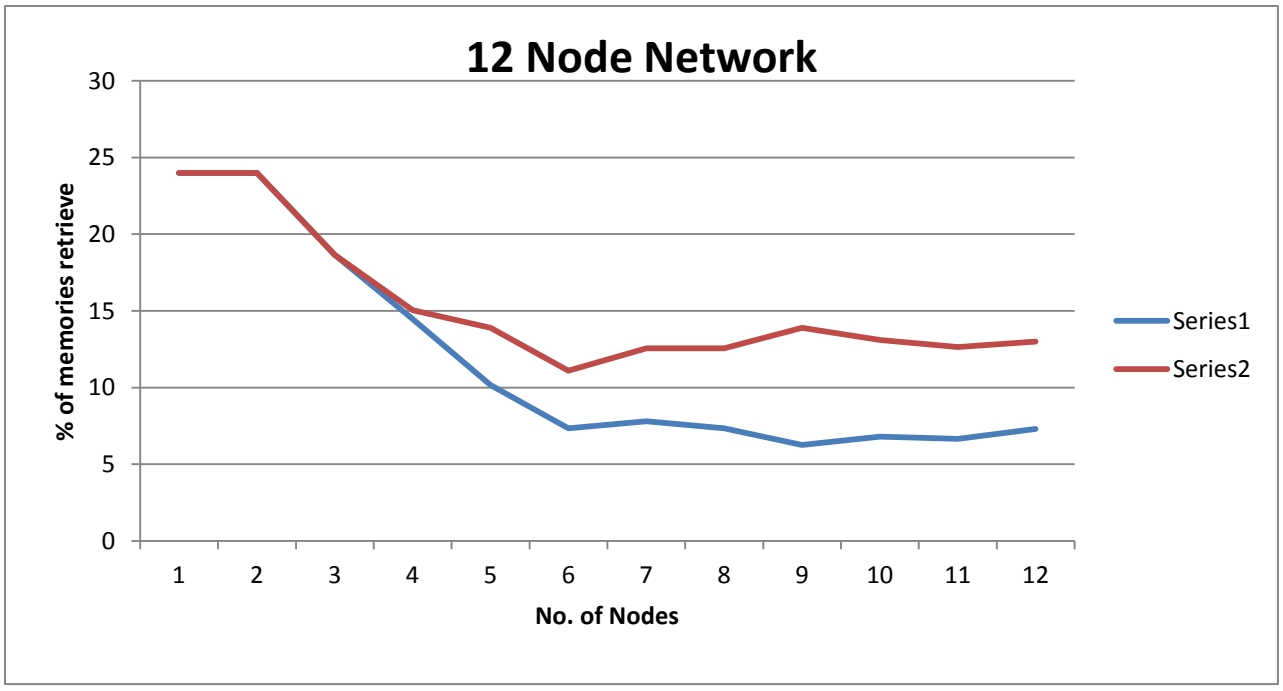


Figure 19: Graph depicting the Active nodes in B-Matrix and learning algorithm in a 12 Node Network.

