

THE ACTIVE SITES IMPLEMENTATION FOR THE  
B-MATRIX NEURAL NETWORK

By

KRISHNA CHAITHANYA LINGASHETTY

Bachelor of Engineering in Computer Science and Engineering

Osmania University

Hyderabad, India

2008

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, 2010

THE ACTIVE SITES IMPLEMENTATION FOR THE  
B-MATRIX NEURAL NETWORK

Thesis Approved:

Dr. Subhash Kak

---

Thesis Adviser

Dr. John P. Chandler

---

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 like to take this opportunity to express my gratitude and thanks to Dr. K.M. George for supporting me throughout my study at OSU. I have worked on one of the most interesting projects under him. He had mentored and taught me as to how research is done for which I would be ever grateful to him.

I would also like to thank Dr. Michel Toulouse for encouraging me with his invaluable comments and thought provoking questions which led me to some conclusive research. I would also like to thank Dr. John Chandler for his patience and support as the graduate coordinator who has helped me on numerous occasions and Dr. Blayne Mayfield for teaching us that making video games was as interesting and time consuming as playing them.

I would like to thank my friends, Hanaa Musa, Haani Mahmoodi, Kalyan Ram Vempati, Monisha Sabu, Sandeep Reddy, Suresh Babu Thippireddy, Aditya Machani, Karthik Lavangu, Ashu Passi, Shravan Parepalli, Santosh Rao, Bhanu Kishore, Sanketh Kadri, Thejaswi Puthraya, Pranav Kumar, Samuel Anand Raj, Naveen Kumar and Pradeep Gaddam for always being there for me.

I would like to thank my entire extended family for always believing in me and giving me constant support in everything I intended to do. I would like to thank my brother, Vamshi Lingashetty who has always been someone I could totally depend on for moral support.

Lastly and most importantly, I would like to dedicate my thesis to my parents Laxman Lingashetty and Yogeshwari Devi Lingashetty, for believing in me at every point of my life and teaching me to be a better human being.

This research was made possible by a grant from Center of Telecommunications and Network Security (CTANS).

## TABLE OF CONTENTS

| Chapter   | Page |
|---|------|
| I. INTRODUCTION.....                            | 1    |
| Biological Neural Network .....                 | 1    |
| Artificial Neural Networks.....                 | 3    |
| Information Storage .....                       | 4    |
| II. REVIEW OF LITERATURE .....                  | 6    |
| Hebbian Model .....                             | 6    |
| B-Matrix Approach.....                          | 7    |
| Widrow Hoff Learning Rule .....                 | 10   |
| Non-binary Neural Networks .....                | 11   |
| III. METHODOLOGY .....                          | 13   |
| Indexing and Sub Neural Networks.....           | 13   |
| Active Sites .....                              | 14   |
| Identifying the Active Sites.....               | 15   |
| Determining the Update Order .....              | 16   |
| Decreasing Computational Complexity .....       | 17   |
| Delta Rule.....                                 | 18   |
| Experimental Setup.....                         | 19   |
| Calculation of the Proximity Matrix .....       | 20   |
| IV. FINDINGS .....                              | 22   |
| Experiment – 1 : B-Matrix Approach.....         | 22   |
| Experiment – 2 : Active Sites Model.....        | 29   |
| Experiment – 3 : Delta Rule .....               | 34   |
| Experiment – 4 : Non-binary Neural Network..... | 36   |

| Chapter             | Page |
|---------------------|------|
| V. CONCLUSION ..... | 40   |
| REFERENCES .....    | 43   |

## LIST OF TABLES

| Table   | Page |
|---------|------|
| 1 ..... | 31   |
| 2 ..... | 31   |
| 3 ..... | 32   |

## LIST OF FIGURES

| Figure   | Page |
|----------|------|
| 1 .....  | 2    |
| 2 .....  | 3    |
| 3 .....  | 4    |
| 4 .....  | 7    |
| 5 .....  | 9    |
| 6 .....  | 14   |
| 7 .....  | 22   |
| 8 .....  | 23   |
| 9 .....  | 23   |
| 10 ..... | 24   |
| 11 ..... | 24   |
| 12 ..... | 25   |
| 13 ..... | 25   |
| 14 ..... | 27   |
| 15 ..... | 27   |
| 16 ..... | 28   |
| 17 ..... | 29   |
| 18 ..... | 30   |
| 19 ..... | 30   |
| 20 ..... | 33   |
| 21 ..... | 34   |
| 22 ..... | 34   |
| 23 ..... | 35   |
| 24 ..... | 36   |
| 25 ..... | 37   |
| 26 ..... | 37   |
| 27 ..... | 38   |



## CHAPTER I

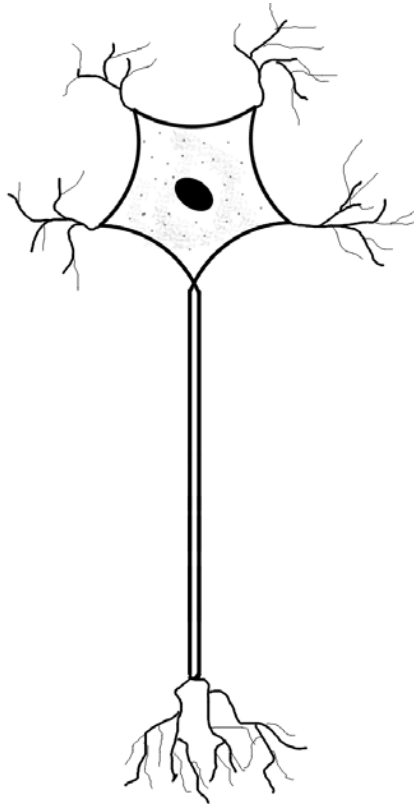
### INTRODUCTION

#### **Biological Neural Network**

The importance of neural networks as a subject is partly owing to the possibility of the comparison of machines to biological brains. The first model of a neural network was proposed by Warren McCulloch and Walter Pitts over fifty years ago. It is believed that the large number of neurons and their hierarchical organization in the brain are a prerequisite for an organism to be conscious and exhibit complex behavior. The neural network acts as an information processing unit that receives information from the organs of its body which are reacting to the external conditions of the environment. The information that is received is decoded and processed to generate an appropriate response.

Historically, neural networks have been compared to computers or machines and vice versa. According to John von Neumann, the description of machines in the future would be in terms of the brain, rather than that of brains in terms of computers [1] .

The fundamental differences between machines and biological neural networks are the built-in parallelism and the redundancy with which a biological network gains reliability as compared to individual stand alone computational systems. The following figure shows the typical biological neuron.



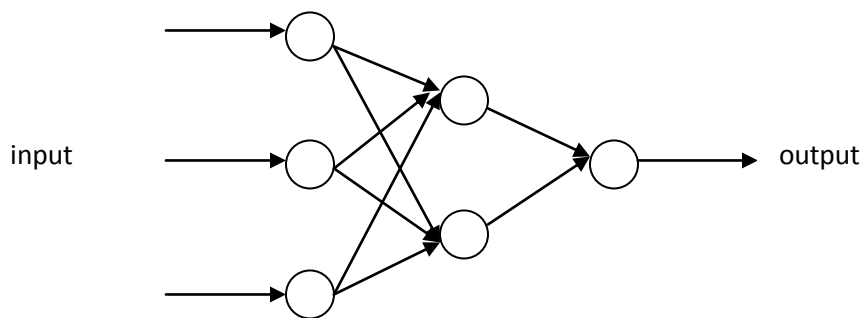
**Figure-1,** A Biological Neuron

As seen in Figure 1, the neuron consists of a cell body which houses the nucleus, a thin long body structure called the axon and the branching structures called dendrites. The dendrites act as connectors to other neurons or to vital organs. The signals that are generated by the organs are carried by these neurons to the brain, which is a complex network of millions of such entities. The brain is thought of as an abstract computational

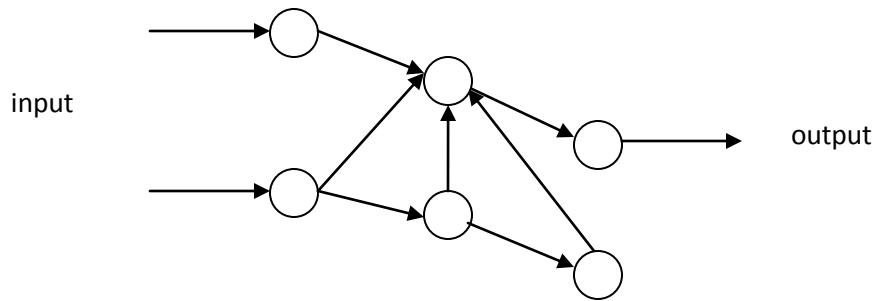
machine for which many models have been proposed, but, none of which match the potential of a biological network. These models stress the issues of information processing and information storage/retrieval.

### **Artificial Neural Networks**

Artificial neural networks are designed keeping into consideration that the information processing capacity of a neuron is limited and that as a whole, a neural network should be able to generate intelligent behavior. 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.



**Figure-2, Feedforward neural network**



**Figure-3,** Feedback neural network

### **Information Storage**

Information storage is generally viewed as a pattern of activity [2]-[7] while in other models, it is seen as information stored at specific locations [8]-[9]. Also, there is the question of the processing power of an individual neuron. Is a neuron limited to processing very low fundamental information? Or is it capable of processing higher level information? There is substantial evidence reported by Quiroga *et al*, that specific neurons are “selectively activated by strikingly different pictures of individuals, landmarks or objects and in some cases, even by letter strings with their names [9]”.

Fifty years ago, the American-born Canadian neurosurgeon Wilder Graves Penfield made an amazing observation [16] . While operating on the brain of patients to cure them of epilepsy, he noticed that as he was stimulating the outer cortex of the brain, patients seemed to recall past experiences. Recently, scientists who were operating on an obese man to reduce his appetite, have confirmed this finding. Scientists who had injected electrodes into the man’s brain and were trying to stimulate certain regions of his brain, to could control his appetite [11] , found that the patient had claimed that he was remembering, “in intricate detail, a scene from 30 years earlier”. This recent find

indicates there are control/activity centers in the brain and certain information might be indexed at specific locations in the brain. When a stimulus is given to this particular neuron or set of neurons, it led to the retrieval of a specific memory.

A mathematical model based on the B-Matrix approach does show how memories are retrieved from a single neuron in a network. This uses the Hebbian rule of learning and can be adapted to multiple such generator neurons. In this thesis, we examine new implications of the B-Matrix approach and show how it may be modified to lead to specific neuron sites with which fixed memories are associated.

