HCI 585X Project Proposal

March 10, 2010

# Modeling the Neo-Cortex

Nandhini Ramaswamy

&

Christopher Walck

Under the guidance of

Prof. Alexander Stoytchev

Department of Electrical & Computer Engineering,

Iowa State University

# CONTENTS

## Introduction:

The brain is the most complex organ of the body in terms of both its functions and architecture, and when it comes to studying artificial intelligence, many people turn to the brain because it is the only existing system that demonstrates intelligence. The neo-cortex is a thin sheet of neural tissue that envelopes a good portion of the brain. In the evolutionary timeline, it is the newest component to evolve, only being found in the brains of mammals. It is considered to be the center of human intelligence, being responsible for activities such as reading and language, motor control, perception, art, music and so on.

According to Hawkins, the neo-cortex is made up of six layers which he compares to that of the size of six business cards arranged as a stack. According to him, this six layered structure is common to all the organisms, the only difference in all these animals is the size of the brain.

The neo-cortex is comprised of neurons so densely packed that it is difficult to calculate the exact number of neurons present in the layer. It is estimated to contain around thirty billion neurons. These thirty billion neurons constitute the intelligence, memory, skill or other experience we acquire during our life.

There are different functionalities that are performed by different parts of the brain but there is no clear boundary between the different parts of the brain. In Figure 1, Hawkins describes the structure of the brain.