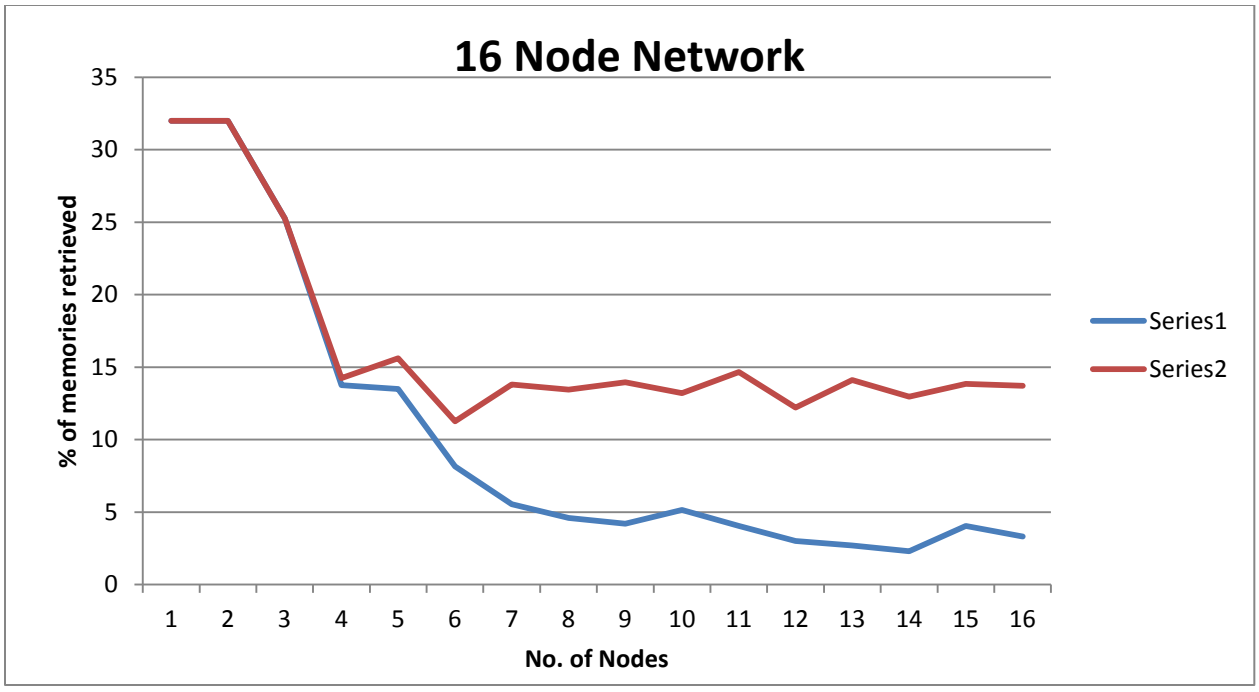


Figure 20: Graph depicting the Active nodes in B-Matrix and learning algorithm in a 16 Node Network.