## CHAPTER II

### REVIEW OF LITERATURE

#### **Hebbian Model**

The Hebbian model was proposed by Donald Hebb in 1949 [2] . 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. The Hebbian learning rule is given by

$$W_{i,j} = x_i \cdot x_j'$$

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

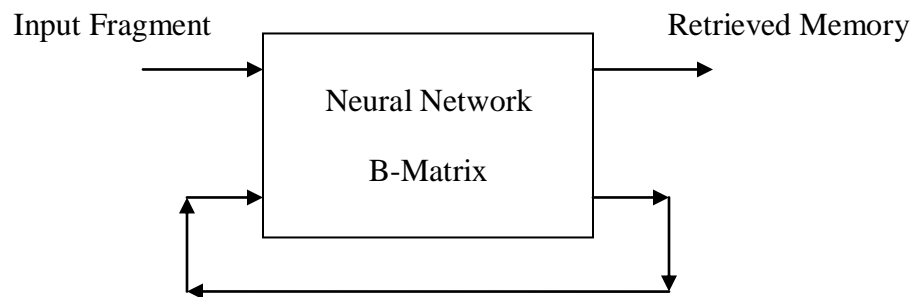
$$x = \text{sgn}(W \cdot x)$$

where  $\text{sgn}$  is the signum function

## B-Matrix Approach

The B-Matrix network is a feedback network with indexed memory retrieval. Developed by Kak, the B-matrix Approach [12] , [13] is a generator model for memory retrieval. In it, the activity starts from one neuron and then spreads to the adjacent neurons to increase the fragment length. The new fragment is fed back to the network recursively until the memory is generated.

The use of the B-matrix approach together with a specified proximity matrix for the neurons was recently shown by Kak [12]. In this thesis, we perform experiments to see the relationship of single neuron memories to their location using the proximity matrix. The location of memories and the capacity for this storage was obtained by experiments on a large number of networks with random memories.



**Figure-4,** B-Matrix generator model

Recollection of memories in the B-Matrix Approach is by using the lower triangular matrix  $B$ , constructed as,

$$T = B + B^t$$

The activity starts from one single neuron and then spreads to additional neurons as determined by the proximity matrix which stores the synaptic strength between neurons. Starting with the fragment  $f^1$ , the updating proceeds as:

$$f^i = \text{sgn}(B \cdot f^{i-1}),$$

where  $f^i$  is the  $i^{\text{th}}$  iteration of the generator model. Notice that the  $i^{\text{th}}$  iteration of the generator model produces only the value of the  $i^{\text{th}}$  binary index of the vector memory but does not alter the 'i-1' values already present.

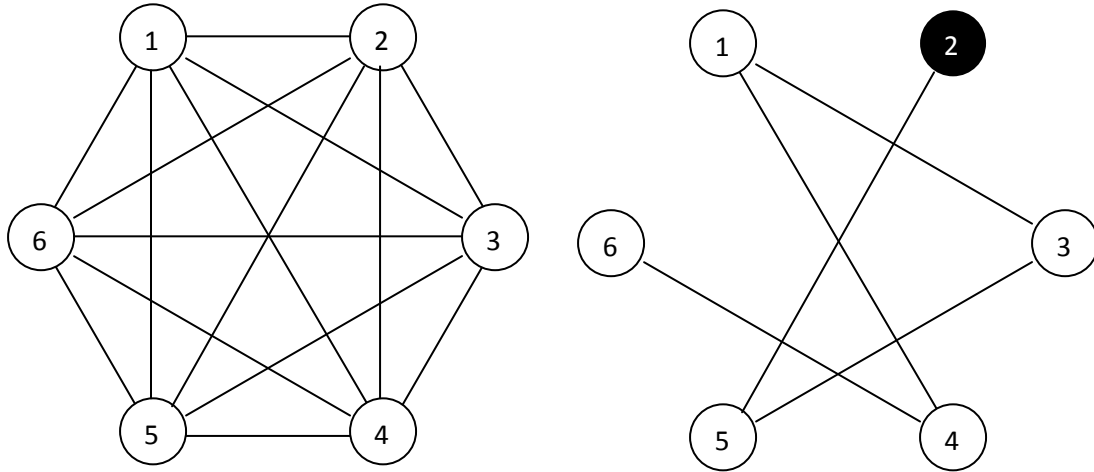
This model relies heavily on the geometric organization of the network. The proximity matrix is a matrix which holds the geometric proximity of each of the neurons with every other neuron in the network.

The neural network of  $n$  neurons may be thought of as a three dimensional network of  $n$  nodes interconnected with each other with different synaptic strength between each node. We can construct a two dimensional graph of the network as a polygon of  $n$  sides with all diagonals connected and each corner being a node. For example, consider the neural network of 6 nodes as shown in Figure 5.

Let us assume without loss of generality that this network is already trained with a set of memories. When retrieving a memory from this network, the activity starts from one node and then spreads to the adjacent nodes as described by the proximity matrix. Assume that the activity starts at the second neuron and spreads from there on. If the



synaptic order given by the proximity matrix is [2 5 3 1 4 6], then memory retrieval proceeds as shown in the figure.



**Figure 5.** Graph showing the Neural Network and Activity Spread from Neuron 2.

Each neuron is capable of storing and retrieving a particular memory by the spread of activity as long as the network is not overwhelmed with information. As seen in the above example, it is possible to retrieve the right memories if we do know what the index of the neuron to be stimulated is, and what should be used to stimulate that selected neuron. Hence indexing plays a major role in the retrieval of memories. To better understand how a complex neural network might function, we introduce the concept of a sub-neural net and an indexing neural net. A different approach to indexing is provided in [10], [15].

This model eliminates the need for a complete vector of memory needed for verification and a partial fragment of memory would be sufficient for recall. 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.

### **Widrow-Hoff Learning:**

The Widrow-Hoff learning rule was proposed to increase the memory storage capacity of the Hebbian network [3] . In this learning, we try to adjust the weights that are stored in the network iteratively to increase the chances of retrieving memories from the network. The idea behind this mode of learning is that as new memories are brought into the network, the learning of these new memories would have an overwriting effect on the previously learned memory.

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 the particular 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,

$$W_{n+1} = W_n + \Delta(W_n) , \text{ where } \Delta(W_n) = \eta(x_i - W_n x_i),$$
  $W$  is the weight matrix,  $x_i$  is the present input, and  $\eta$  is a small positive constant.

This way of learning is “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.

An error vector is estimated for every iteration of the weight matrix adjustment. We then calculate an error term associated with these vectors, and average it over the number of memories that are trained to the network. Defining a good bound on this error term forms a critical problem.

### **Multi-Level/Non-Binary Neurons**

In the discussions above, we were considering binary neural networks. It is not easy to compare such networks to the neural networks in biological organisms, since the biological neurons not only carry electrical impulses of varying magnitude, but they are also associated with a train of voltage spikes. Conversions of all data that is being fed to the network to the binary form can cause loss of information. Non-binary or *n-ary* neural networks were proposed [14] motivated by the complexity of biological neural networks.

Consider the construction of a quaternary neural network [14] instead of a binary neural network. Such a network implements the same principles as does a binary one, with the exception that the neurons now, map to a larger set of activations.

$$x_i = \sum_j T_{i,j} V_j$$

$$V_i = \begin{cases} -3 & x_i < -t \\ -1 & x_i < 0 \\ 1 & x_i < t \\ 3 & x_i > t \end{cases}$$

The problem involved in extending this approach to the B-Matrix approach would involve the inclusion of varying levels of thresholds at each step of the multiplication of the B-Matrix to the fragment, considering that there are a different number of neurons

involved in each step. Specifically, the number of neurons involved increases by *one* in each step. Hence, the selection of a good function which would accommodate the number of neurons as they get incremented in each step forms a highly improbable task.

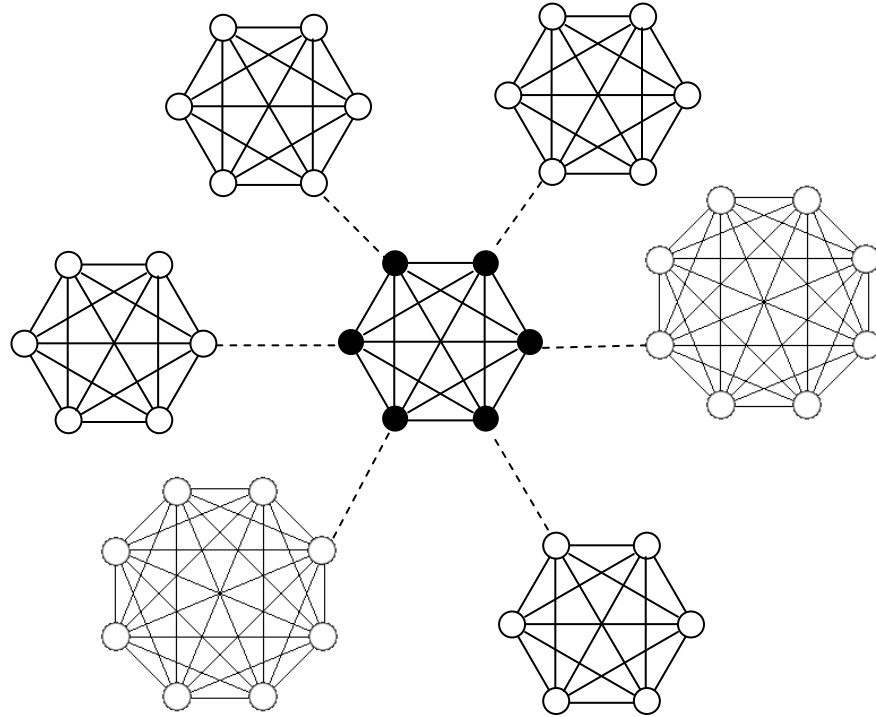
## CHAPTER III

### METHODOLOGY

#### **Sub-Neural Nets and Indexing Neural Nets**

Consider a complex neural network, similar to the human brain, which can store and retrieve memories. This entire neural network can be further divided into much smaller, functional units of networks with the same capabilities as discussed above. In this sub-neural net, each neuron is connected with every other neuron with a proximity matrix that is based on the synaptic strengths of its interconnections.

As mentioned in a previous chapter, it is possible to retrieve the right memories if we know which neuron to stimulate and what signal to stimulate this neuron with. Since a complex neural network is large, it is safe to assume that the network is made up of small sub-neural networks, which are again interconnected with each other to retrieve and store memories. Thus, we can visualize this complex network as a network of networks. Collectively, these neural networks are managed by indexing neural networks, which may help in the retrieval of memories by clamping a certain set of neurons as illustrated in Figure 6.



**Figure 6.** An Indexing Neural Net surrounded by Sub-Neural Nets

The sub-neural nets are the building blocks for the larger network. They are responsible for storing and retrieving memories. The indexing neural network, on the other hand, receives a signal from an external source, say an organ, and then stimulates a few sub-neural nets with varied signals until the memories from each are retrieved. Our proposed active sites model is based on these ideas.

### **Active Sites**

The motivation behind defining active sites comes from the fact that different organs in the human body generate signals which are directed to specific parts of the brain. We assume that this particular signal is not only sent to a specific neural network, but also a particular neuron in the neural network, to target the right neuron with the right stimulus,

so as to generate the specified memories that are stored in the network. A set of activations sites would retrieve for us a memory, and a different set might retrieve another.

