

SP2172 INVESTIGATING SCIENCE AY 2004/2005

# Artificial Learning

*by*

Abel Yang (U031876N)

Hu Yi (U032048J)

Wee Wei Wen (U032150L)

*Staff Mentor:* Dr. Ng Gee Wah

*Student Mentor:* Shruti Kapoor

November 5, 2004



## **Abstract**

Most artificial neural networks today are designed specifically for particular tasks rather than simulating general brain function. In this project, we attempt to create a novel artificial neural network to simulate the role of memory formation. It carries out unsupervised learning by following Hebb's rules and incorporates a number of features that are biologically inspired. This project investigates how our network should be designed and fine-tuned for it to be a good simulation of memory formation.

# **1 Biological Features to be Simulated**

## **1.1 Introduction**

Relatively little is known to science about the human brain and how it functions. As such, it is difficult to construct a precise model for the entire human brain or even merely a part of it. Our aim is to create a neural network model to simulate a biological neural network and we have chosen to focus on the human hippocampus. However, in light of the current limitations in this area of study, we have selected only a few of the more important and better documented features of the human hippocampus to incorporate into our neural network model.

## **1.2 Random Partial Connectivity**

The Vertebrate hippocampus is a partially connected network[1]. In a partially connected neural network, not every neuron in the network will be connected to every other neuron in the same network. There are a number of ways that a partially connected network can be established. As such, our network is modeled as being randomly connected with an average

probability of connection that a given neuron may be connected to another neuron. In this way, we approximate our network to a random section of a larger biological neural net.

These two features are implemented in our model such that each neuron has a random chance of being connected to another neuron. This is governed by the *connectivity parameter*, which will be explained in Section 2.1.

### 1.3 Hebbian Learning

Hebbian learning is a theory first proposed by Donald Hebb in 1949[2]. This theory specifies how the connection strength between two neurons would be altered depending on how they were firing at any point in time. Connections between neurons that are active synchronously are strengthened, and connections between neurons that are not active synchronously are weakened.

Hebbian learning has been confirmed to occur in several biological systems from the rat hippocampus[3], to certain marine invertebrates[4]. As such, we have used it as our neural networks learning process. The exact process used is actually a variation of the covariance Hebbian model where the difference of the presynaptic and postsynaptic signals is considered with a time-dependent value.

This has thus far covered all the biological features we have implemented in our neural network. The exact workings of our Neural Network will be detailed in Section 2.

## 2 Model Architecture

The network used in this design closely resembles a *Hopfield network*[5]. A Hopfield network is a feedback network that will, when given enough time to run, settle to one of a set of

steady states depending on the initial state. Such a network is an *associative network*, as it associates a certain final state with an initial state. In our context, the inputs provide the initial state while the outputs are a representative sample of the final state.

In terms of the learning process, we wish to model the following distinct features of the biological neural network.

## 2.1 Random Partial Connection Model

Previous studies have shown that most neural networks in young animals are partially connected. In our project, we also assume that connections on the individual neuron level are formed at random via the random growth of the synaptic dendrite. To model such a feature, we set a connectivity parameter for our neural network. This parameter is the probability for any two given neurons to be connected during the network initialization stage.

## 2.2 Discrete Time Approximation Model

A biological network functions according to a continuous time model. In such networks, stimuli may arrive at any point of time, and responses may be fired at any point of time.

Due to constraints of technology at our disposal, a continuous time model was not possible and we use a discrete time approximation of the continuous time model. In every timestep, there can be a maximum of one firing in each synapse. Two neurons activated in the same timestep are considered to be activated simultaneously.

As our artificial neural network learns according to Hebb's rule (Section 2.3), a discrete time model also facilitates the simulation of signal decay. In every timestep, after a signal is

transmitted across a synapse, the signal value is multiplied by the neuron decay parameter before it is added to the value of the next neuron.

### 2.3 Hebbian Learning Model

Our neural network learns according to Hebb's rule[6], which states that

1. *If two neurons on either side of a synapse (connection) are activated synchronously, then the strength of that synapse is selectively increased.*
2. *If two neurons on either side of a synapse (connection) are activated asynchronously, then that synapse is selectively weakened or eliminated.*<sup>1</sup>

Following Hebb's rule, the synapse weight will increase if there exists a highly positive correlation between the presynaptic and postsynaptic activities; otherwise the synapse weight will decrease.