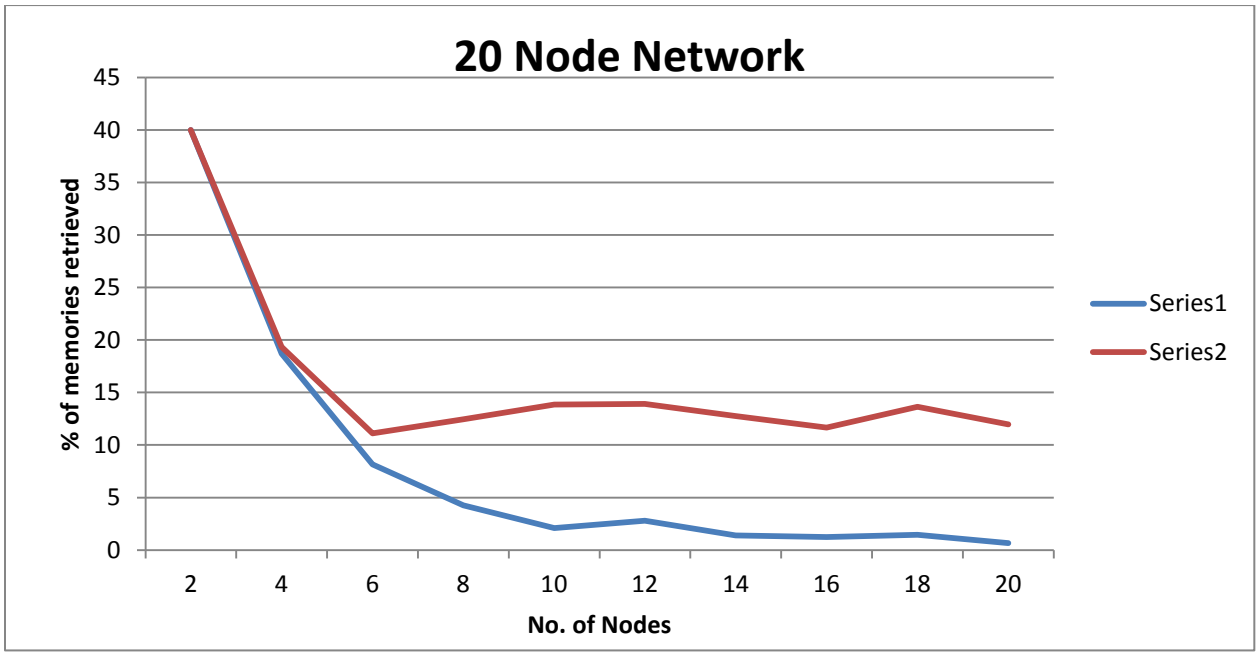


Figure 21: Graph depicting the Active nodes in B-Matrix and learning algorithm in a 20 Node Network.

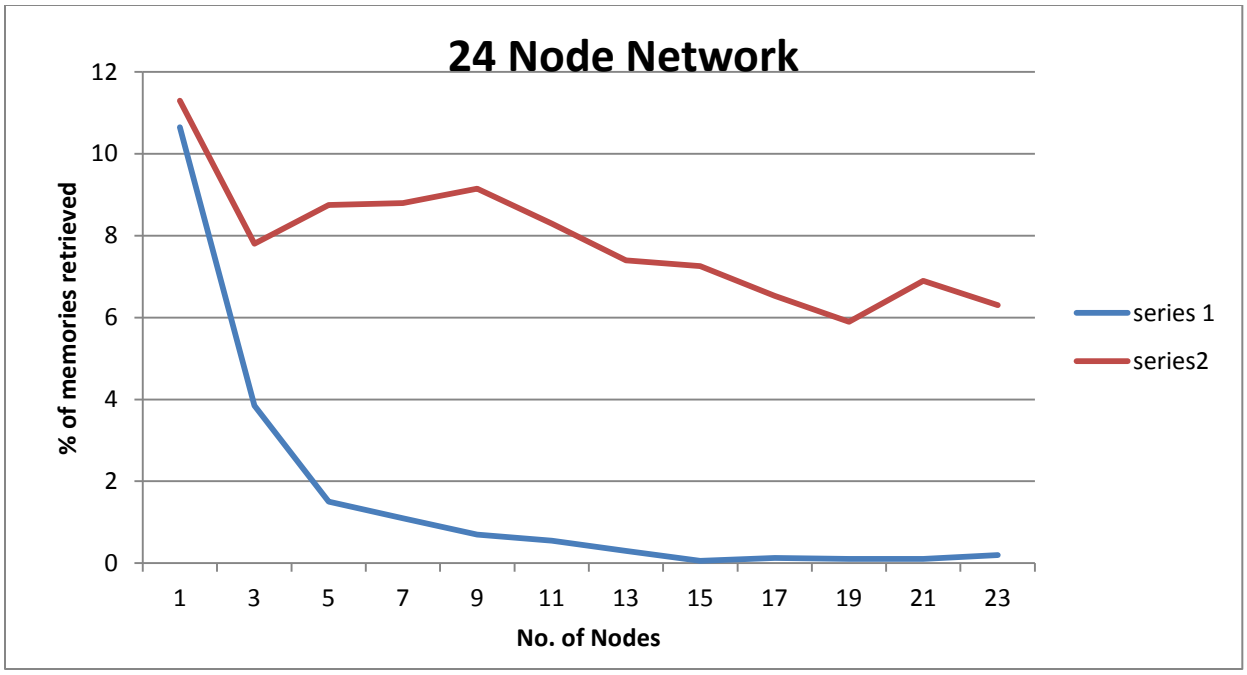


Figure 22: Graph depicting the Active nodes in B-Matrix and learning algorithm in a 24 Node Network.