Consider a network consisting of  $n$  neurons. According to the Hebbian rules of learning, the activations/synaptic strengths that are contained within the network,  $n(n-1)/2$  in number, can be represented in an  $n \times n$  matrix with the lower and upper triangles containing the specified strengths between successive neurons. By saying that a network can identify a neuron as a possible active site, we propose that the network not only assigns a synaptic strength between two neurons, but also assigns an activation level for the specified neuron. In such a case, when the network is stimulated externally, it is these activation sites that are triggered and these are the sites from where activity spread takes place.

### **Identifying Active Sites**

The neural network identifies the active sites within it during training. We would like that triggering the right neurons with the right stimulus, yields successful memory retrieval. This makes identifying the right neurons to be the crux of the problem. For a given set of memories, say,  $M [1 : m]$ , having a hamming distance of at least 1 among each other and a neural network of size  $n$ , we try to identify those sites in  $M[i]$  which are distinct from the memories of  $M[1 : m] - M[i]$ . Indirectly, we are trying to find those neurons for a particular memory,  $M[i]$ , where the input for that particular neuron is different than the other  $M[1 : m]-M[i]$  memories.

After identifying the various active sites for a particular memory, the neural network then assigns an activation level on these neurons. Different memories have different activation sites. The network cannot identify which activation belongs to which memory, but it can differentiate between activation levels of neurons corresponding to different memories based on the level of activation. For the simulation purposes, the activation levels are represented by prime numbers.

We now consider the problem of trying to retrieve memories from the active sites. Here, we confront questions such as 1) How many of these active sites do we use to retrieve memories?, 2) How do we generate update orders for potentially non-successive active sites?. For this, we propose three methods for determining the update order.

### **Determining the Update Order**

Previously, as we were clamping single neurons, the update order of a neuron was dictated by the corresponding row of the proximity matrix for that particular neuron. However, while considering active sites, there is always a possibility that there are more than one active site for a particular memory, and that the activity spread should ideally take place from these sites until the network reaches a stable state. In order to generate such an order, we propose three methods

- **Arbitrary** : Here, the neurons in the network are selected arbitrarily from the set of neurons for the update order.
- **Averaged** : Here we consider the average of the proximity matrix with regard to the synaptic strength between the active sites and then obtain the new update order relative to these active sites.



- Independent : Here, the partial memories retrieved from all active sites are taken individually along with the values obtained from the ( $B \times clamp$ ) and these values are then taken as a summation over the set of active sites to determine the resultant memory retrieved from the network.

### Decreasing the Computational Complexity

Consider the case where a neural network of size 64 is used and the number of memories that are fed to the network is 6. Assume that the length of the fragment that is being considered for retrieval from this given neural network is 4. In such a case, we need to consider the number of combinations in which the sites are chosen in the classical B-Matrix approach. The total number of combinations of fragments and their originators are characterized by the selection of the number of sites and the fragment with which it is being stimulated.

$$O(BM) = \sum (\text{combination of neurons} * \text{possible fragments})$$

$$O(BM) = (64c_1 * 2^1) + (64c_2 * 2^2) + (64c_3 * 2^3) + (64c_4 * 2^4)$$

$$O(BM) = \sum_{i=1}^r nc_i * 2^i$$

where  $n$  is the size of the network and  $r$  is the chosen fragment size.

Consider that the active sites model is presently being used. In such a scenario, the number of neurons that are presently under consideration are the neurons which are

chosen as active sites for each memory. Hence, every memory has at most  $r$  such sites from which we need to retrieve memories.

$$O(AS) = \sum (\text{combination of neurons} * \text{possible fragments})$$

$$O(AS) = (4c_1 * 2^1) + (4c_2 * 2^2) + (4c_3 * 2^3) + (4c_4 * 2^4)$$

$$O(AS) = \sum_{i=1}^r rc_i * 2^i$$

As we can notice, the number of executions of the retrieval process are reduced drastically, which would help reduce the computational complexity of the network.

### **Delta Rule for the Active Sites Model**

The Widrow-Hoff learning rule was implemented with one goal in mind, to incrementally increase the memory retrieval capacity of the Hebbian model. At each iteration of a particular memory, the effectiveness of the retrieval capacity of the network is improved. Hence, comparing the results of the increase in Hebbian model to the B-Matrix approach or the active sites model would not be justified.

Since at each step, the delta rule computes the change it needs to make to the weight matrix such that it can accommodate the incoming memory into the network. For the same result to be expected from a delta rule for the B-Matrix approach, we need to know which site has to be selected for learning the particular memory. Hence, the active sites model becomes the default model for implementing a delta rule for the B-Matrix approach.

Once the site/sites are chosen by the active sites model, the delta rule would then update the rows individually until the desired memory is retrieved from the weight matrix.

$$\Delta_{i+1,k} = 0, \quad \text{if } \text{sgn}(B \cdot f_i) = \text{input}_{i+1}$$

$$= \text{sgn}(\text{input}_{i+1} - \text{sgn}(B \cdot f_i)) * \text{input}_k * \eta, \quad \text{otherwise}$$

where  $\eta$  is the learning constant.

The delta rule discussed can be applied to a non-binary neural network, as we can specify the threshold individually for each level at learning. This model solves the problem of implementing a non-binary neural network for the B-Matrix approach.

The delta rule can be implemented with having one active site per memory and multiple active sites per memory. Forcing the restriction of having only one active site per memory, gives us only 16 unique active sites per memory and hence, a maximum of only 16 memories can be trained to the network. On removing that restriction however, we have many combinations of active sites available and hence, a higher probability of memory storage. For generating the update order of multiple active sites, the averaged method mentioned above is used.

## **Experimental Setup**

The proposed method was programmed in Matlab and used to carry out the above procedure for a given number of iterations. In each iteration, the neural network is trained incrementally, one memory at a time, i.e., once a memory is fed into the neural network,

the process of counting the number of stored memories and the number of retrieved memories is executed. After execution, the program generates the following output :

- 1) A plot of the average of the number of memories stored against the number of memories fed to the network.
- 2) A plot of the average of the number of memories retrieved against the number of memories fed to the network.
- 3) A 2-D graph of the neural network with highlighted generators.

### **Calculation of the Proximity Matrix:**

Since we do not have a physically defined neural network and we are simulating one, we need to create the proximity matrix for the simulated network. The labeling of neurons as  $[ 1,2,\dots,n ]$  comes from the fact that we select a neuron to be labeled as  $1$  and the proximity order or the update order defines the labeling of the rest of the neurons as  $2,3,4,\dots$  in the increasing order of synaptic strength. Since the simulated neural network is already labeled, we need to create a proximity matrix such that it has the update order of  $[ 1 , 2 , 3 , \dots,n]$ . Having established this, we define a bound on the maximum geometric strength value possible (increasing order of proximity matrix values implies decrease in the geometric strength of neurons). Consider that the maximum possible value for any element in the proximity matrix can never exceed  $(n-1)$ . So the rest of the elements (excluding the update order of  $[ 1 , 2 , 3 , \dots,n]$  and having all the diagonal elements to be zero) are filled with random values in between  $0$  and  $(n-1)$ . This implementation works fine, except that we are not constructing a *fair* proximity matrix.

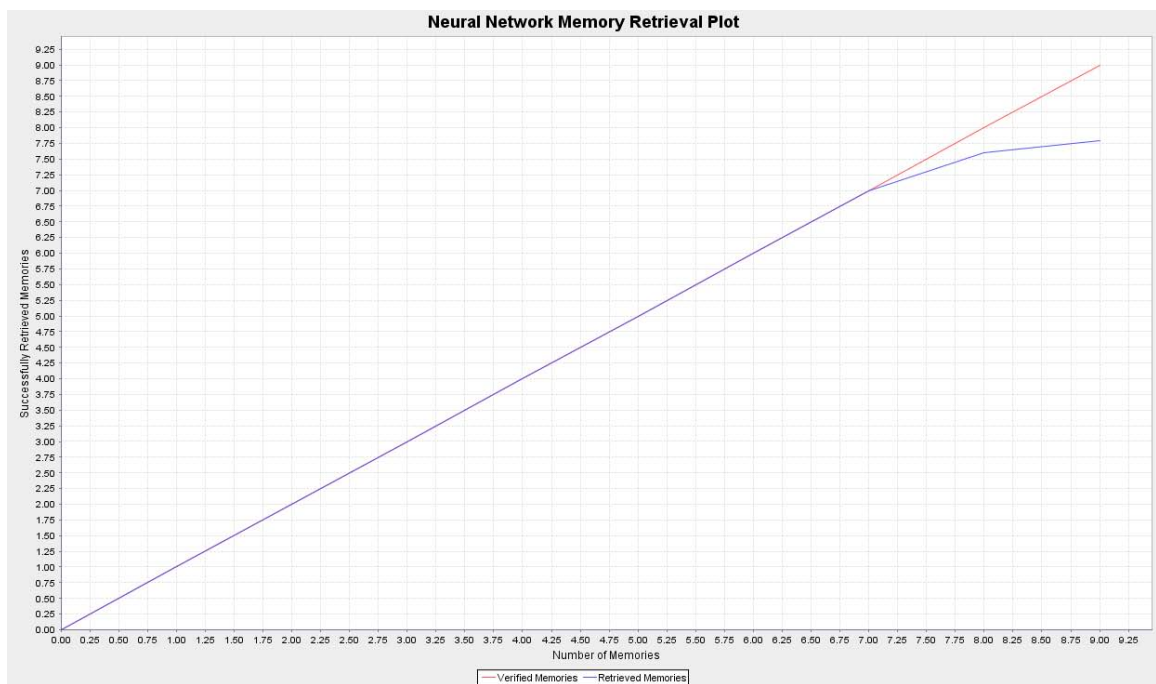
Since we are pre-specifying a proximity order of [ 1 , 2 , 3, ...n ], it is quite possible that neuron-1 might have a very high synaptic strength with neuron-2 and hence, always tend to produce an update order that starts with [ 2 , 1 , .... ]. To give a fair chance for all the other neuron to be able to be geometrically closer to such neurons, the assignment of the proximity values for the update order [ 1 , 2 , ....n ] should be closer to the mean of all the possible values for the proximity matrix. Hence, the values to be associated with the update order of [ 1 , 2 , ...n ] need to be closer to  $n/2$ .