We can formulate this model in mathematical terms as follows. We consider two neurons,  $i$  and  $j$ , which are connected by a synapse whose weight is denoted by  $w_{ij}$ . The direction of the signal is from  $i$  to  $j$ . In any particular timestep  $n$ , let

$x_i(n)$  = presynaptic signal at timestep  $n$

$y_j(n)$  = postsynaptic signal at timestep  $n$

$\Delta\omega_{ij}(n)$  = change in the synapse weight due to  $x_i(n)$  and  $y_j(n)$

---

<sup>1</sup>Part 2 was not included in the original Hebb's rule. It was added by Changeux & Danchin in 1976 and widely accepted now.

Hence at timestep  $n$ , the adjustment  $\Delta\omega_{ij}(n)$  made to the synaptic weight can be written as a function  $F$  of both  $x_i(n)$  and  $y_j(n)$

$$\Delta\omega_{ij}(n) = F(x_i(n), y_j(n))$$

Evidently, the function  $F$  must describe both the correlation feature as well as the *rate of learning*. In our network, we follow the *covariance hypothesis* introduced by Sejnowski[7], in which the weight adjustment is give by

$$\Delta\omega_{ij}(n) = \eta(x_i(n) - \bar{x})(y_j(n) - \bar{y})$$

where  $\eta$  is the *learning rate parameter* and  $\bar{x}, \bar{y}$  denote the *time-avareged values* of all the previous  $x_i$  and  $y_j$  values respectively.

Note that  $\bar{x}$  and  $\bar{y}$  are not simply the average of all previous  $x_i$  and  $y_j$  values. Those  $x_i$  and  $y_j$  values at earlier time steps should have less effect on the weight adjustment in the current timestep. In our neural network, we define  $\bar{x}$  and  $\bar{y}$  as

$$\bar{x} = \sum_{k=1}^{n-1} 2^{k-n} x_i(k) \quad \bar{y} = \sum_{k=1}^{n-1} 2^{k-n} y_j(k)$$

Using this formula for  $\bar{x}$ , after each timestep, the contribution of every previous  $x_i$  value towards the new  $\Delta\omega_{ij}$  is halved. Therefore  $\bar{x}$  and  $\bar{y}$  are reasonably good measures for the strengths of the presynaptic and postsynaptic signals. Consequently, we can observe that our algorithm follows Hebb's rule nicely:

1. The synaptic weight is enhanced ( $\Delta\omega_{ij}(n) > 0$ ) when the presynaptic and postsynaptic signals are both sufficiently high ( $x_i(n) > \bar{x}$  and  $y_j(n) > \bar{y}$ )
2. The synaptic weight is depressed ( $\Delta\omega_{ij}(n) < 0$ ) when

- The presynaptic signal is sufficiently strong while the postsynaptic signal is not  
( $x_i(n) > \bar{x}$  and  $y_j(n) < \bar{y}$ )
- The postsynaptic signal is sufficiently strong while the presynaptic signal is not  
( $x_i(n) < \bar{x}$  and  $y_j(n) > \bar{y}$ )

### Why not back-propagation model

Besides the Hebbian model, we also considered the back-propagation model, which is probably the single most popular algorithm for training neural networks in engineering applications. However, back-propagation's success as an engineering tool does not imply that it is a good model for simulating the biological learning mechanism. It requires transmission of error information back to the nodes that the stimuli were first input from. Since complex biological networks lack the high symmetry present in back-propagation network, it is very unlikely for such precise feedback mechanisms to exist. Therefore, Hebbian learning provides a more feasible model for simulating biological learning.

### Summary of the models of learning

- Partial random connectivity

Each neuron has a random chance of being connected to another neuron. The probability for a connection to be established between any two neurons is the same.

- Hebbian learning with time-based decay

In each timestep, the strength of every signal is multiplied by the decay parameter. Synapses weights are updated through Hebbian learning where the presynaptic weights are the synapse values, and the postsynaptic weights the current neuron values before

decay.

- Discrete time approximation of a continuous time model

To better simulate the continuous time model, we approximate the model with a discrete time model. In each timestep, stimuli are accumulated; neuron values are summed; activation then takes place.

- The model is dependent on the following 6 parameters.

**Network size** is the total number of neurons in the network.

**Connectivity** is the probability for any two neurons to be connected during initialization. A larger connectivity parameter implies that the network is more densely connected.

**Learning rate** is the coefficient  $\eta$  as described in Section 2.3. The synapse weight adjustment is directly proportional to this parameter.