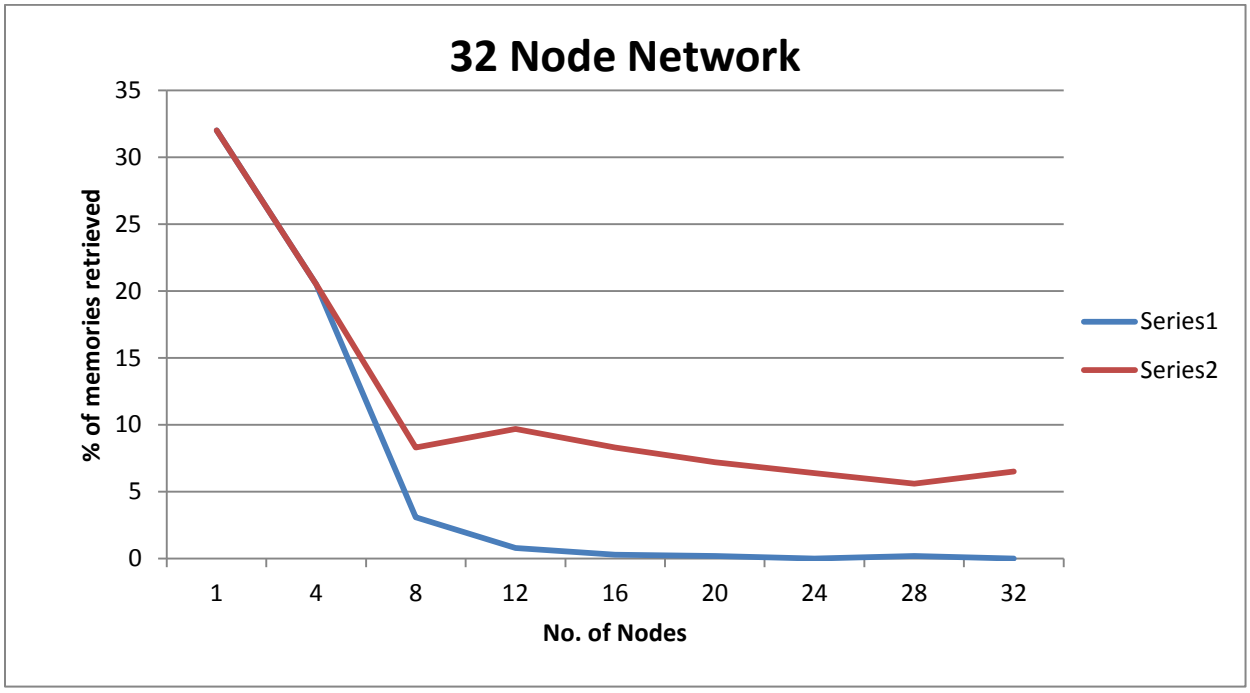


Figure 23: Graph depicting the Active nodes in B-Matrix and learning algorithm in a 32 Node Network.