## CHAPTER IV

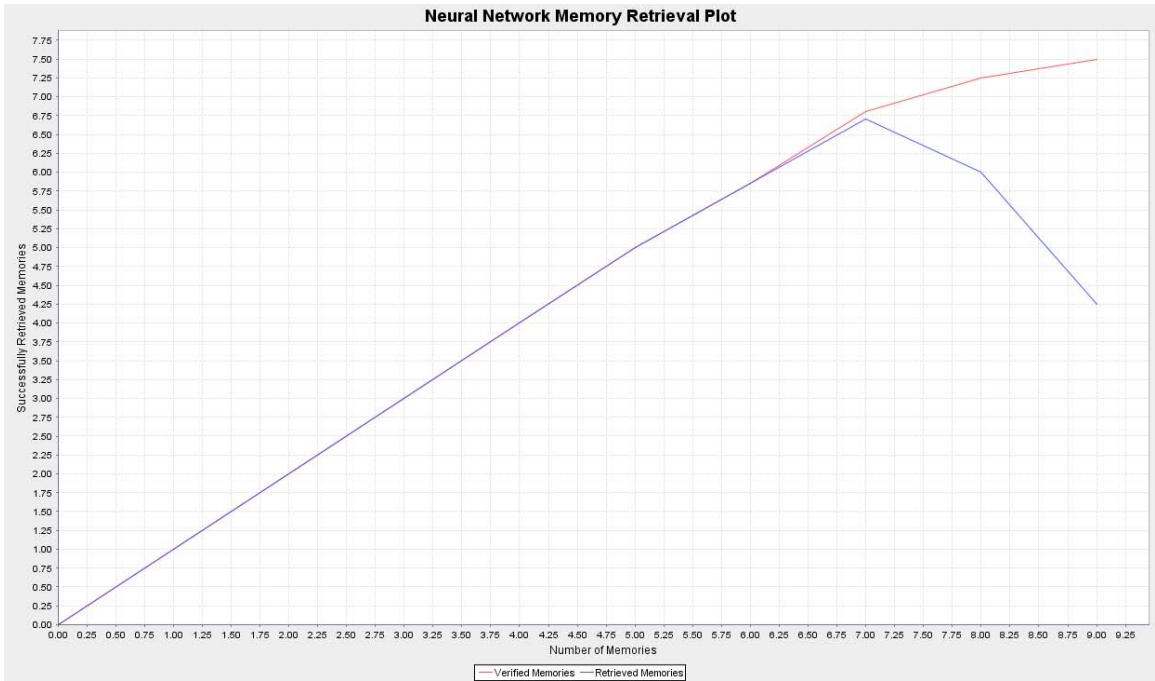
### FINDINGS

#### Experiment – 1 : B-Matrix Approach

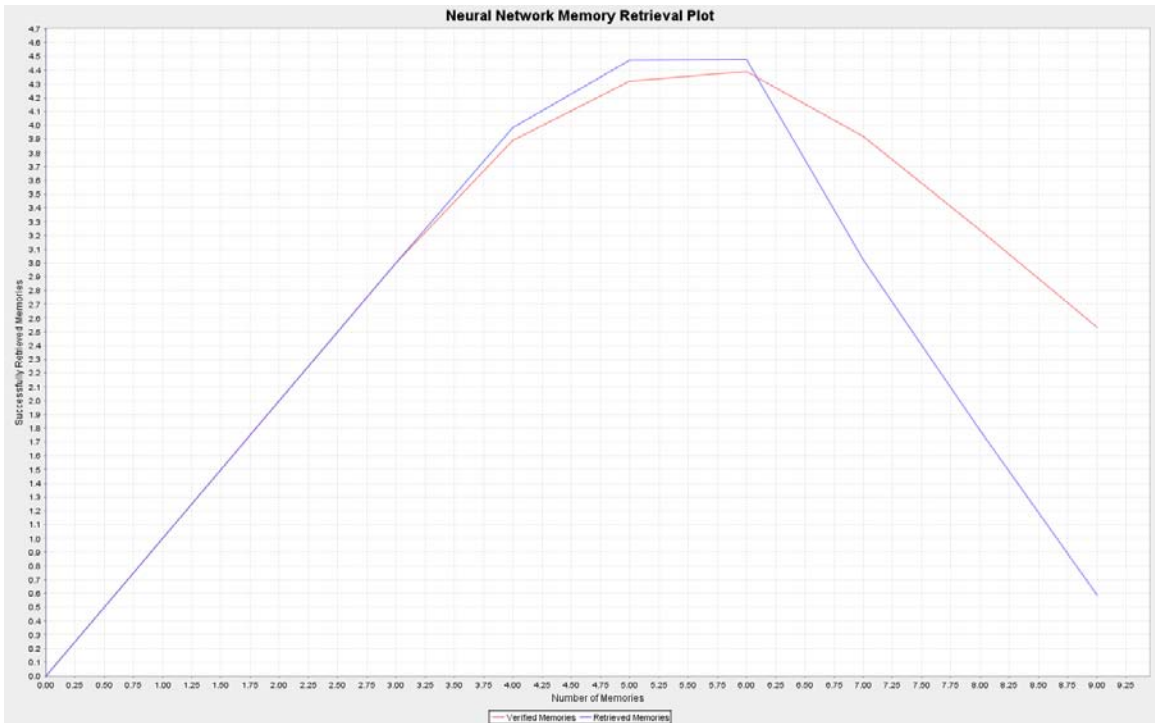
##### A. Relationship between the size of the network and the number of one-bit generators



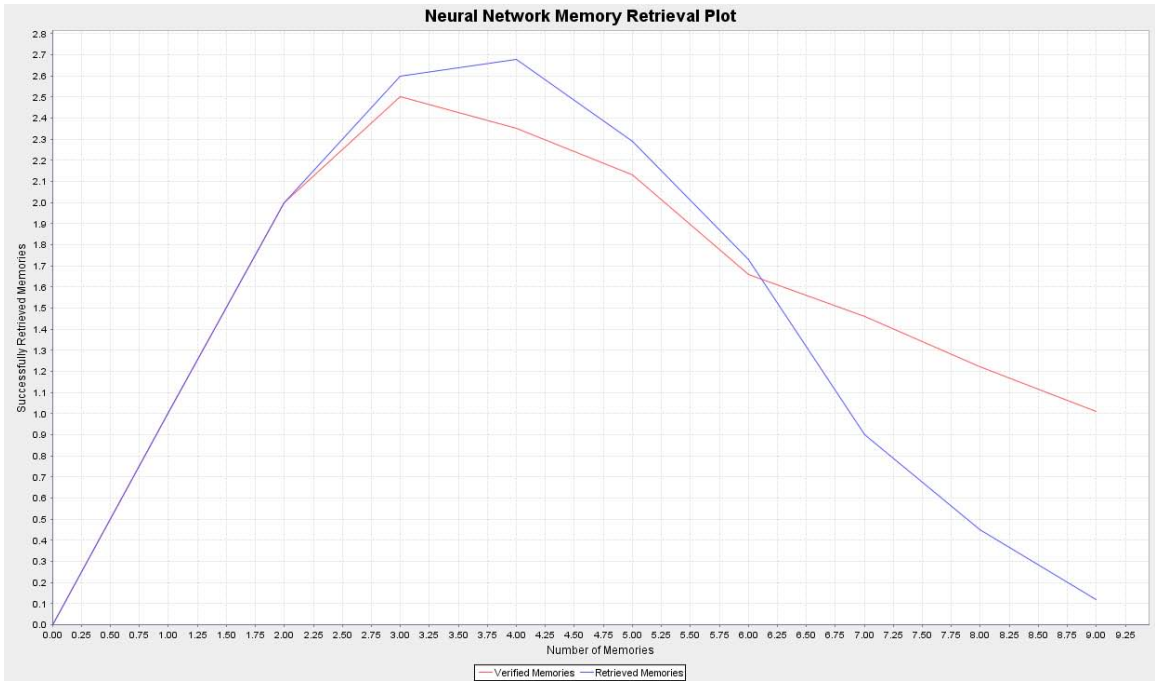
**Figure 7.** 1024 neurons, B-Matrix approach



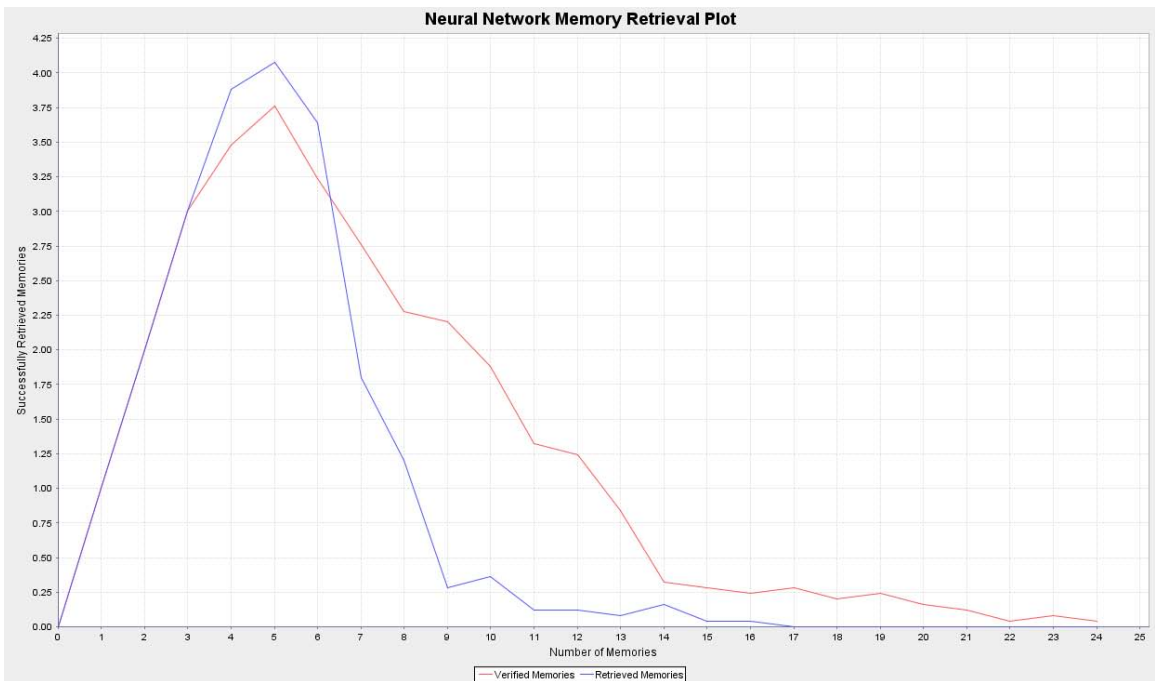
**Figure 8.** 512 neurons, B-Matrix Approach



