The Effects of a Learning Algorithm on the Hopfield Network

Davon Webb

ABSTRACT

Modeling the human brain has become an increasing interest in many fields of science. One way to model a brain is to use an Ising model where the spins represent the brain's neurons. An effective neural network is able to store, recall and learn new things. Using a Hopfield network, I was able to study the behavior of a simple learning algorithm.

Background

In 1943 Warren S. McCulloch and Walter Pitts proposed that a network of simple neurons was capable of universal computation. This means that the network could perform any calculations that can be done with the most basic computer imaginable. This discovery stimulated the interest of many scientist who were interested in simulating the brain. John Hopfield's work became a particular interest to physicists. The network model that he studied, and the one that is used in this project, has been dubbed the Hopfield network. The Hopfield network distinguishes itself from other types of neural network models by being fully connected, meaning that every neuron is connected to every other neuron, while other models have a layered structure. When connections are chosen to be symmetric then the model resembles a spin model for which statistical mechanics can be applied.

Ising model is made up of simple units called spins that are connected together through interaction energy, J. The Ising model was first conceived in the 1920s by Wilhelm Lenz, who suggested it as a topic for his graduate student, Ernst Ising.

Neural networks are used by a number of different fields in order to gain insight on how the brain learns and processes memory. Researchers who study artificial intelligence use neural networks because they generalize the aspects of memory and learning, which are difficult to capture with normal computational approaches. Neural networks can be used in image processing, and speech and handwriting recognition.

Methods

Since we used an Ising spin to model a neuron then there are only two possible states for our neurons to be in, $s = \pm 1$. +1 corresponds to a state of firing, while -1 corresponds to not firing. We decided to make every neuron in our network connected to every other neuron, so the total energy of the system is written as

$$E = -\Sigma J_{i,j} s_i s_j \tag{1}$$

where s_i and s_j are the state of the two neurons in the observed pair and $J_{i,j}$ is the strength of the synapse connection between them. This way the sum of the synaptic inputs on neuron i is $\Sigma_j J_{i.j} s_j$. This was used to determine the firing rate for neuron i. If the synaptic input is negative then it will be preferred that neuron i is not firing or $s_i = -1$. If it is positive then $s_i = +1$ is preferred.

Since the purpose of our neural network is to study memory, we specified a particular configuration of our network as a stored memory. The memory is a predesignated pattern that we want our neural network to converge to. This will simulate remembering. The stored memory is related to the strength of the synapse connection by

$$J_{i,j} = s_i(m)s_j(m) \tag{2}$$

where m is the stored pattern. Combining (1) and (2) allowed us to change our total energy equation to

$$E = -\Sigma s_i(m)s_j(m)s_is_j. \tag{3}$$

The equation shows that a having a network that matches pattern m, $s_i(m) = s_i$ and $s_j(m) = s_j$, the energy will be large and negative. This gave us an simple way to recall a memory. We use (1) to calculate a value, ΔE_{flip}, that represented the energy contribution of that pair if the state of neuron i was switched. If ΔE_{flip} is negative then the spin is reversed resulting in a lower total energy. If $\Delta E_{flip} \geq 0$ then the spin is left alone. These conditions allowed our network to converge to the lowest possible energy, which is the configuration of our stored memory.

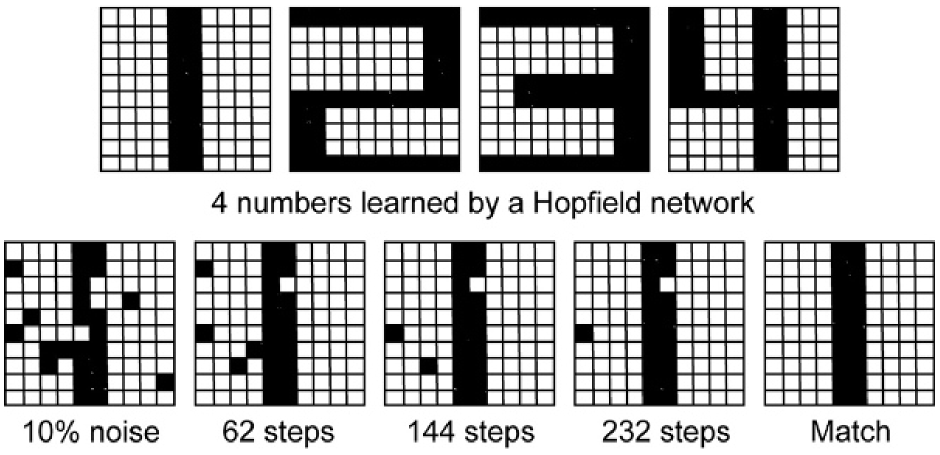

Figure 1. A visual of what is going on when we recall a memory.

In Figure 1 we have four stored images in our memory, the numbers 1,2,3, and 4. The white boxes represent neurons with a spin of +1, while the black boxes are neurons with a spin of -1. The image on the bottom left is our initial random neural network. When it is run through the algorithm that functions as memory recall, the spins get flipped until the network matches the nearest low energy state, which in this case is 1.