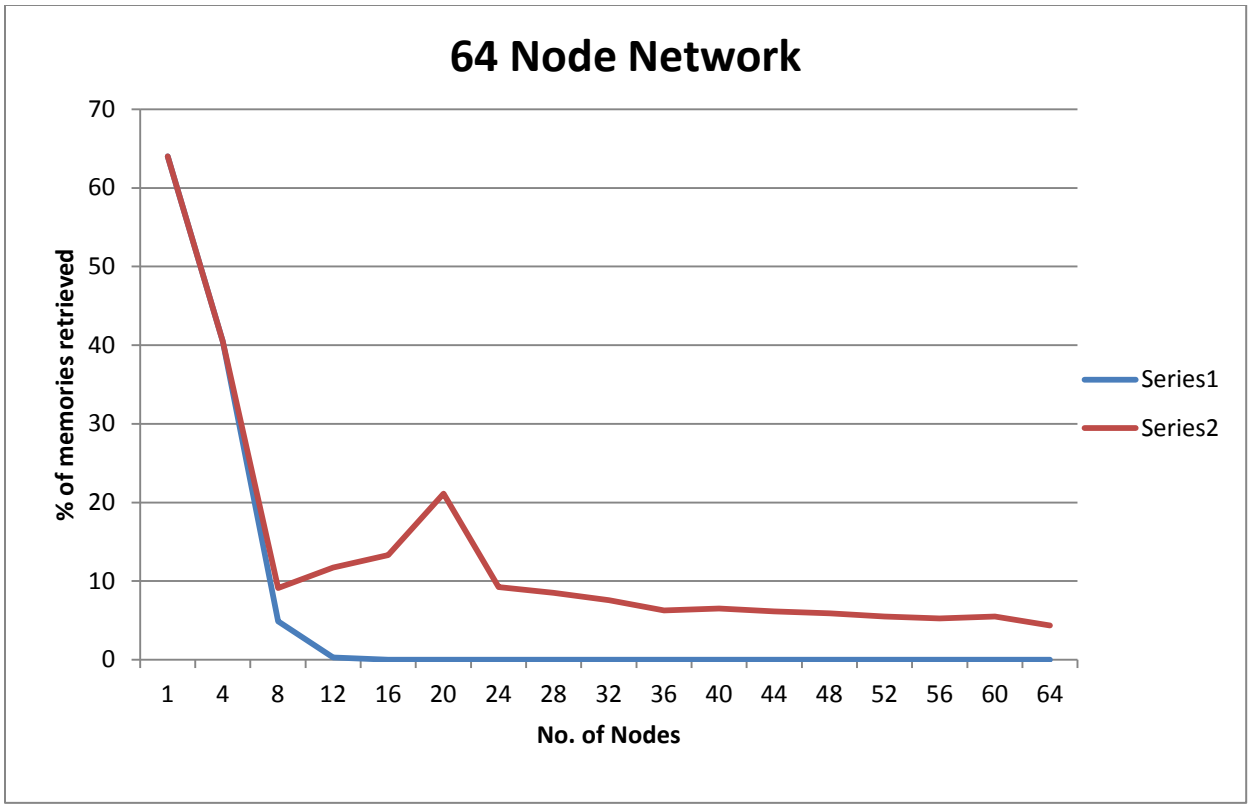


Figure 24: Graph depicting the Active nodes in B-Matrix and learning algorithm in a 64 Node Network

The following graph depicts the comparison between the average increase in the memory retrieval rate in the B-matrix approach and the learning algorithm:

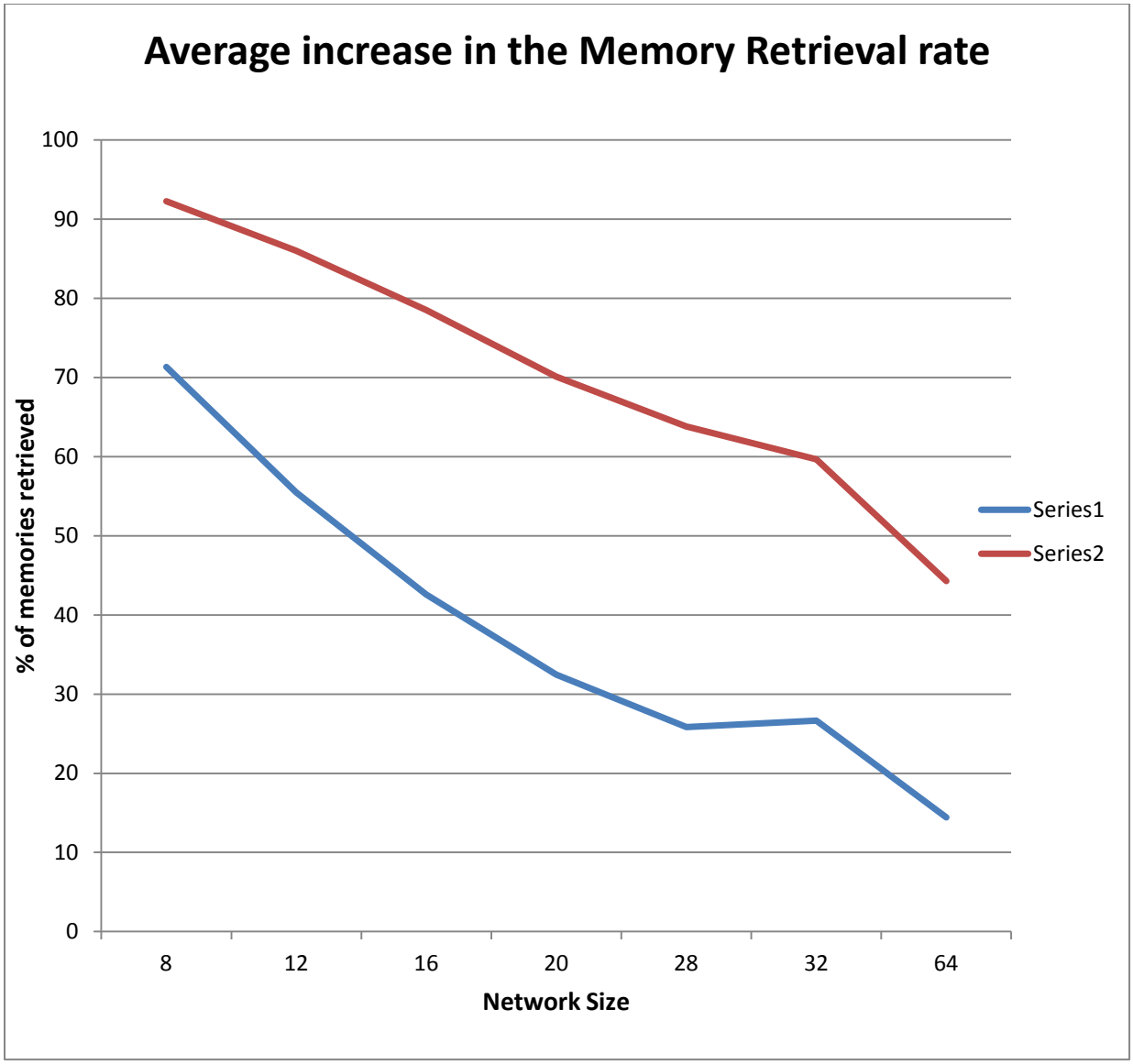


Figure 25: Graph showing increase in average retrieval rate

From the graph we calculated the average increase in the memory retrieval rate on applying the learning algorithm is more than 50%.

The following graph depicts the comparison between the average increase in the number of active nodes in a network in the B-matrix approach and the learning algorithm:

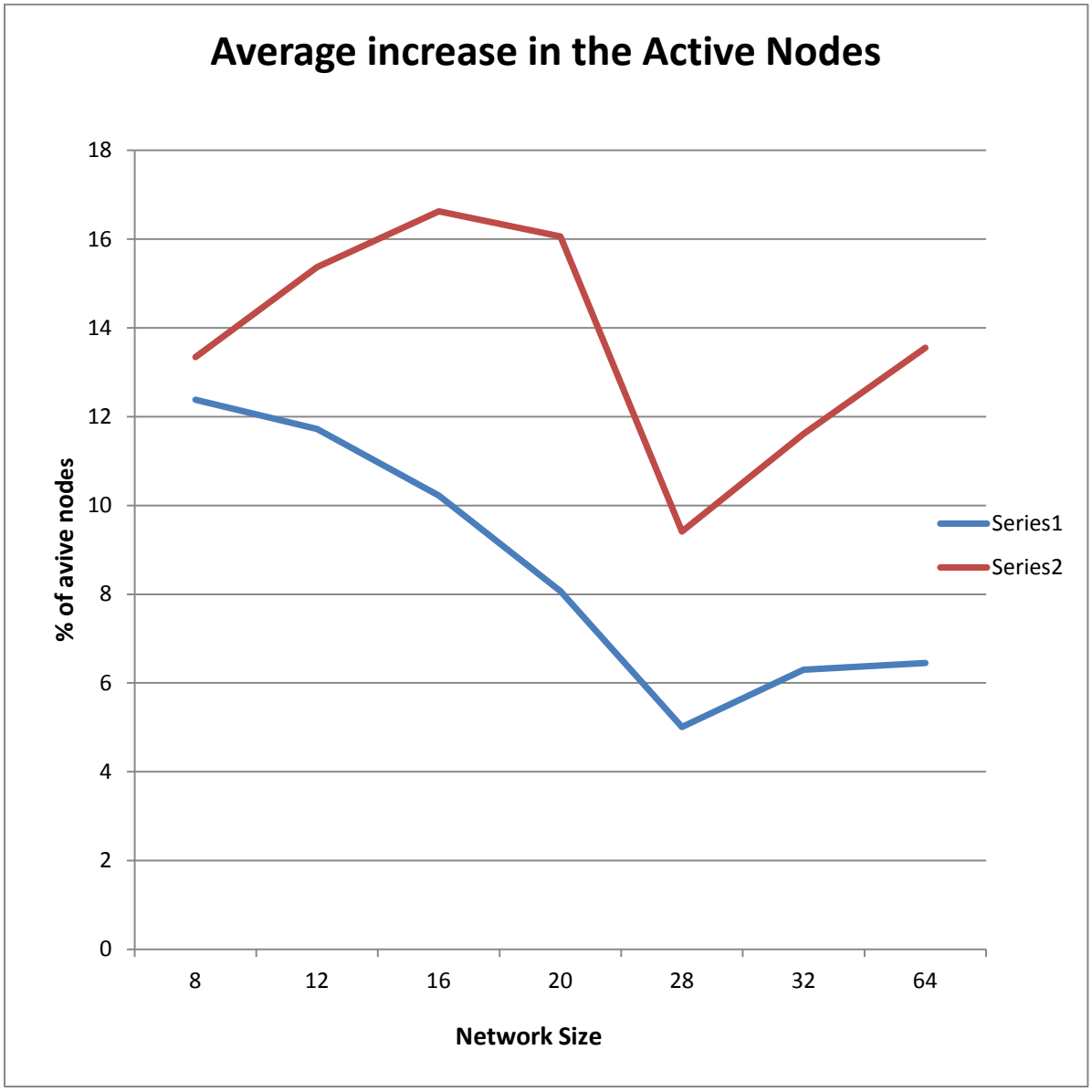


Figure 26: Graph showing increase in the number of active nodes.

From the graph we calculated the average increase in the memory retrieval rate on applying the learning algorithm is approximately 5%.

CHAPTER V

Conclusion

In this dissertation, we suggest an approach to increase the memory retrieval rate from the B-Matrix. This approach mimics the way neurons from different parts of the brain are stimulated to participate in the completion of specific cognitive task. We consider an indexed neural network, which in turn stimulates its corresponding sub-neural networks to retrieve all the pieces of memory required to perform a given task.

We attempt to implement the concept of synaptic plasticity in the artificial neural networks to increase the retrieval rate of the network. We also analyze the change in the state of neurons after the learning algorithm is applied to the network. We have taken measures to reduce the computational complexity of the learning algorithm. This approach will benefit various application fields in the area of artificial neural networks.

REFERENCES

1 S. Kak, Active agents, intelligence, and quantum computing. *Information Sciences*, vol. 128, pp. 1-17, 2000.

2 S. Kak, The three languages of the brain: quantum, reorganizational, and associative. In: K. Pribram and J. King, Editors, *Learning as Self-Organization*, Lawrence Erlbaum, Mahwah (1996), pp. 185–219.

3 D.O. Hebb, *The Organization of Behavior*. Wiley, 1949.

4 J.J. Hopfield, Neural networks and physical systems with emergent collective computational properties. *Proc. Nat. Acad. Sci. (USA)*, vol. 79, pp. 2554-2558, 1982.

5 S. Kak, Artificial and biological intelligence. *ACM Ubiquity* vol. 4, no. 42, 2005.

6 S. Kak, Feedback neural networks: new characteristics and a generalization. *Circuits, Systems, and Signal Processing*, vol. 12, pp. 263-278, 1993.

7 S. Kak, Self-indexing of neural memories. *Physics Letters A*, vol. 143, pp. 293-296, 1990.

8 S. Kak and M.C. Stinson, A bicameral neural network where information can be indexed. *Electronics Letters*, vol. 25, pp. 203-205, 1989.

9 M.C. Stinson and S. Kak, Bicameral neural computing. *Lecture Notes in Computing and Control*, vol. 130, pp. 85-96, 1989.