**Figure 9.** 256 neurons, B-Matrix Approach

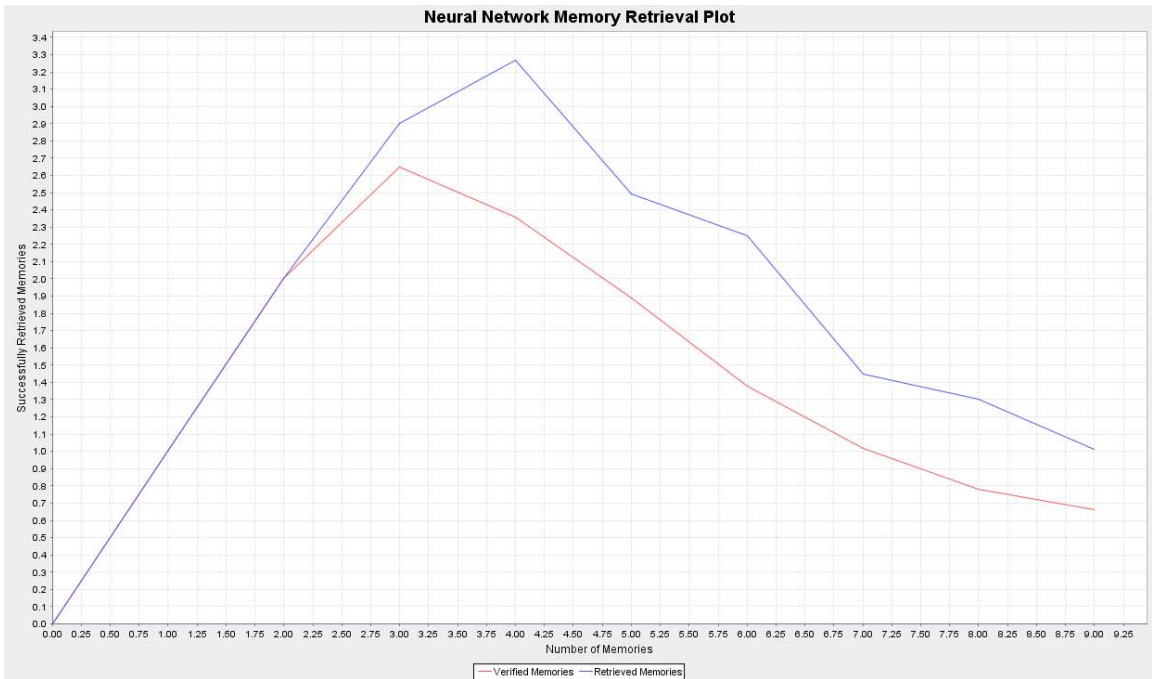


**Figure 10.** 128 neurons, B-Matrix Approach

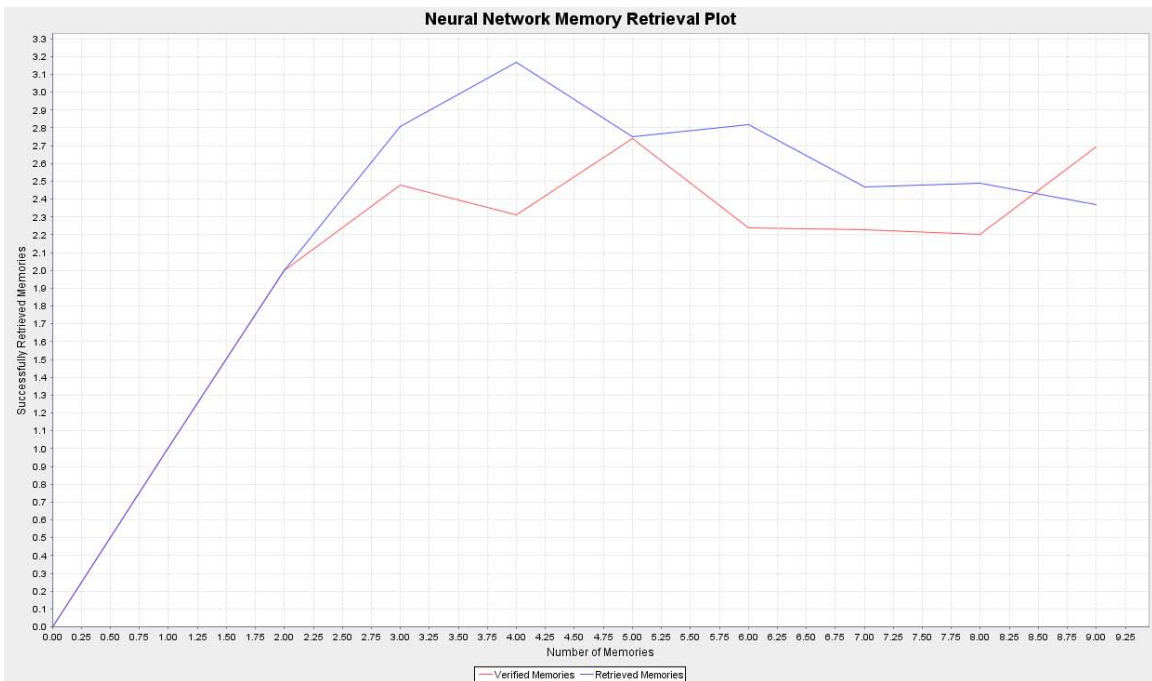


**Figure 11.** 64 neurons, B-Matrix Approach





**Figure 12.** 32 neurons, B-Matrix Approach



**Figure 13.** 16 neurons, B-Matrix Approach

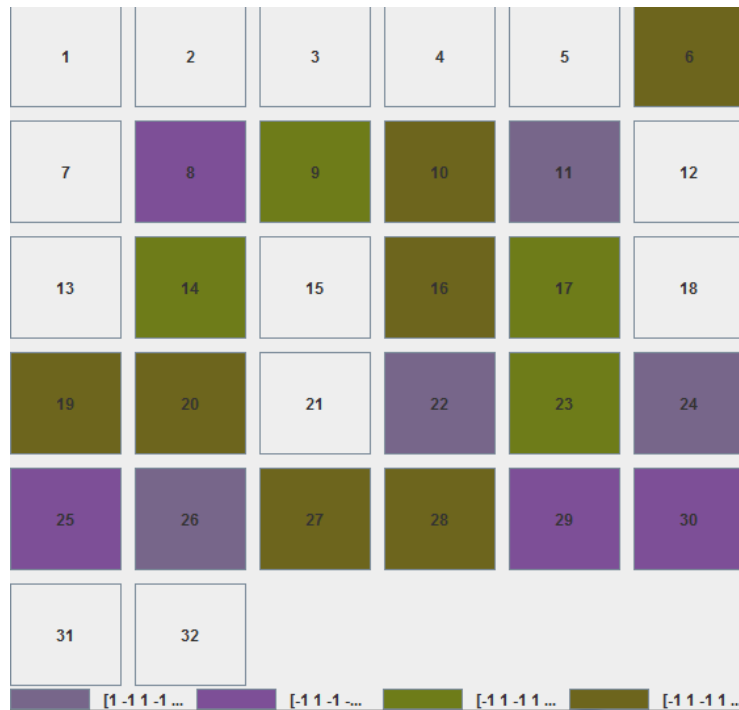
As can be noted from the graphs ( Figures 7 – 13 ), the number of retrieved memories does increase with the increase in the fed memories until a certain point and declines from there on. This is expected, When we overwhelm the network with lots of information, the network becomes incapable of storing additional memories and may even lose the memories that it had already stored in it.

Looking at the graphs, we can notice that for smaller neural networks, the B-Matrix approach gives us a better chance at retrieving more memories as compared to the traditional approach. But as the number of memories increases, the memories retrieved by the B-Matrix approach fall below the traditional approach. Also as the size of the neural network increases, the effectiveness of the B-Matrix approach decreases. This can be observed in the 512 and the 1024 neural network example, as the retrieved memories using the B-matrix approach never exceeds the traditional method.

B. *The number of unique generators associated with a network*



**Figure 14.** 16 neuron generators



**Figure 15.** 32 neuron generators



**Figure 16.** 32 neuron generators

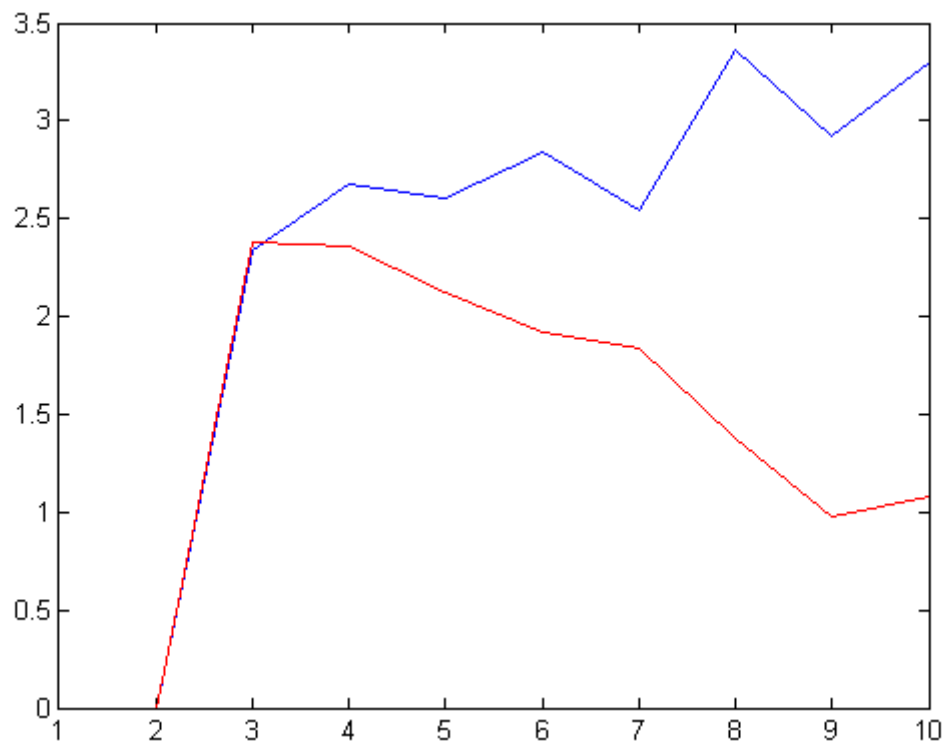
The Figures 14 – 16 show the graphs of a neural network of 16, 32 and 64 neurons, trained with 4, 4 and 5 memories respectively. The highlighted neurons are the ones that have generated a memory successfully. The rest of the neurons are the ones that have not successfully generated any memory. Each color of the nodes corresponds to a different memory.