**Activation threshold** is the threshold value that a neuron's value must be above before it fires in the next timestep.

**Neuron decay** is a decay factor multiplied to the neuron's value at the end of each timestep. The larger this parameter is faster the neural value decays.

**Holding time** is the number of timesteps during which the signals are held during every training run. This is to simulate the situations in which input signals last for more than one timestep.

## 3 Experiments and Analysis

### 3.1 Objectives

To show that learning does take place in this network, we consider the following:

- When learning takes place, we have some connections strengthening, and others weakening, resulting in less of the connection weights in the middle.

In terms of an information theoretic perspective, we want to observe a change in the distribution of synapse weights, or in the case of the state of the network, the distribution of neuron values.

In addition, the network should ideally behave with the following characteristics:

- The pattern at the outputs should be reproducible with subsequent stimuli of a similar pattern.
- The resulting activation should result in a steady state that has part of the network active.

### 3.2 Experiment 1

#### Procedure

We model a network with the following parameters:

- network size = 100
- learning rate = 10.0
- threshold = 0.8

- range of holding time H: 1 to 2
- range of connectivity C: 0.01 to 0.025
- range of decay: 0.01 to 0.07

Activation is performed on 5 neurons chosen at random while the outputs are taken from 10 neurons, chosen at random. During the process, learning may be deactivated such that the synapse weights do not change at all during the run.

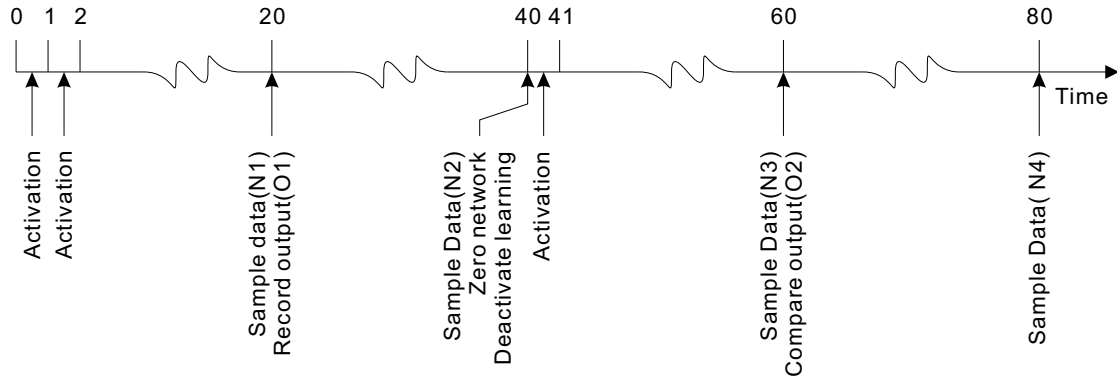


Figure 1: Training procedure for varying Holding time, Decay and Connectivity

After the initial activation, the network is run for 20 timesteps before a sample is taken. We measure the number of firing neurons, number of active outputs, and record a sample of the output pattern. The network is allowed to run for another 20 timesteps, and another sample is taken. The network is then reset, and activated again, with the learning turned off such that the network behaves like a non-learning associator. We run the network again for 20 timesteps and take a sample. Here, the network output pattern is sampled again and we record the hamming distance from the sample taken earlier. The network is then run for another 20 timesteps before the last sample is taken.

For each set of parameters, 20 runs are performed. The average, minimum and maximum of the values are taken. To show that the network learns, we consider the distribution of the synapse weights at our sample points.

### 3.3 Experiment 2

#### Procedure

To observe the learning taking place, we consider the activation values and synapse weights during each timestep for such a network. From our observations from experiment 1, the parameters for the network are chosen as follows: 100 neurons, connectivity 0.015, learning rate 10, threshold 0.8 and decay 0.05.