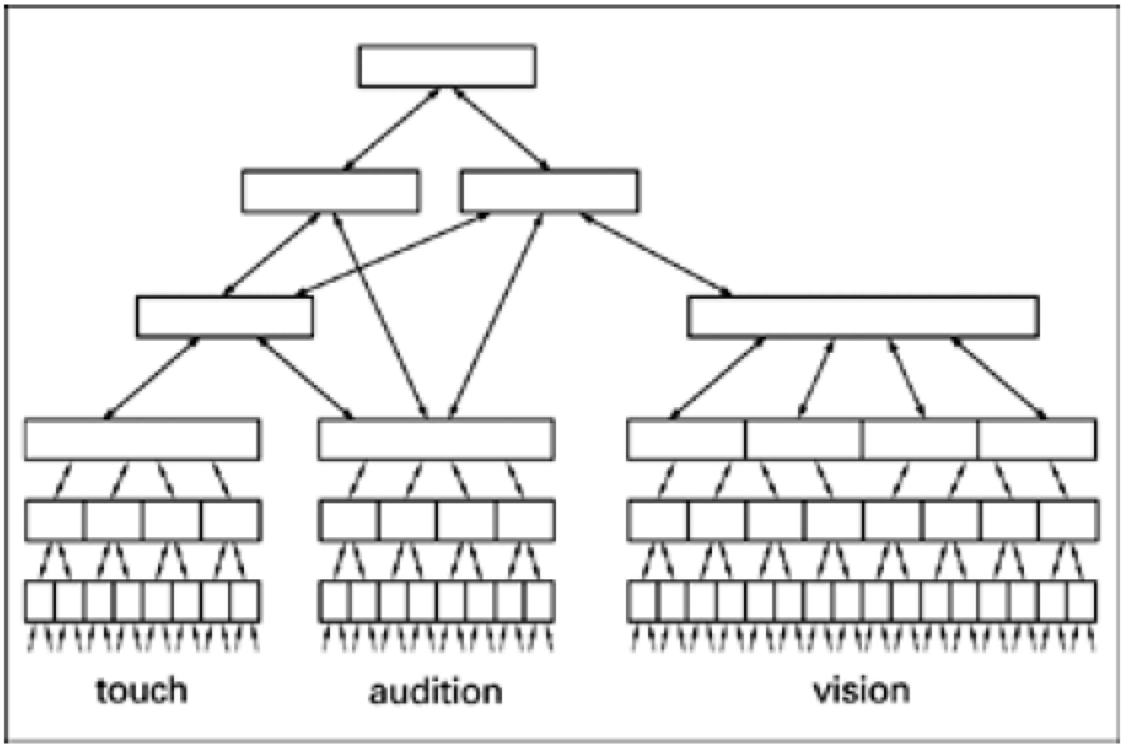


**Figure 1. Hawkins depiction of a neuron hierarchical model**

## Idea:

According to Hawkins's theory, the input from various sensory organs passes through various layers up in the hierarchy to identify a particular object. If the bottom most layer is not able to identify a particular object, then information from various neurons from the bottom layer is passed on to the upper layer and the respective neurons are fired, and this flow of information continues until the object is identified at the top of the network. This theory developed from the observation that within the neo-cortex, we see that the bottom neuron layer is very densely packed, and each layer above is less dense than the last.

At a cellular level, additional processes are existent as information is passed between neurons. These processes occur in reaction to the signals passing through neurons. They are thought to play an important role in how the hierarchal system develops in the neo-cortex. These processes will be better explained later in the proposal.

The aim of this project is to model how the brain organizes itself based on sensory inputs, continue by modeling the behavior of identifying an object through inputs from neurons in the input layer and mapping the connections established in the hidden layers, and finally, the identification of a particular object in the output layer, thereby simulating the functioning of the neo-cortex.

The various layers are explained as follows:

- Input Layer: This layer would serve as the input to the network. The neurons in this layer would represent the basic properties of an object that has to be identified in the output layer. Example of a neuron in the input layer would be representing square, circle, red, blue, green, soft, rigid etc.
- Hidden Layer: This layer is the one that actually does the mapping or the connections between the basic properties represented in the input layer to the actual object in the output layer.
- Output Layer: This is the layer which represents the output produced by the network for the given sets of input.

## Need For This:

We believe that this project will help in understanding how to identify an object say apple as "apple" and how this could be integrated into robots for identifying various objects and further help in categorizing them by taking visual, auditory or other sensory inputs and giving these as input to this network.

## Previous Approaches and Related Work:

The history of neural networks is very extensive. The idea of an artificial neuron was proposed in the mid 1940's, and recently artificial neural network systems have been rapidly increasing in application and usefulness. Some current and previously explored applications of neural networks include:

- Speech recognition. Neural networks are widely used in speech recognition. They have also been used to in lip-reading by tracking lip movements and leaning the associated sounds.

- Robot control. Many experiments and projects have implemented the use of neural networks and genetic algorithms to control a robot. Using a neural network, the robot learns its own routine to move, even with more complicated methods like walking and swimming robots.
- Event recognition. One interesting research study was tracking the normal movements of pedestrians to predict future trajectories. This approach could be used to develop automated surveillance without human behavioral knowledge.
- Detecting and tracking moving objects. Tracking statistics are applied to neural networks. They also have been used for face detection in low resolutions, or with unreliable body movements.
- Toys and computer games. Neural networks have been implemented in some computer games and toys with grouping, body control, and graphics. One interesting simulation was a game of soccer where simulated robots used neural networks to learn.

Such networks come in a broad range of formats, which all prove to be useful in specific scenarios. Based on the broadness of the idea and the number of applications that already exist, there is no clear target audience that will benefit most from this project. The list above shows just a few common implementations of neural networks and with the wide variety of uses that such a tool can provide, neural networks can apply to people of all age groups.

### Previous Experience:

The project would be implemented in two phases by the two team members of the project as follows:

The above explained idea would be implemented using

- Neural network algorithms in Java by Nandhini who has experience working with Java, would be trying to implement this network using neural network algorithms.
- Exploring the dynamics of inner brain function by Chris, who has experience working with simulations in Matlab. This approach would focus the learning of a network viewed as a dynamical system.

The two approaches would be explained in detail in the following sections.

## Approach 1: Using Neural Network Algorithms in Java

### Overview:

The major aim of the project is to represent the neurons as the basic unit of operation in various layers as explained above. Refer to the Figure 2 below.