From the figures, we can see that as the size of the neural network grows, the number of neurons that can successfully generate a stored memory also increases, but the percentage of neurons that do not generate any memory increases (31.25%, 40.62%, and 75%). By this we can conclude that as the size of the network increases, the increase in the number

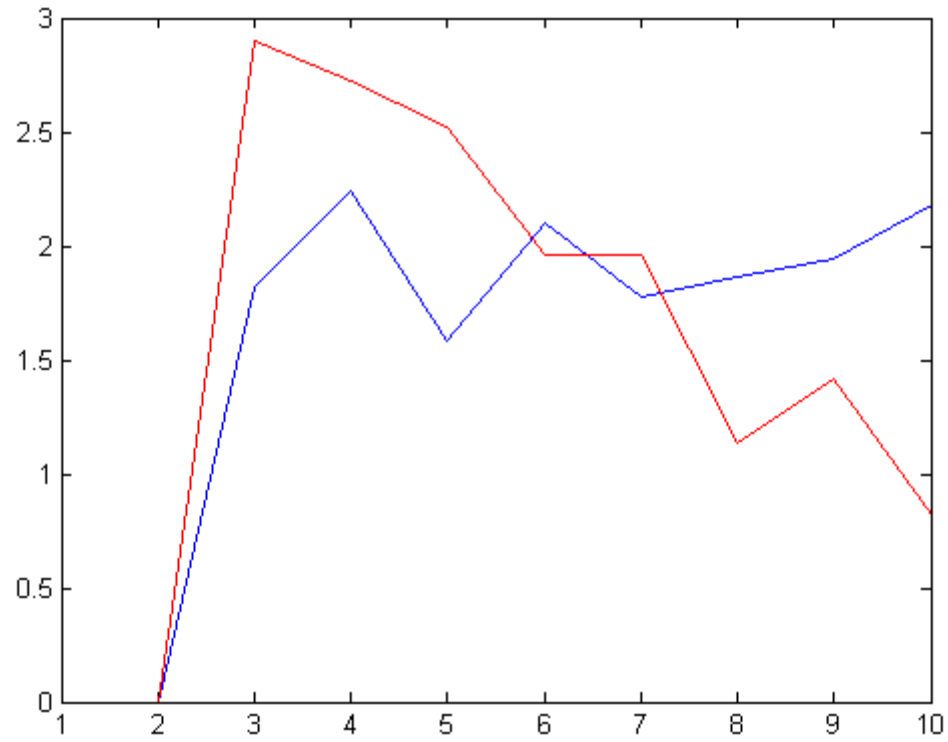
of generators (memory retrieving single neurons) is not proportional. Another interesting property relates to the number of neurons generating a single memory. Even though the number of generators that produce a particular memory may increase with the increase in the size of the network, the percentage of generators that produce a particular memory is decreasing (25%, 21.87% and 7.81% in the best cases).

## Experiment – 2 : Active Sites Model

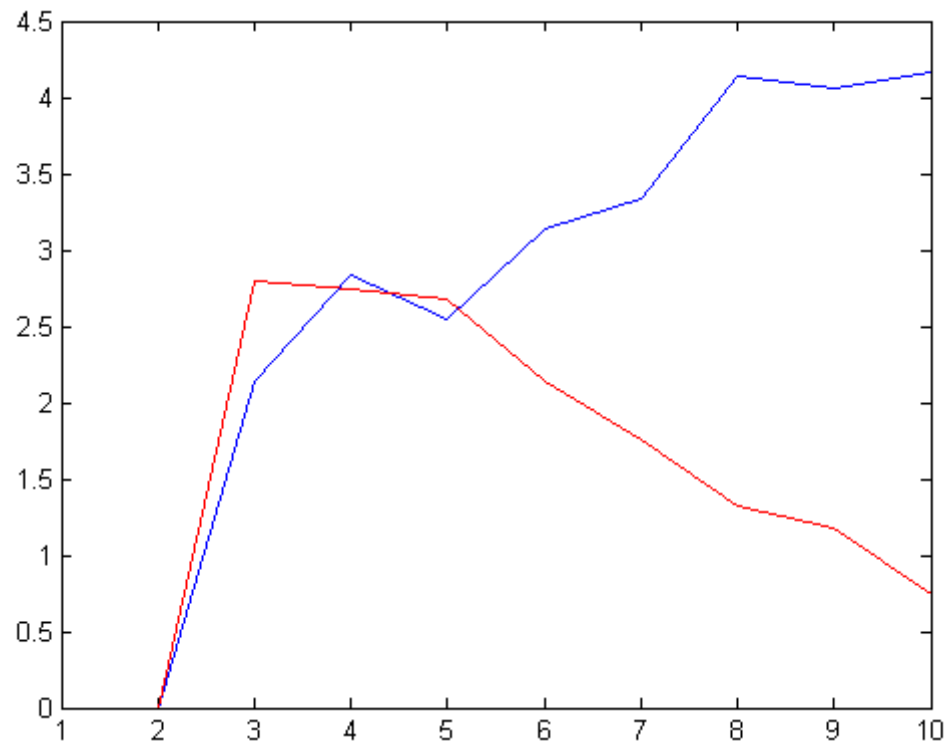
### A. Relationship between the size of the network and the number of memories retrieved



**Figure 17.** 16 neuron Active Sites model – Method 1



**Figure 18.** 16 neuron Active Sites model – Method 2



**Figure 19.** 16 neuron Active Sites model – Method 3

The graphs, Figures 17 – 19, show the results obtained from the three methods of determining an update order. Tables 1, 2 and 3 depict how successful memory retrieval is as the size of the network increases using the arbitrary method, averaged method and the independent method respectively.

| Number of Neurons | Trained Memories | Successful Retrieval |
|-------------------|------------------|----------------------|
| 12                | 8                | 3.4                  |
| 16                | 8                | 3.2                  |
| 20                | 8                | 3.2                  |
| 24                | 8                | 3                    |

Table 1

| Number of Neurons | Trained Memories | Successful Retrieval |
|-------------------|------------------|----------------------|
| 12                | 8                | 2.4                  |
| 16                | 8                | 2.4                  |
| 20                | 8                | 2.3                  |
| 24                | 8                | 2.1                  |

Table 2

| Number of Neurons | Trained Memories | Successful Retrieval |
|-------------------|------------------|----------------------|
| 12                | 8                | 4.5                  |
| 16                | 8                | 4.3                  |
| 20                | 8                | 4                    |
| 24                | 8                | 3.8                  |

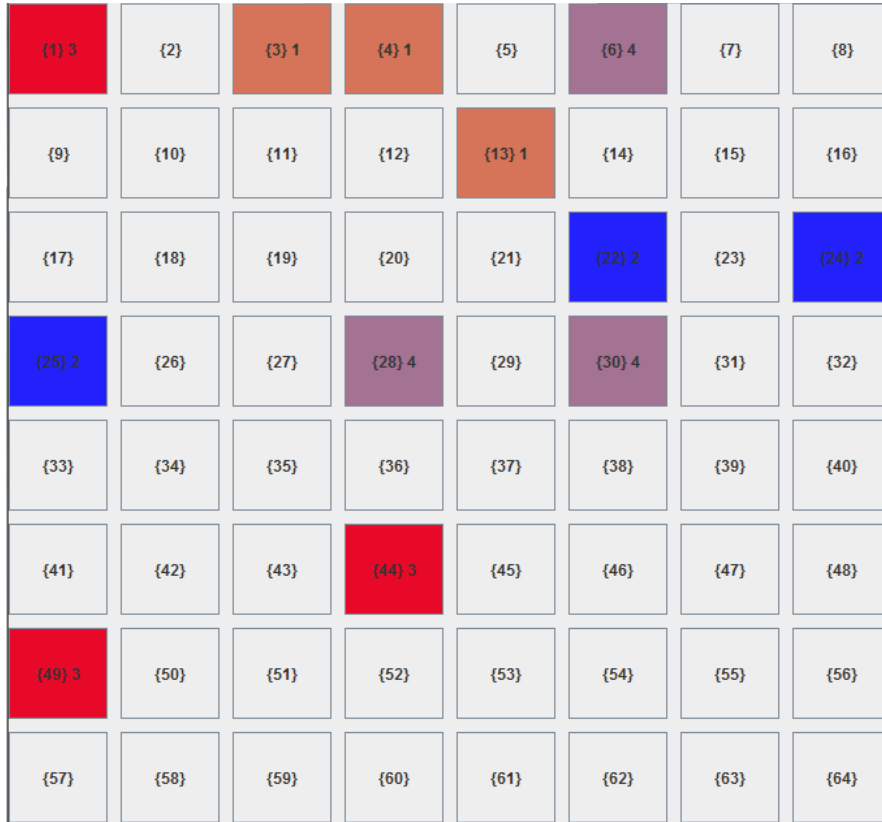
Table 3

The tables above show that the memory retrieval from the neural network is high, when the size of the neural network is comparatively small. As the size of the network grows, the memory retrieval capacity of the network decreases. Although, the pattern that the three methods follow seem to be similar, the rate of the decrease in the memory retrieval capacity does seem to increase across these methods.

We note that the number of memories that are being retrieved are comparably more than the number of memories that were retrieved using the previous approaches. We also notice that the averaged method does not give us a better chance at retrieving memories when compared to the arbitrary or the independent methods. We also notice that as the number of memories being trained to the network increases, the number of verified memories gradually decreases, but the methods proposed do show an increase in successful retrieval.



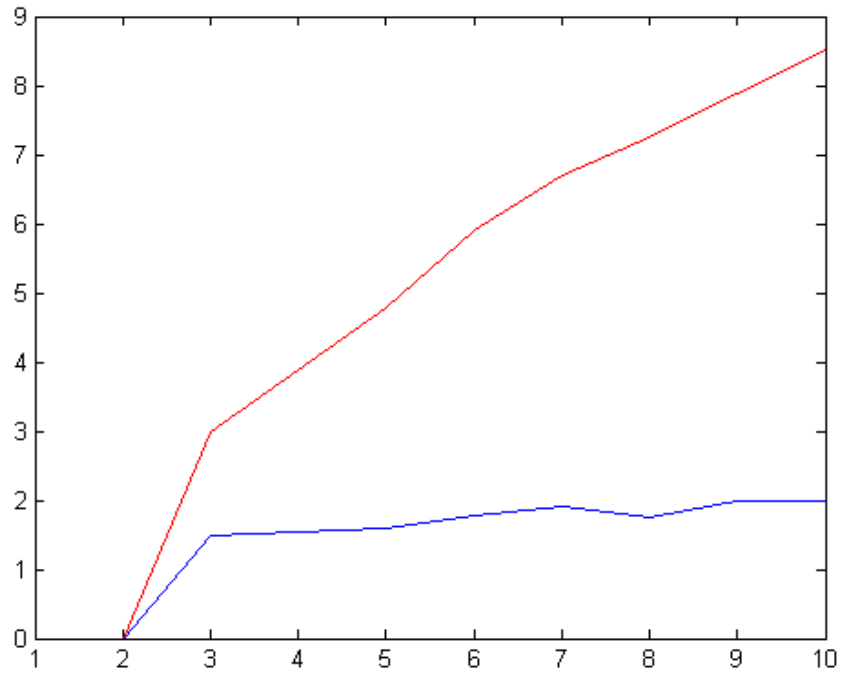
*B. Reduction in computational complexity*