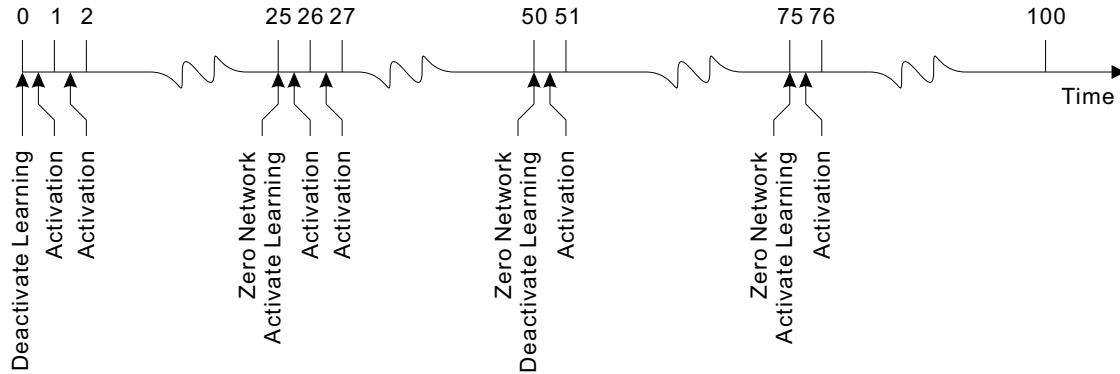


Figure 2: Training procedure for qualitative analysis

We first activate the network with learning turned off, and hold the activation for 2 timesteps. The network is then run for 25 timesteps. The network is zeroed, learning is then turned on and the same procedure is repeated. Then the network is zeroed and learning is then turned off again, and the network is activated and the signal is held for 1 timestep. After 25 timesteps, the network is zeroed again, learning turned on and the process is repeated.

## 3.4 Results

### 3.4.1 Experiment 1 - Varying parameters C, D, H

For both Hold Time = 1 and Hold Time = 2, We first consider the active neuron counts with respect to connectivity and decay. We note that a lower connectivity and higher decay gives us a larger reduction in activation values between learning and recall, as shown in Fig 3 and 4. This indicates that learning has taken place as the stimulus. Likewise, we also record a larger Hamming distance between the outputs during learning and recall when connectivity is low or when decay is high (Fig 5).

In terms of reproducibility however, we note that the range of observed activation values increase with respect to the decay rate, and decrease with respect to the connectivity. This indicates a wide range of resulting values for a low connectivity and/or high decay. By observing the raw data, we note that there is a larger range of observed values for low connectivity. This could be due to the random chance of connection as the mean number of incoming connections per neuron is as low as 1.

Comparing the results from varying the holding time, we note that a longer holding time (2 v. 1) will result in a smaller range of values for the activation patterns as well as a larger difference between learning and recall modes. This is possibly due to the repeated firing of the inputs which allow for a second activation, thus allowing simultaneous activation to take place even when the signal decays very fast.

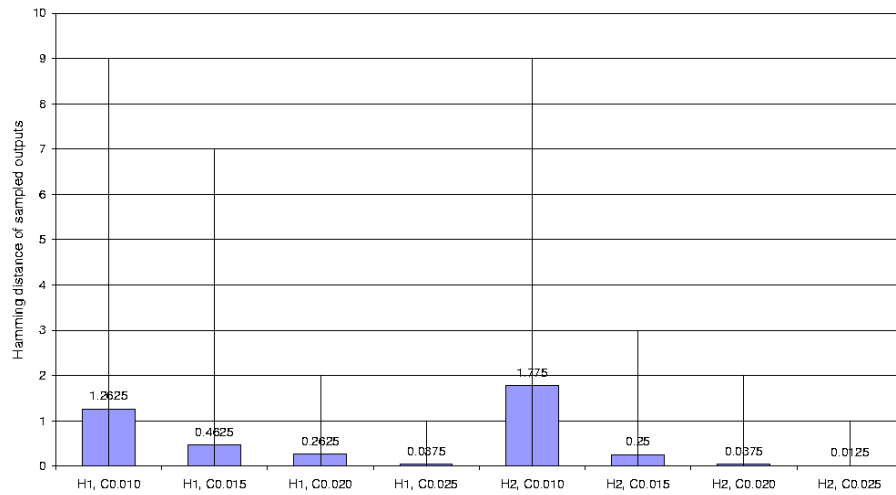


Figure 3a: Hamming Distance against Connectivity and Holding time, 100 Neurons, Learning rate 10.0, Threshold 0.8, Average of values for decay from 0.01 to 0.07