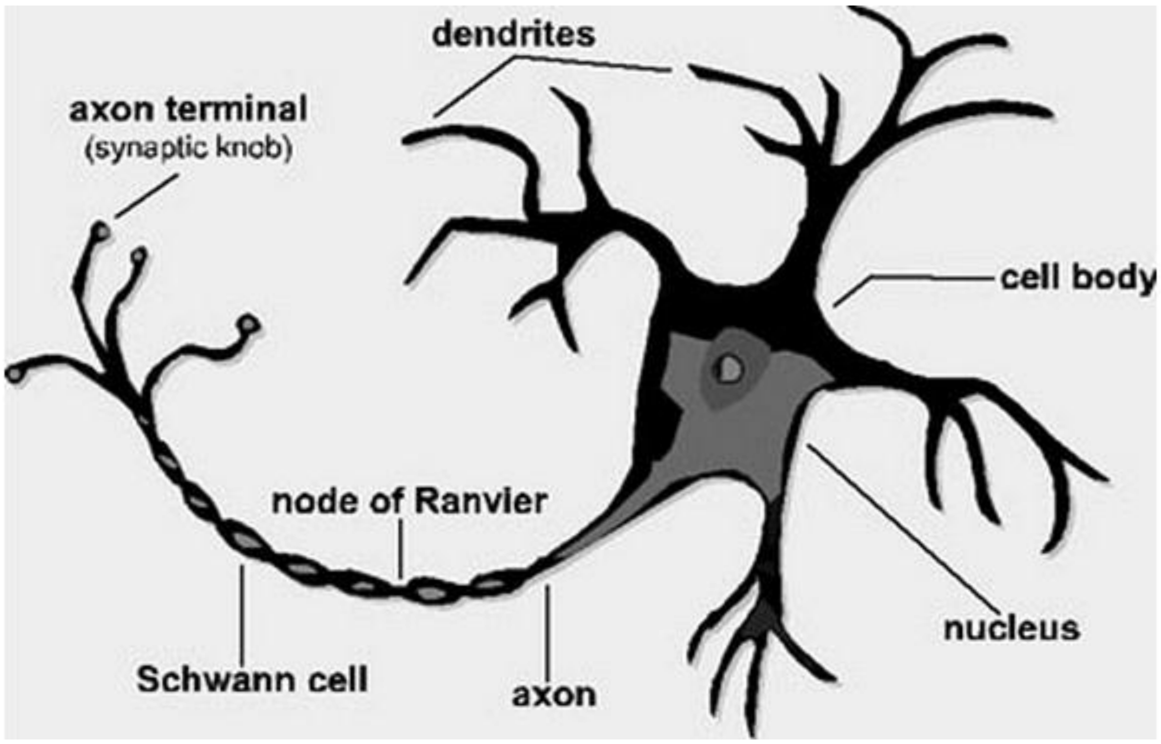


**Figure 2. Diagram of a neuron**

Computers try to simulate these biological neural networks by using artificial neural networks. There is a wide range of difference between the artificial and the biological neurons as the neurons fires by accepting the signals which are analog while most computers process digital signals. As the inputs to the biological neurons are analog, the voltage of each signal varies from each other and the neurons fire when the input signal is above a particular voltage. Hence these neurons make decisions by either firing or not firing. A different number of neurons fire for a particular activity, making decision making a collective activity of a number of neurons.

For the purpose of implementation, there would be no special equipment used other than a computer to run the neural structure.

## Data Structures:

The possible data structures that would be used as a part of the implementation are arrays, decision trees and matrices as matrix mathematics is generally useful for both training and calculating outputs.

A neural network uses the weights and threshold values.

- Weights define the interactions between the neurons.
- Thresholds define what it will take to get a neuron to fire.

Matrices are commonly used to represent the weights and threshold value of a neuron.

## Algorithms:

There are three possible types of training the neural network to make a neural network learn identifying objects in the future.

- Unsupervised learning- The network is provided with training data sets while the anticipated output is not given.
- Supervised learning- The network is provided with training data sets while the anticipated output is given.
- Reinforcement learning- The network is provided with training data sets while the anticipated output is not given as in unsupervised learning. However for each output the network is told whether it is right or wrong.