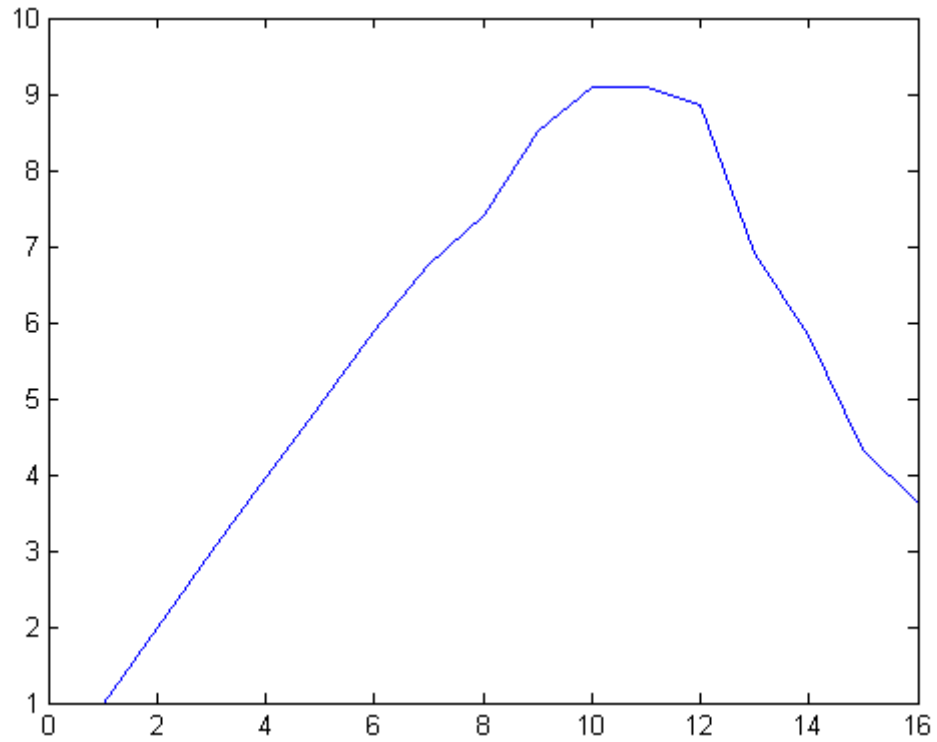
**Figure 20.** Active Site Generators

Figure 20 shows the 2-dimensional spread of the active sites through the neural network. Each color represents a different set of active sites pertaining to a separate memory. These groups of neurons give us a better chance at memory retrieval than the classical approach which reduces the computational complexity.

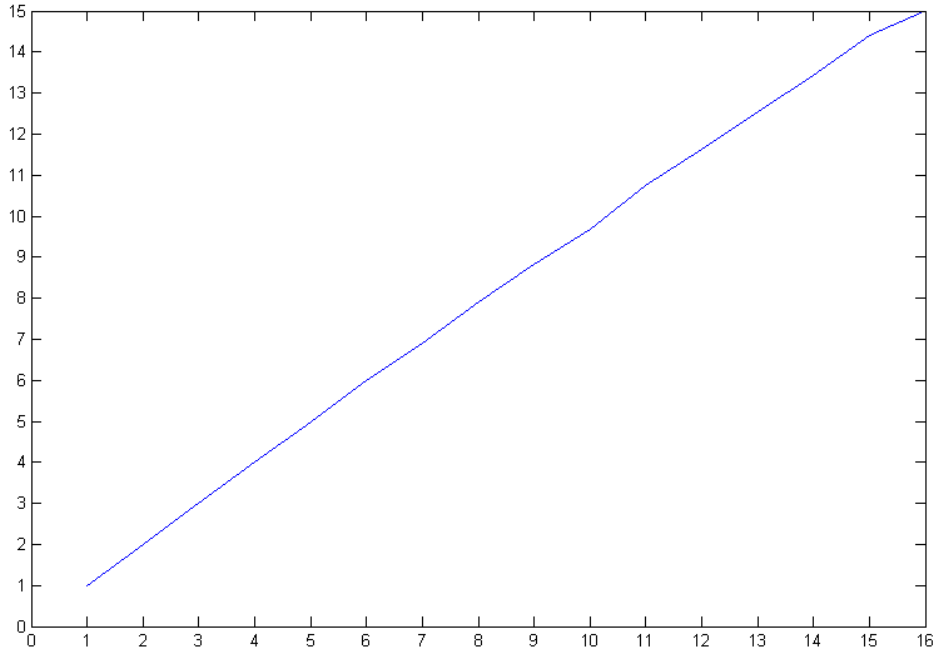
### Experiment – 3 : Delta Rule



**Figure 21.** Active sites model with Widrow-Hoff learning



**Figure 22.** 16 neuron Delta Rule



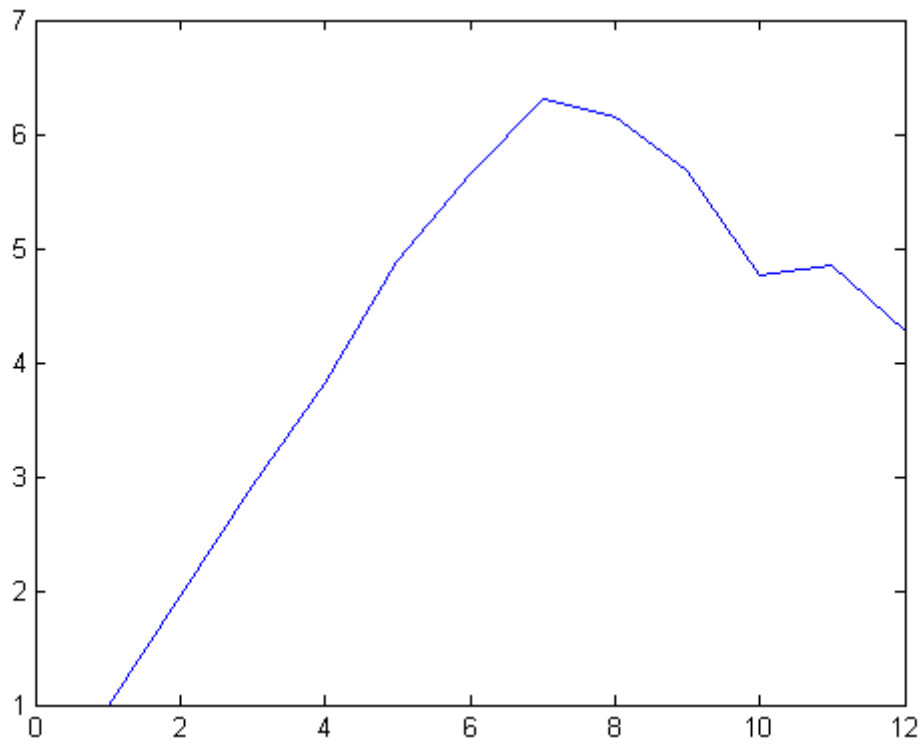
**Figure 23.** 32 neuron Delta Rule

In Figure 21 , we notice that the number of memories retrieved never goes higher than 2 memories, no matter how many memories are fed to the network. Hence, using the Widrow-Hoff learning rule to the active sites/B-Matrix approach is not justifiable as this rule was developed to increase memory capacity of a Hebbian model and not the B-Matrix approach.

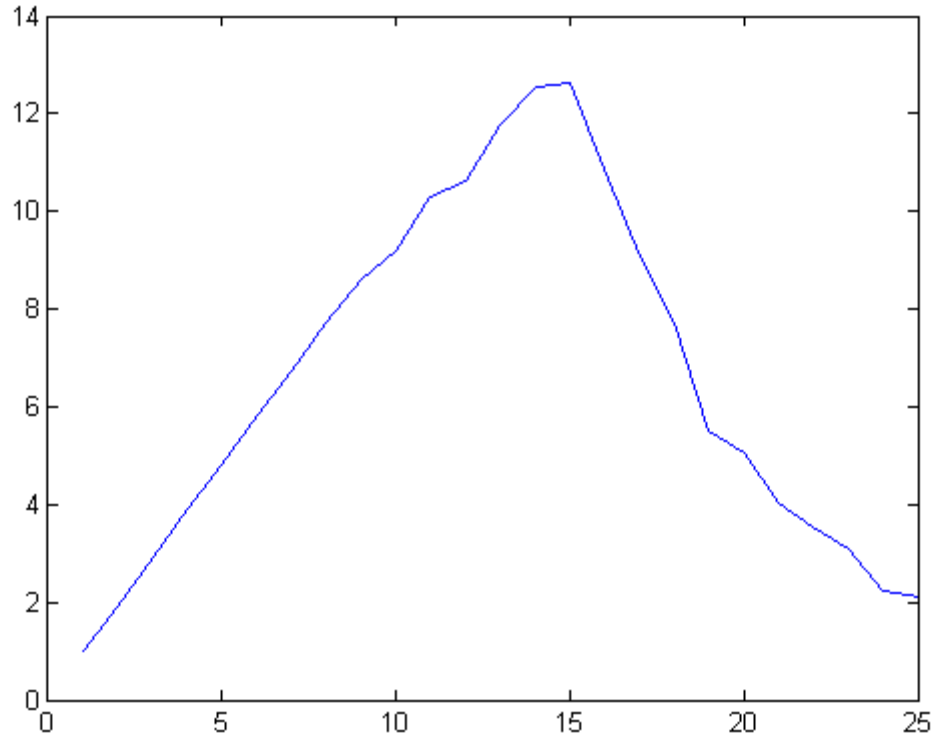
Figures 22 and 23 show the results of the delta rule applied to the active sites model. We can notice from the figures that the number of memories that are being retrieved from the neural network has increased significantly. The increase in retrieval capacity of the neural network has increased more than a 100% as compared to the active sites approach. The number of retrieved memories peaks off slightly over the  $n/2$  mark, where  $n$  is the number of neurons in the network. For a neural network of size 16, we can retrieve nearly

9 memories out of the 10 that are trained to the network. After that, the number of memories stored falls. The reason for this could be that since 16 neurons are available for potential active sites, the later memories are given active sites where they aren't actually as active, but they keep rewriting the already learnt memories, which reduces retrieval.