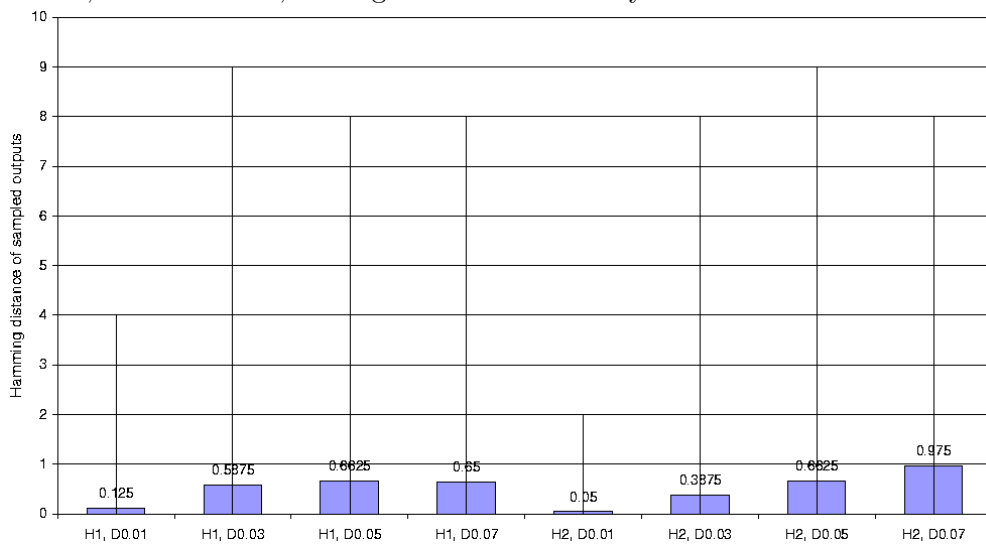


Figure 3b: Hamming Distance against Decay and Holding time, 100 Neurons, Learning rate 10.0, Threshold 0.8, Average of values for connectivity from 0.1 to 0.25

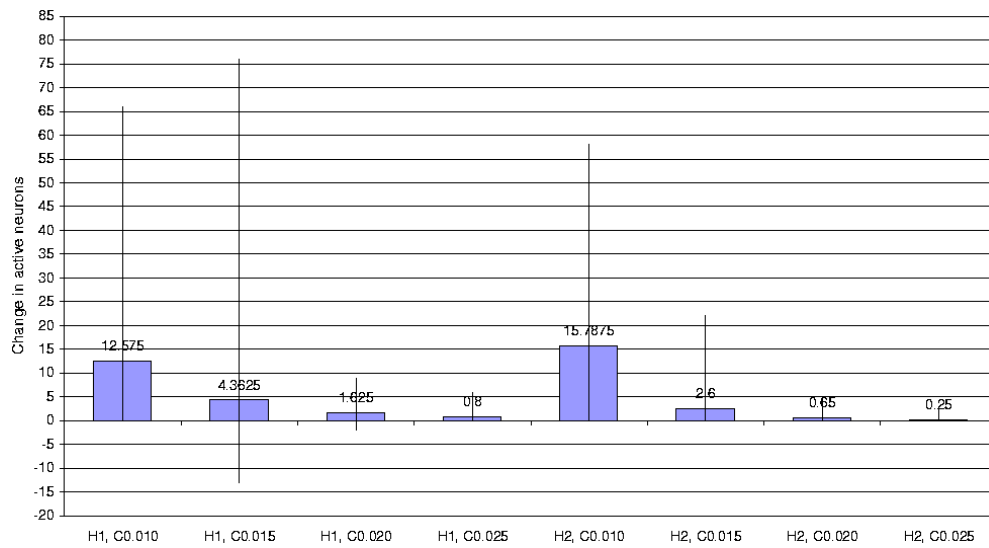


Figure 4a: Change in number of active neurons against Connectivity and Holding Time, Comparison at 20 timesteps after activation, 100 Neurons, Learning rate 10.0, Threshold 0.8, Average of values for decay from 0.01 to 0.07