Deciding on which of the learning types to use is an important factor as it in turn decides the algorithm to be used. While these different techniques are suitable for different problem types its tricky to decide which one to use.

Also various ways of constructing the neural network like the Kohonen neural networks (Also known as Self organizing map-SOP works by making one output neuron win among other neurons resulting in that neuron firing for a particular pattern) would be tried in the initial phase. This includes deciding on the number of neurons in each layer and the number of hidden layers. The approach for this is to start with a random number of neurons proportional to the input size. This would be altered after every iteration.

I would first like to describe the two algorithms that would be suitable for this project.

## Hebb's Rule for Unsupervised Learning:

The Hebb's rule is represented as follows:

$$\Delta w_{ij} = \mu \, a_i \, j_i \, ,$$

where delta is used to calculate the needed change in the connection from neuron i to neuron j and mu is the learning rate.

This rule works by reinforcing what it already knows. Hence this approach is popularly known as "fire together, wire together". This is because, if the two neurons have similar activations, their weight is increased. If two neurons have dissimilar activations, their weight is decreased showing the associations between the connections.

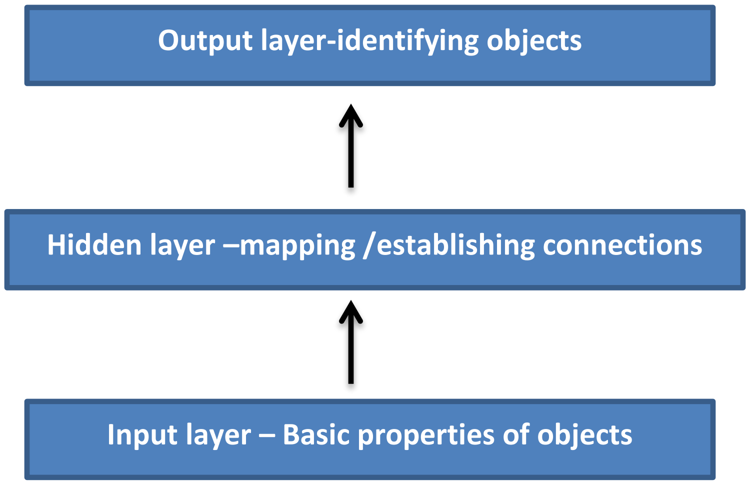
## Feedforward-Backpropagation for Supervised Learning:

In a feed forward network, neurons are only connected foreword. Each layer of the neural network contains connections to the next layer.

Back propagation is a form of supervised training. In supervised training, the network must be provided with both sample inputs and anticipated outputs. The anticipated outputs are compared against the actual outputs for given input. From these anticipated outputs, the back propagation training algorithm calculates the error and adjusts the weights of the various layers backwards from the output layer to the input layer.

As both the algorithms are efficient in various cases, one of the algorithms would be chosen by trial and error method by checking its performance with the trial data.

## Dataflow:


Output layer-identifying objects
Hidden layer –mapping /establishing connections
Input layer – Basic properties of objects

This is the flow of data as in any neural network. The user interface is not of much emphasis in this project, it would be a simple command line interface to give inputs to start the training and testing.

## Software Libraries:

I am planning on using Weka for learning and to train and visualize the results of the data sets in training and testing.

Other libraries in weka for usage in java for creating classifiers and networks would be used. Also, packages provided as a part of Heaton research would be used for implementation. I have used them a little in my AI class.

## Timelines:

March 20:

- Come up with an initial set of objects.
- Decide on the properties to represent these objects.
- Decide the size of training data set.
- Try various algorithms and decide on the most appropriate one.
- Decide on the structure of the neural network which includes deciding the number of neurons in each layer, the number of hidden layers and choosing a type of network.

April 10:

- Implementation with the initial set of data using java & weka.

- Reiterate and increase the number of input samples.
- Decide on size of training & test data.
- Start evaluation for larger data sets.