10 D.L. Prados and S. Kak, Neural network capacity using the delta rule. *Electronics Letters*, vol. 25, pp. 197-199, 1989.

11 D. Prados and S. Kak, Non-binary neural networks. *Lecture Notes in Computing and Control*, vol. 130, pp. 97-104, 1989.

12 Anne-Johan Annema, Feed-Forward Neural Networks Vector decomposition Analysis, Modelling and Analog Implementation, Kluwer Academic Publisher, 1995.

13 Alexander S. Poznyak, Edgar N. Sanchez and Wen Yu, Differential Neural Networks for Robust Nonlinear Control, pp: 28-37, World Scientific Publishing Co., Singapore, 2001, ISBN: 981-02-46240-2.

14 National Institute of Neurological Disorders and Stroke, Life and Death of Neuron http://www.ninds.nih.gov/disorders/brain_basics/ninds_neuron.htm#architecture [last accessed - Feb 11, 2011]

15 S. Kak, Memory retrieved from Single Neurons. 2009. arXiv:0905.3771

16 Colin Fyfe, Hebbian Learning and Negative Feedback Networks, Springer, 1st Edition, 2005, ISBN: 978-1-85233-883-1.

17 S Kak, Single Neuron Memories and the Network's Proximity Matrix. 2009. arXiv:0906.0798

18 S. Haykin, *Neural Networks and Learning Machines.* Prentice Hall, 2008.

19 K.C.Lingashetty, Active Sites model for the B-Matrix Approach, 2010. arXiv:1006.4754v1

20 K.C.Lingashetty, Delta Learning Rule for Active sites Model, OSU, Stillwater, **arXiv:1007.0417v1 [cs.NE].**

21 Robert C. Malenka, Neuropsychopharmacology – 5th Generation of progress, Synaptic Plasticity, Chapter 11, pp: 147-158 , The New England Journal of Medicine, 2002; 347:1289.

22 H. Abdi, D. Valentin, B. Edelman, Neural Networks- Quantitative Applications in the Social Sciences, Sage University paper, Series no. 07-124, 1999, ISBN: 0-7619-1440-4.

VITA

Prerana Laddha

Candidate for the Degree of

Master of Science

Thesis: INCREASING THE CAPACITY OF A B-MATRIX NEURAL NETWORKS

Major Field: Computer Science

Biographical:

Education:

Completed the requirements for the Master of Science in Computer Science at Oklahoma State University, Stillwater, Oklahoma in July, 2011.

Completed the requirements for Bachelor of Technology in Computer Science at Jawaharlal Nehru Technological University, Hyderabad, AP, India in 2009.

Experience:

Web Developer in the Undergraduate Admissions, Oklahoma State University, Stillwater, OK April'11- Present

Teaching Assistant for Computer Systems under Dr. Cline and Theory of Computation under Dr. Debao Chen, Oklahoma State University, Stillwater, OK August'10- May'11

Research Assistant under Dr. Subhash Kak, Oklahoma State University, Stillwater, OK August'10- Dec'10

Graduate Assistant in Computer Science department, Oklahoma State University, Stillwater, OK August'10- Dec'10

Name: Prerana Laddha Date of Degree: July, 2011

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: MEMORY RETRIEVAL IN B-MATRIX NEURAL NETWORKS

Pages in Study: 42 Candidate for the Degree of Master of Science

Major Field: Computer Science

In an effort to create computing structures that are as efficient as the brain at cognitive tasks, interconnected artificial neurons are used in cognitive science and artificial intelligence. In this dissertation we focus on the memory retrieval mechanism in an artificial neural network and suggest an algorithm to increase the memory retrieval rate in the B- Matrix neural network.

The B-matrix is a model of recall by index, where in the activity spreads locally. This approach to neural network function accounts for spreading of activity from one region to others based on adjacency of neurons. We propose an algorithm to increase the memory retrieval rate by identifying the inactive nodes in the network and applying Widrow-Hoff learning function to the weighted matrix. This may be seen as implementing the concept of synaptic plasticity in artificial neural networks to increase their memory capacity.

ADVISER'S APPROVAL: Dr. Subhash Kak