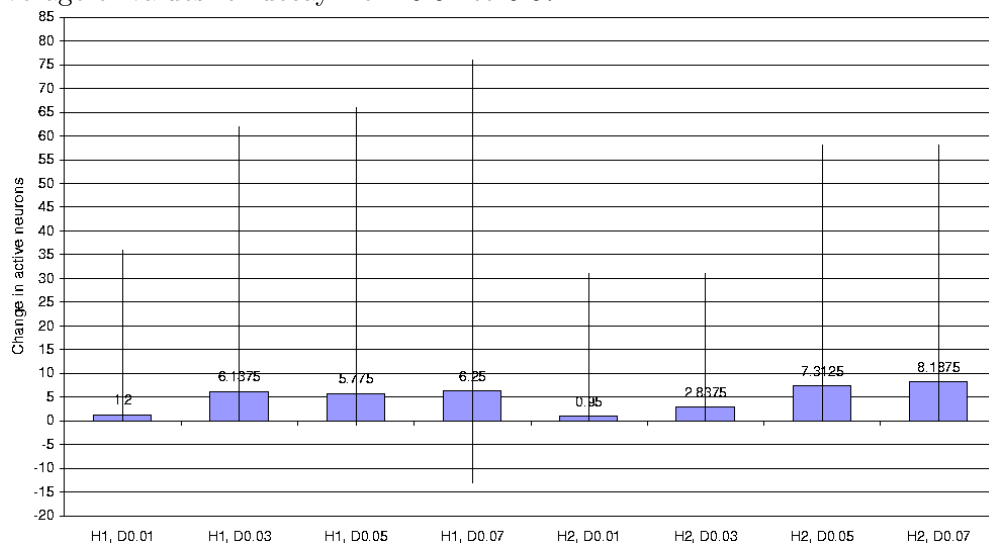


Figure 4b: Change in number of active neurons against Decay and Holding Time, Comparison at 20 timesteps after activation, 100 Neurons, Learning rate 10.0, Threshold 0.8, Average of values for connectivity from 0.1 to 0.25

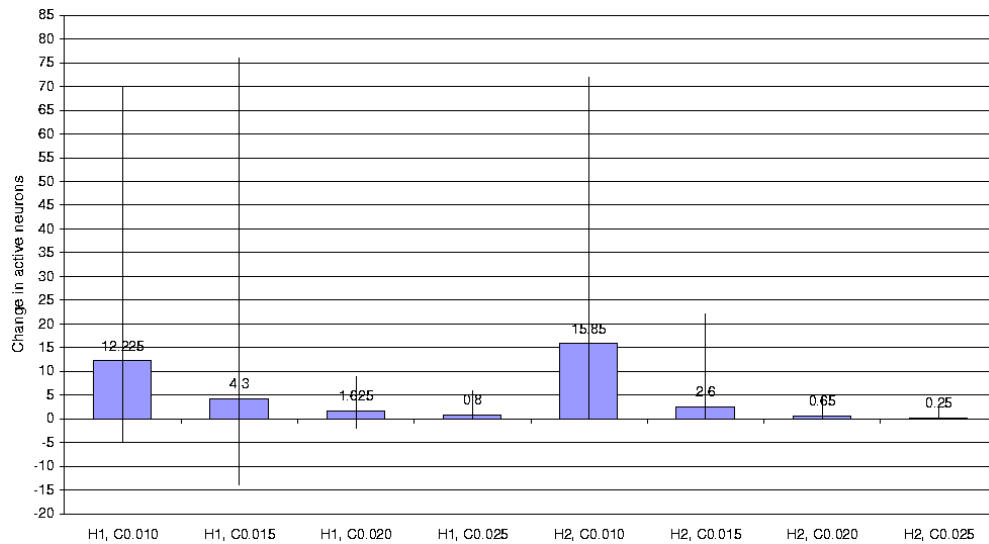


Figure 5a: Change in number of active neurons against Connectivity and Holding Time, Comparison at 40 timesteps after activation, 100 Neurons, Learning rate 10.0, Threshold 0.8, Average of values for decay from 0.01 to 0.07

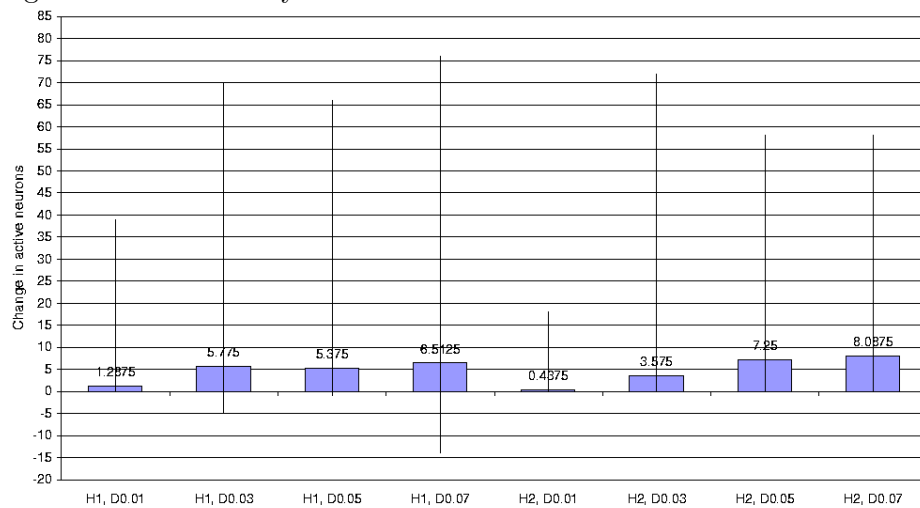


Figure 5b: Change in number of active neurons against Decay and Holding Time, Comparison at 40 timesteps after activation, 100 Neurons, Learning rate 10.0, Threshold 0.8, Average of values for connectivity from 0.1 to 0.25