April 20

- Evaluate the efficiency of the implementation.
- Prepare report
- Poster

## Evaluation:

The evaluation will be based on the number of samples that are correctly identified after the training phase using the test data. The number of samples that would be correctly identified based on the objects characteristics would be measured. Also the connection paths taken in identifying an object would be observed. It is natural to have confusion in identification as it happens with humans to differentiate between two objects that look alike. The success in this experiment would be obtaining correct identification for about 80% of the data that is presented.

Improvements on these results could be achieved by giving input characteristics of better precision or additional details to resolve conflicts or by trying if any improvements could be achieved by changing the number of neurons in various layers in the initial stages of implementation.

## Approach 2: Modeling the dynamics of inner brain function in Matlab

### Overview:

Approach 1 signifies the idea that neurons represent data, and the collective firings of neurons are the essence of how decisions are made.  However, in modeling the functioning of the neo-cortex, it is also important to study the process of neuron grouping, and how such a network can arrange itself to make these decisions based on external stimuli.  Therefore, this approach will focus on the behavior of an artificial neural network system as it responds to its input information.  Furthermore, the overall idea of developmental robotics suggests that intelligence can only be acquired after undergoing a stage of development, where interactions with the environment are essential.  In a crude sense, this approach will attempt to relate the process of neuron grouping within an artificial neural network to aforementioned stage of development.

There is still much uncertainty in our understanding of how the connections of neurons within the neo-cortex arrange to produce intelligent behavior.  Below are two different approaches.

- Hebb's rule.  This is the idea of "neurons that fire together, wire together."  This idea is explained in depth in approach 1.
- Axon guidance.  This is the study of how neurons send out axons to reach other target neurons. The set of mechanisms behind axon path-finding is a science in itself, and an in-depth explanation would defeat the purposes of this paper.

When it comes to modeling the neo-cortex, one or both of the above processes will be used because they address how neurons interact between each layer of the neo-cortex. These processes may attribute to the grouping that allows for intelligence to develop in the brain.

## Data Structures:

The data structures here will closely resemble that of approach 1. However, programming in Matlab will allow for the use of cell-arrays. This is a tool where one can store an array within each array element. It will prove useful because it will make the act of tracking neuron connections from one level to another level easy, when the numbers of neurons in each level are different. The following data will be used in the simulation.

1. Input and resulting output
2. Firing neurons and threshold values
3. Connection pathways and strengths
4. Output uncertainty

Sets 1 and 2 are needed to simulate the flow of signals through the network. Sets 3 and 4 are used to model the process of development of the system.

## Algorithms:

Approach 1 introduces 3 methods of training a neural network. I will run my simulation as an unsupervised neural network because the general methodology best resembles the neuron interactions in the neo-cortex. The algorithm will follow process outlined below

Initial conditions

1. Define the size of the system
    a. How many layers
    b. How many neurons in each layer
    c. Firing threshold of the neurons
2. Define neuron connections
    a. Make random pathways between neurons from one layer to the next
    b. Initially, make every pathway the same strength
3. Define input sets