Since our goal was to model the human brain another important aspect that we needed to included in our model was learning. We used a simple learning algorithm that changed our interaction energies like so,

$$J_{i,j}(new) = \alpha J_{i,j}(old) + \beta s_i(p)s_j(p), \tag{4}$$

where p is the new pattern, α is the parameter that controls the fading of old memories, and β is the parameter that controls the how fast learning occurs. In order to understand how Eqn.(4) affects how memory is recalled for our neural network, I stored one pattern in the memory manually and then added a

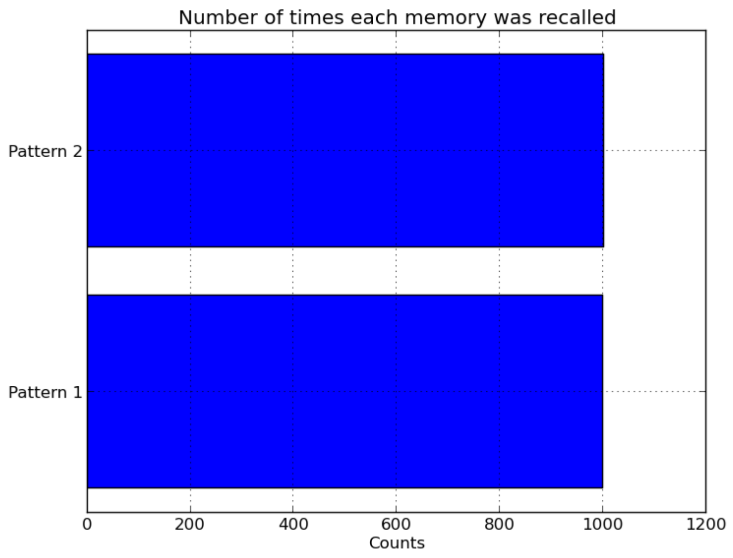

Figure 2. $\alpha = \beta = 0.1$. $N_1 = 999$ and $N_2 = 1001$.

second using the learning algorithm. Then I created 2000 11×11 neural networks with random spins and attempted to recall a memory. The random spins converged to either Pattern 1 or Pattern 2 and we counted how many times each memory was recalled. N_1 is the number of times Pattern 1 was recalled and N_2 is the number of times Pattern 2 was recalled. This was repeated many times for different α and β values.

Results

While writing the code in order to recall memory I ran into a problem that I was unable to fix. As stated before, our random spins were supposed to flip until they converged to the lowest energy, which was only meant to occur when $s_i(m) = s_i$ and $s_j(m) = s_j$. However, in Eqn.(4) you can see that $-s_i(m) = s_i$ and $-s_j(m) = s_j$ is also at the lowest energy since the two negatives cancel. Therefore, for every stored memory there where two states that the spins could converge to, one that was equal to the stored memory and one that was equal to the negative of the stored memory. This didn't have an impact on our overall data since I just specified that the negative of a pattern is also a valid response, but I was unable to prevent it from happening.

When I set α and β equal to 0.1 I found, as you might expect, that both patterns had an equal chances of being recalled (Figure 2). Initially, α was kept constant and β was increased gradually. I realized that the values of α and β themselves did not matter. What was important was the ratio of the two numbers. The input parameters for Figure 1 and Figure 2 are different by a factor of 10, yet they produce very similar results because the ratios are the same.

A change in the ratio of $\frac{\beta}{\alpha}$ produced a logarithmic change in the ratio of $\frac{N_2}{N_1}$. The relationship is shown in Figure 5. The slope of the graph tells us that $\frac{N_2}{N_1} \propto (\frac{\beta}{\alpha})^{2.5}$