### 3.4.2 Experiment 2 - Analysis of a training run

The data obtained for Fig 6 was taken as the best representative out of 10 runs of which some could not be sufficiently activated, while others did not show a change in the number of active neurons. This is indicative of the random nature of the network, and shows that not all randomly initialized networks are viable. In some cases, we see that the signal quickly dies, while in other cases, we see that the signal spreads and activates a large number of neurons very quickly.

Observing the activation values against time taken in Fig 9, we note that during the first phase, the network quickly settles to a steady state where there is no change in the number of active neurons, quickly, indicating that due to the random connectivity, there is a distribution in the chance a neuron will be activated, hence indicating that the network already has some amount of encoded information. In the second phase, we note that the number of active neurons is approximately the same as the previous phase, however, we note that the time taken to settle to the steady state is slightly longer, and there is more fluctuation prior to the steady state. This indicates that there is a longer propagation delay, presumably caused by the change in connection weights.

During the third phase, we see a shorter recall stimuli being applied to the network with the learning turned off. In this case, we see a much longer period of fluctuation, indicating that there is a longer propagation delay and therefore that the connection weights are already differentiated.

In the fourth phase, we turn on the learning and repeat the process. We see that there is an even longer period of fluctuation, followed by a larger number of unactivated neurons. This shows the reinforcement caused by a subsequent signal.

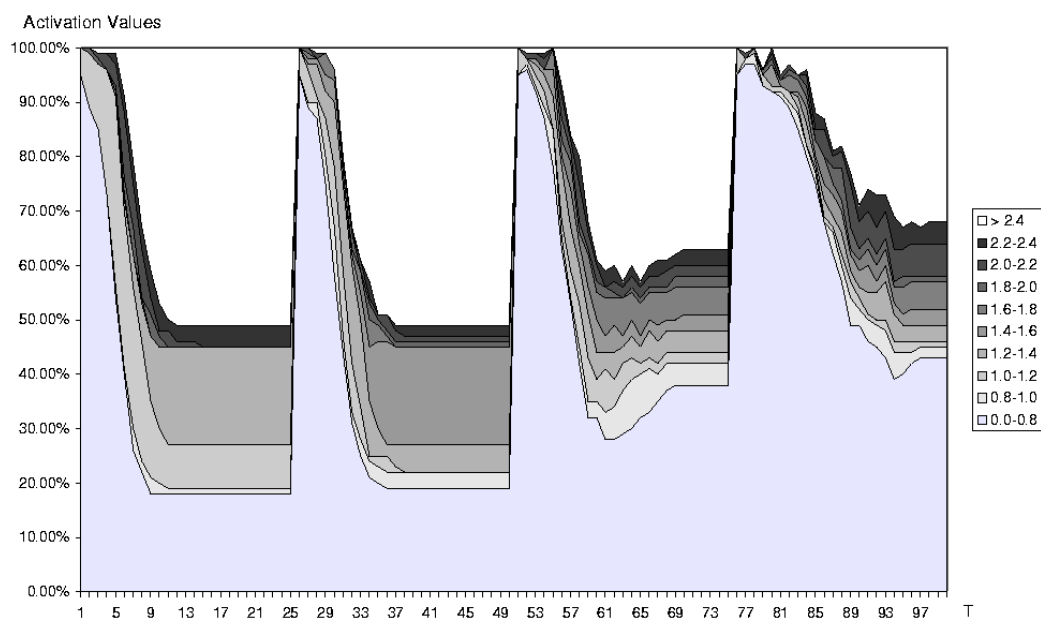


Figure 6: Proportion of neuron values with respect to time, 100 Neurons, Learning rate 10.0, Threshold 0.8, Decay 0.05, Connectivity 0.015.