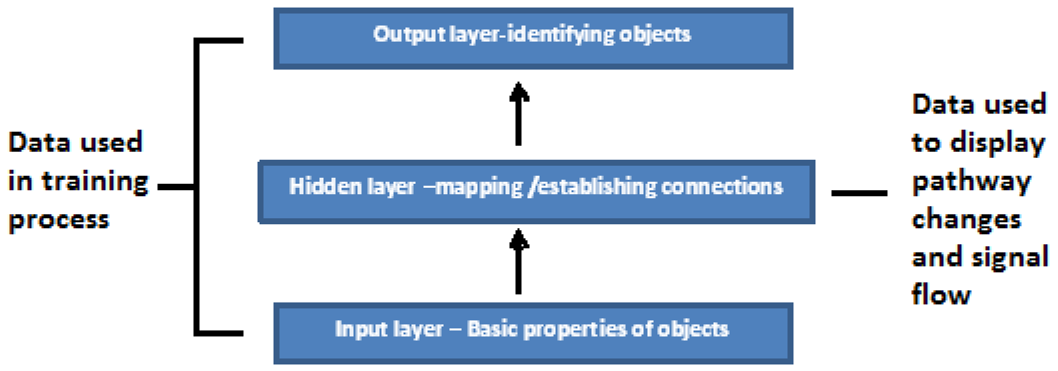
Learning process

1. Randomly select one of the input sets
2. Apply the input data to the bottom layer, and calculate the signal paths to the output layer
3. Re-evaluate the strengths of each pathway as described by Hebbian theory.
4. Store the following for data analysis.
    a. The top layer results
    b. The changes in pathway strengths
    c. The neurons fired
5. Repeat the process many times.

The stored data would be used to create multiple graphical displays of the development of the neural network. These displays would show various properties of signal flow through the network.

## Dataflow:

The flow of data will again, closely resemble that of approach 1, however data will be separated into two sets as shown below.


Output layer-identifying objects
Hidden layer –mapping /establishing connections
Input layer – Basic properties of objects
Data used in training process
Data used to display pathway changes and signal flow

As the above figure shows that the simulation will process the input/output data, and the signal and pathway dynamics.

## Software Libraries:

I will use Matlab to create my simulation. The complete simulation will be done in a regular script. No libraries will be needed other than general operations basic to computation. Matlab has an excellent set of user interface tools that will be used to display data. Using this, I will be able to show the progression of the model by creating video clips of the developing connections.

## Timelines:

March 20:

- Decide on the structure of the neural network which includes deciding the number of neurons in each layer, and the number of hidden layers.
- Write neural network algorithm. Work on code that displays the results.
- Evaluate the effectiveness of the method, and explore ways to make the model more realistic.

April 10:

- Finalize data.
- Evaluate the data, compare it to our goals

April 20

- Prepare report
- Make the poster

## Evaluation:

In evaluating the data, we will look for the following characteristics.

1. Running various inputs into the system will cause changes in the neural pathways.
2. After repeating the process several times, we will see the changes of neural pathways approach some transient or minimum value. This will indicate that the system has found a way to group the characteristics of each input.
3. Each input will lead to a different output neuron firing output on the top layer.

A general success of the simulation will be achieved if the system displays the three characteristics above. If 1 and 2 are observed, we will have demonstrated that our model is interacting with the passing signals. Approaching transience in (characteristic two) may indicate that the algorithm has found some way to relate data from different sets of inputs. If we observe characteristic three, we will have confirmed the process of neurons self-arranging based on input stimulus.

If the model proves to be successful in every way mentioned above, the algorithms data can be used in the following ways.

- Compare the output uncertainty to the behavior of the systems connections as it learns.
- Test the network with the sets of objects and properties used in approach 1.
- Test the accuracy of grouping at different points in the algorithms learning.

If the algorithm is successful enough to perform the above additional tests, they will provide additional information about the learning processes of the artificial neural network. This is important because we could possibly compare and contrast the results with an actual stage of development observed in the desired in developmental robotics.

## References:

[1] Introduction to Neural Networks with Java, Jeff Heaton, Heaton Research, Inc.; 2 edition (October 1, 2008)

[2] Brian, A Journal Of Neurology, Oxford Journal

[3] An Introduction to Neural Networks, James A. Anderson, The MIT Press (March 16, 1995)

[4] Hawkins, Jeff. On Intelligence. Times books, sep. 9, 2004

[5] Neural Network Applications, Japan Singapore AI Centre. Service 22 Dec. 2006, Web 4 Mar. 2011. http://tralvex.com/pub/nap/

[6] Brain Basics, Centre for Synaptic Plasticity, University of Bristol. Service 30 Mar. 2010, Web 4 Mar. 2011, http://www.bris.ac.uk/synaptic/basics/

[7] Gilat, Amos. *Matlab, An Introduction With Applications 2nd ed.*, John Wiley and Sons Inc.