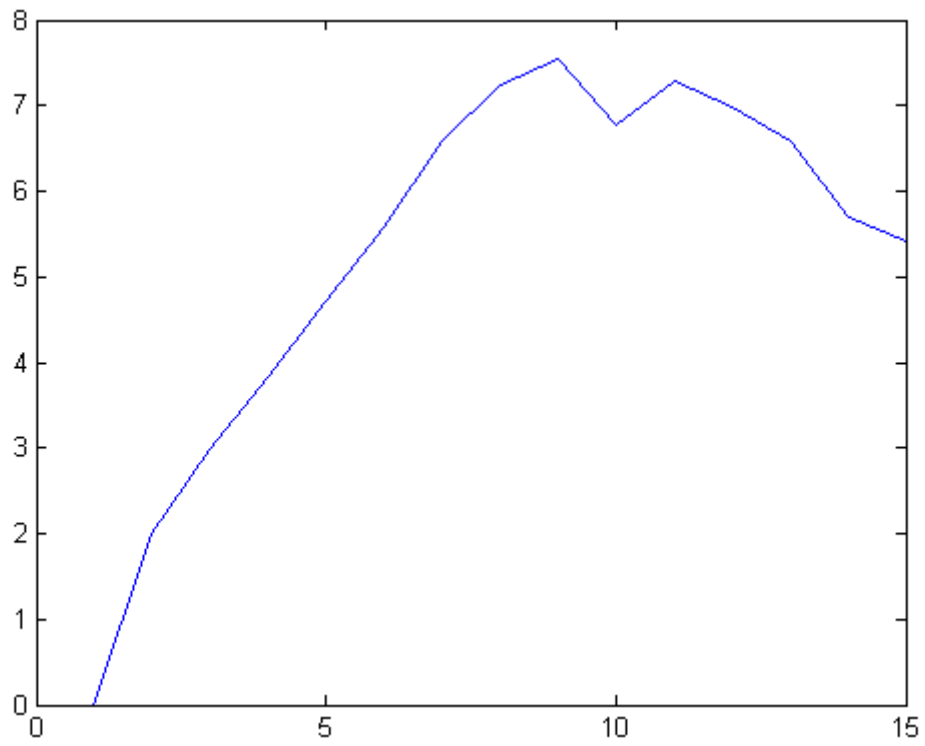
#### Experiment – 4 : Non-Binary Neural Network



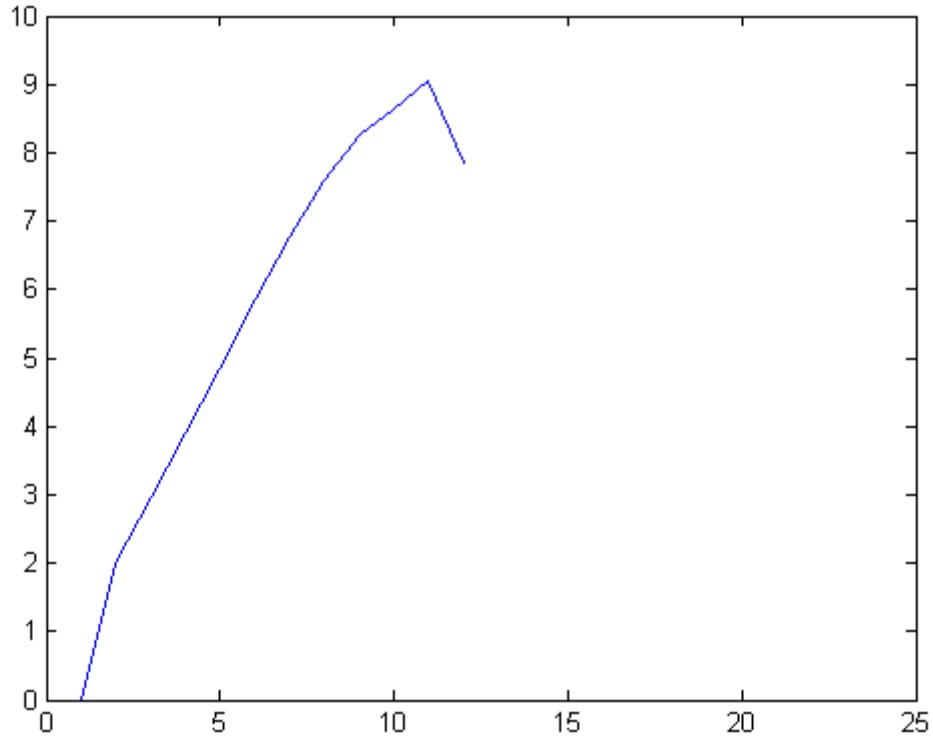
**Figure 24.** Non-binary 16 neuron Delta Rule



**Figure 25.** Non-binary 32 neuron Delta Rule



**Figure 26.** Non-binary 16 neuron Delta Rule with 2 Active Sites per memory



**Figure 27.** Non-binary 16 neuron Delta Rule with 3 Active Sites per memory

Figures 24 and 25 show the delta rule being applied to a non-binary neural network to networks of size 16 and 32 respectively. We notice that non-binary retrieval does not fare as well as the binary neural network. The maximum retrieved memories peaks off at around 6.5 and 13 for the non-binary model as compared to the 9 and 16 of the binary model. However, considering the amount of information that is stored in these networks, the non-binary memory incorporates twice as much information than the binary model. Hence, the non-binary model stores more information than the binary model.

Figures 24, 26 and 27 show the application of the delta rule with more than one active site per memory. We see that the number of memories retrieved successfully increases with the increase in the number of active sites (6.5 for one active site, 7.5 for 2 active

sites and 9 for three active sites). Increasing the number of active sites does increase memory retrieval but increases the computational complexity of the model, as there are an increased number of combinations of sites and the fragments of memory.

## CHAPTER V

### CONCLUSION

In the examination of indexing of memories by the B-Matrix approach, we found that triggering the right neurons with the right stimulus gives us the retrieved memory. This may be compared to the thinking or recollecting action performed by humans. When we are thinking about a past memory or an event, it is quite possible that we might stimulate an indexing neural network, which then stimulates its corresponding sub-neural networks until the pieces of memory are retrieved. If the retrieved fragments collectively are not the desired memory, then the indexing neural network may be subjected to another stimulus in anticipation of successful retrieval.

Further work needs to be done on the retrieval patterns and behavior of single generators and multi generators consisting of multiple neurons. Also, as this model is heavily based upon the geometrical proximity of the neurons in the network, there needs to be further research on the properties of the proximity matrix.



In the active sites model, we were able to sustain a retrieval capacity even though the number of memories being fed to the neural network have been increasing. This can be attributed to the fact that as we keep accumulating new memories, some of our old memories fade away. The proposed model thus, incorporates the B-Matrix approach and determines a way of storing or retrieving memories with better computational complexity and maintaining a sustained retrieval of memories.

The Widrow-Hoff model proposed for higher retrieval capacity of the Hebbian model does not increase the memory retrieval capacity of the B-Matrix or the active sites model. The proposed delta rule increases the memory retrieval capacity of the neural network by more than a 100% and using more active sites per memory increases the retrieval capacity.

The delta rule proposed in this thesis gives us a new insight on how a complex biological neural network might perform. A complex biological network need not be a binary model, for, the communication between the neurons and the brain has many non-binary elements to it.

This work provides an understanding into how sub-neural nets in a large network might perform. Future work could include the functioning of the indexing neural network and how it would integrate with the workings of the sub-neural network. Other interesting questions that come to mind are : 1). Are there other ways in which the update order may be modified? 2) Can a neural network theoretically evolve over time where it changes the proximity relations it has for the neurons that belong to it? 3) What is the complexity of

such an approach? 4) How does the indexing neural network work? 5) Are the memories indexed based on temporal significance? Or are they based on relational significance?

## REFERENCES

- [1] Raul Rojas, “Neural Networks : A Systematic Introduction”, Springer, 1996.
- [2] D.O. Hebb, *The Organization of Behavior*. Wiley, 1949.
- [3] S. Haykin, *Neural Networks and Learning Machines*. Prentice Hall, 2008.
- [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 and J. F. Pastor, Neural networks and methods for training neural networks. *US Patent* 5,426,721, 1995.
- [6] S. Kak, On training feedforward neural networks. *Pramana*, vol. 40, pp. 35-42, 1993.
- [7] S. Kak, “Faster web search and prediction using instantaneously trained neural networks,” *IEEE Intelligent Systems*, vol. 14, pp. 79-82, November/December 1999.
- [8] C.G. Gross, D.B. Bender, and C.E. Rocha-Miranda, Visual receptive fields of neurons in inferotemporal cortex of the monkey. *Science*, vol 166, pp. 1303–1306, 1969.
- [9] R.Q. Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried, Invariant visual representation by single neurons in the human brain. *Nature*, vol 435, pp. 1102–1107, 2005.
- [10] S. Kak and M.C. Stinson, A bicameral neural network where information can be indexed. *Electronics Letters*, vol. 25, pp. 203-205, 1989.
- [11] “Scientists discover way to reverse loss of memory”, <http://www.independent.co.uk/news/science/scientists-discover-way-to-reverse-loss-of-memory-775586.html>, The Independent, 2008.
- [12] S. Kak, Single neuron memories and the network’s proximity matrix. 2009. [arXiv:0906.0798](https://arxiv.org/abs/0906.0798)
- [13] S. Kak, Feedback neural networks: new characteristics and a generalization. *Circuits, Systems, and Signal Processing*, vol. 12, pp. 263-278, 1993
- [14] D. Prados and S. Kak, Non-binary neural networks. *Lecture Notes in Computing and Control*, vol. 130, pp. 97-104, 1989.
- [15] M.C. Stinson and S. Kak, Bicameral neural computing. *Lecture Notes in Computing and Control*, vol. 130, pp. 85-96, 1989.

[16] W. Penfield, P. Perot, The brain's record of auditory and visual experience."  
*Brain*, vol 86, 1963, pages 595-696.

## VITA

Krishna Chaithanya Lingashetty

Candidate for the Degree of

Master of Science

Thesis: THE ACTIVE SITES IMPLEMENTATION FOR THE B-MATRIX NEURAL NETWORK

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, 2010.

Completed the requirements for the Bachelor of Engineering in Computer Science and Engineering at Osmania University, Hyderabad, India in 2008.

Experience:

*Research Assistant* Oklahoma State University, Stillwater, OK, January 2009 to present

- Prediction of Time Series (Jan 2009 - present)

Study and development of a model for the prediction of time series data using generic trends.

- Cryptography/Neural Networks (May 2009 - present)

Study and development of a Hebbian neural network model for distributed data/key storage over insecure networks.

*Intern Software Developer* Progress Software Development, Hyderabad, India. August 2007 – May 2008

Sonic-Debug – Development of a Web-Application using GWT and JAVA Servlets for debugging a Sonic Process online.

*Intern Software Developer* Progress Software Development, Hyderabad, India.

May 2007 – July 2007

Sonic-Doc Plugin – Development of a ‘Plugin’ for the Sonic Workbench using Java and XML, which automatically creates documentation for the processes created by clients.

Professional Memberships: Computer Society of India

Name: Krishna Chaithanya Lingashetty

Date of Degree: July, 2010

Institution: Oklahoma State University

Location: Stillwater, Oklahoma

Title of Study: THE ACTIVE SITES IMPLEMENTATION FOR THE B-MATRIX  
NEURAL NETWORK

Pages in Study: 42

Candidate for the Degree of Master of Science

Major Field: Computer Science

Scope and Method of Study:

This thesis is concerned with the problem of memory recall for the feedback neural network called the B-Matrix network. A new model -- the Active Sites model -- is proposed to reduce the computational complexity of the B-Matrix approach. We also develop a new delta learning rule that increases the memory retrieval capacity of the Active Sites model. These techniques are extended to a multi-level, non-binary neural network, which has a much higher capacity than the binary network.

Findings and Conclusions :

Through simulations and analysis, it is demonstrated that to retrieve a memory from a neural network, one does not need to have the whole memory and, as in real life, a fragment of that memory may be sufficient to recall it.

ADVISER'S APPROVAL: Dr. Subhash Kak

---