## 4 Discussion

For an associative network to properly function, it has to associate a stimulus pattern with an activation pattern. For this to occur, the signal must propagate through the network such that it activates enough neurons to reinforce itself and hence form the activation pattern. This self-reinforcement can be obtained by a structure within the network that forms a loop. A loop for this purpose is a set of edges that when traversed allow a node to be connected to itself (Fig 7). Of relevance here will be the length of the loop, which affects the propagation delay and number of available nodes for the signal to receive further stimuli.

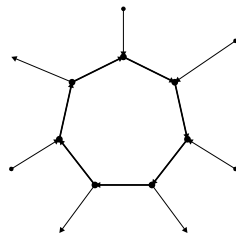


Figure 7: A Loop in a network

#### 4.1 Factors that affect learning

From the results, we see that unsupervised learning does take place with a Hebbian system, however the learning is not always consistent. Due to the random connectivity of the network, the loops formed may not be suitable for learning and may cause problems in the following ways:

- Excessive Feedback

This occurs when neurons are connected in a loop that is too short, allowing the signal less time to decay before it arrives back at its originator. While this presents an opportunity for the network to quickly form a strengthened loop, such feedback could also spread to other parts of the network before learning can take place, preventing the irrelevant portions of the network from being depressed. This results in the network being over-stimulated too early such that learning does not take place from subsequent stimuli. In terms of parameters, reducing connectivity or decay factor may solve this problem.

- No Loops accessible

When there are no connections that allow a signal to loop back to its originator neurons, the signal will not loop back within the network, and therefore learning via

simultaneous activation will not take place so strongly.

Therefore, for learning to take place, a loop of an appropriate length is necessary. This implies that there may be some order in the connections between neurons. Loops of unsuitable length can also be a plausible explanation for the large range of observed values when connectivity is low. With reference to a biological system, such results are indicative of the idea that some connections may not be random.

## **4.2 Multiple Stimuli**

As the number of active neurons decrease after the learning process, the network effectively partitions itself to respond to a specific stimuli. When a different stimulus is applied to the network, it is likely to activate a different part of the network. We hypothesize that should a different area of the network be connected and active, the simultaneous activation will strengthen the connections and allow the two stimuli to be associated.

## 5 Conclusion and Further Work

Through this simulation, we show that Hebbian learning may be a viable model for unsupervised learning in a feedback neural network. This however has a number of problems: Parameters with high learning rates are achievable, but the high learning rates are not necessarily repeatable; of the parameters possible, it is likely that there is a small band of parameters that work, while others will not work.

In this study, the learning rate was set at 10.0 and hold time varied between 1 and 2. The learning rate in this case can be said to be rather high, as it is used to artificially cause the network to be sensitive to a short input signal such that learning may take place within a limited number of timesteps, and to allow the holding time to be kept low. Further work can be done to determine the effect of a lower learning rate and longer hold time, and also to examine the possibility of a high and low learning rate by adjusting parameters. This would support a Hebbian learning version of the Hippocampal-cortical model proposed by Gluck and Myers[1], where a fast-learning hippocampus trains a slow-learning cortical network.

Partial recall and multiple stimuli was not tested in this study and may be a subject for future work.

## **Acknowledgement**

We would like to thank our supervisor Dr. Ng Gee Wah for guiding us throughout the span of this project. He has helped us in deciding the general research direction and suggested numerous improvements to our experiments. This project would not have been achievable without his generous guidance.

We are also grateful to our student mentor Shruti, and Gaurav for their many interesting discussions with us. We would like to thank everyone who has helped us in one way or another, too.

## References

- [1] Rolls E. T. and Treves A. A computational analysis of the role of the hippocampus in memory. *Hippocampus*, 4:374–391, 1994.
- [2] Hebb D. O. *The Organization of Behavior*. Wiley, 1949.
- [3] Johnston D. and Magee J. C. A synaptically controlled, associative signal for hebbian plasticity in hippocampal neurons. *Science*, 275:209–213, 1997.
- [4] Kandel E. R. Activity-dependent presynaptic facilitation and hebbian LTP are both required and interact during classical conditioning in Aplysia. *Neuron*, 37:135–147, 2003.
- [5] Hopfield J. J. Neural networks and physical systems with emergent collective computational abilities. *Biophysics*, 79:2554–2558, 1982.
- [6] Haykin S. *Neural Network - A Comprehensive Foundation*. Prentice Hall, 2nd edition, 1999.
- [7] Sejnowski T. S. Statistical constraints on synaptic plasticity. *J. Theoretical Biology*, 69:385–389, 1977.