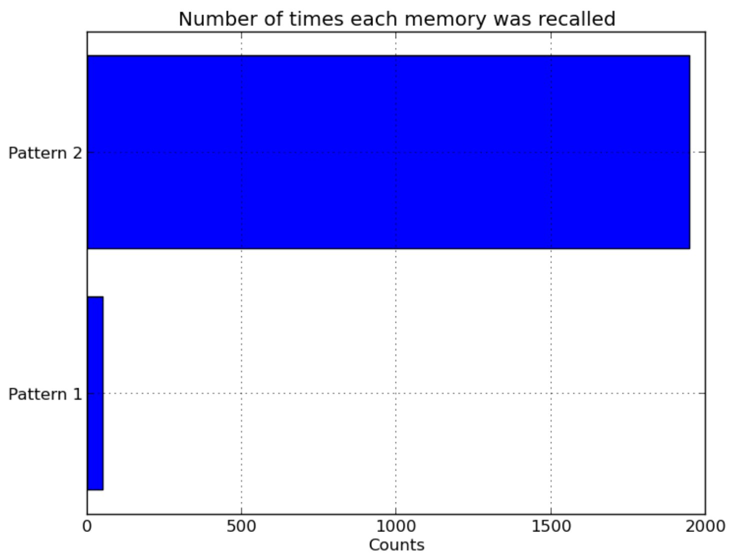

Figure 3. $\alpha = 0.1$ and $\beta = 0.4$. $N_1 = 51$ and $N_2 = 1949$

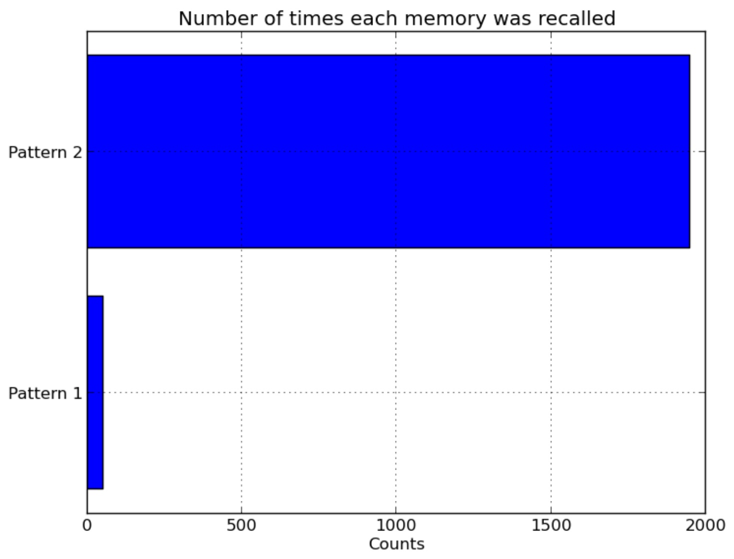

Figure 4. $\alpha = 1$ and $\beta = 4$. $N_1 = 53$ and $N_2 = 1947$

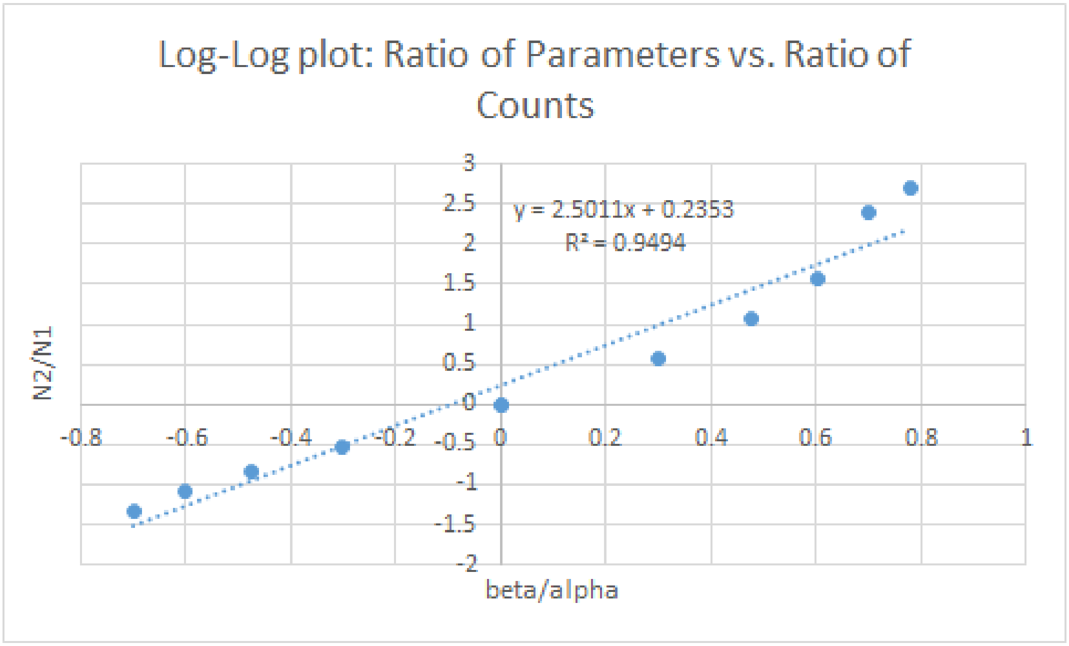

Figure 5. Shows how $\frac{N_2}{N_1}$ changes with $\frac{\beta}{\alpha}$

Conclusion

We created a Hopfield Network in order to study how the learning algorithm, Eqn.(4), affected the networks ability to recall a memory. We found that the actual values of α and β alone, but rather their relationship to each other, $\frac{\beta}{\alpha}$. We found that N_2 is proportional to $(\frac{\beta}{\alpha})^{2.5}$. However, it is worth noting that this may only be true for network that had one memory stored initially and another memory that was added using Eqn.(4). This dependency is likely to be different if there were more memories stored, either initially or through the learning algorithm.

References

1. N.J Giordano and H. Nakanishi, *Computational Physics*. (Pearson Education, 2006)