Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 3

Hopfield Networks

Jesús Ventas Muñoz de Lucas, Willem van der Schoot

September 24, 2019

1 Main objectives and scope of the assignment

Our major goals in the assignment were

- Build Hopfield Networks from scratch
- Investigate Convergence, Attractors, Update methods, Energy, Distortion Resistance, Capacity and Sparse Patterns

2 Methods

The programming language used was Matlab, all networks were built from scratch so no toolboxes were utilised.

3 Results and Discussion

3.1 Convergence and attractors

The first experiment was to look at the 'Little model' Hopfield Network. All updates were performed synchronously (matrix multiplication update). The network was composed of 8 neurons (8x8 weight matrix), allowing self connections. The network was trained on 3 patterns $x1, x2, x3$. It was able to store all 3.

Next, given three patterns with slight noise, $x1d$ - 1 bit flipped compared to $x1$ and $x2d, x3d$ - two bits flipped compared to $x2, x3$ respectively. After 2 iterations each, $x1d \rightarrow x1, x3d \rightarrow x3$, while $x2d$ reach another fixed pattern, a spurious pattern.

14 attractors were found in this network. When variations of $x1, x2, x3$ with too much added noise were fed into the network, the network converged to other attractors, such as in Table 1.

3.2 Sequential Update

Here, the network had 1024 neurons with self connections, the inputs were flattened 32x32 images. To start the network was trained on 3 patterns and it was able to store those correctly, stabilising after only 1 iteration. Two degraded patterns were then tested, as seen in Figure 1. The degraded

x2	-1	-1	-1	-1	-1	1	-1	-1
	-1	-1	-1	-1	1	-1	-1	-1
	-1	-1	1	-1	-1	1	-1	1
x1	-1	-1	1	-1	1	-1	-1	1
	-1	-1	1	-1	1	1	-1	1
	-1	1	-1	-1	-1	1	-1	-1
x3	-1	1	1	-1	-1	1	-1	1
	-1	1	1	-1	1	-1	-1	1
	1	-1	-1	1	1	-1	1	-1
	1	1	-1	1	-1	1	1	-1
	1	1	-1	1	1	-1	1	-1
	1	1	-1	1	1	1	1	-1
	1	1	1	1	-1	1	1	1
	1	1	1	1	1	-1	1	1

Table 1: The 14 attractors, including $x1, x2, x3$

version of $p1$ converged immediately with one synchronous update. The degraded pattern $p11$, converged with two iterations to a spurious pattern.

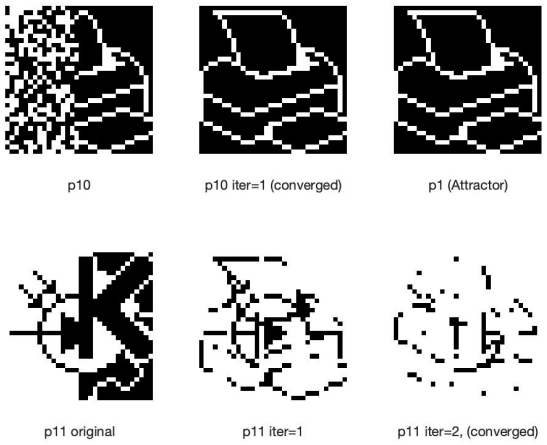

Figure 1: Feeding degraded patterns to network with synchronous update

Next the update method was modified to the random sequential update. The degraded patterns $p10, p11$ were once again fed in, refer Figure 2. Once again $p10$ converged to $p1$, after roughly 1200 iterations. $p11$ this time however, did not always converge to the same patter. $p11$ either converged to the spurious pattern previously, or to $p3$.

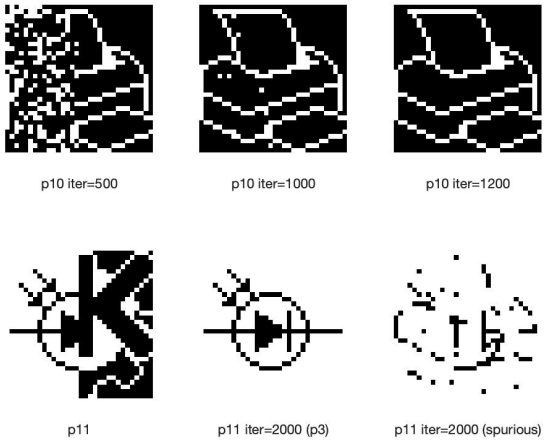

Figure 2: Feeding degraded patterns to network with asynchronous update

3.3 Energy

In the section the energy function was studied, $E = -\sum_{n=j}\sum_{n=i} w_{i,j}x_i x_j$.
For the patterns $p1, p2, p3$ the respective energy was:
$E_{p1,p2,p3} = [-4.9131 \times 10^5, -4.6614 \times 10^5, -4.9911 \times 10^5]$
For the two degraded patterns the energy was less negative:
$E_{p10,p11} = [-1.4199 \times 10^5, -0.5922 \times 10^5]$
This was as expected, attractors such as $p1, p2, p3$ are Energy minima.

When studying how the Energy changed using the sequential update, for two degraded patterns $p10, p11$, it was found that $p10$ simply converged straight to $p1$, while $p11$ drops to the level of $p2$, before dropping to a further attractor (Figure 3).

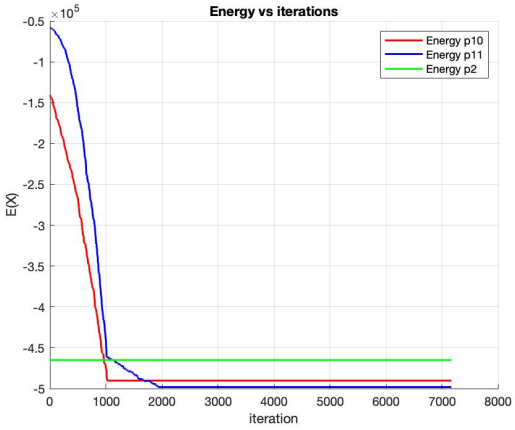

Figure 3: How energy changes from iteration to iteration when the sequential update rule is used to approach an attractor

Finally for this study of Energy, two cases were studied: symmetric and non-symmetric random initialised matrices. The non-symmetric matrix as seen in the left of Figure 4, never converged but oscillated between states. While with the symmetric matrix, convergence did occur, this follows the theory that a symmetric matrix will always decrease energy towards a minima when the state changes.

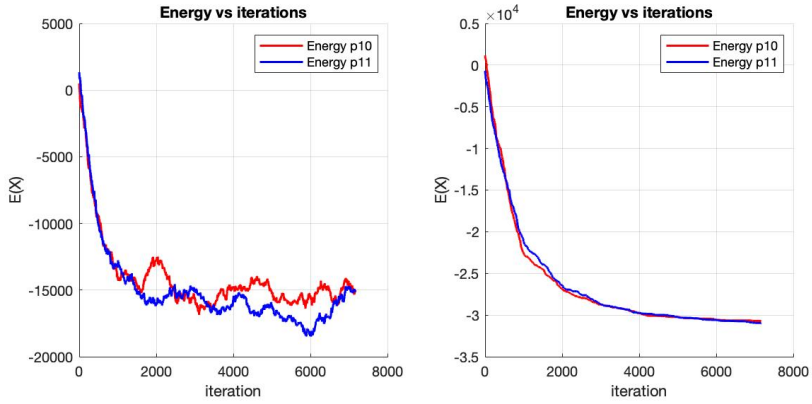

Figure 4: Left, random initialised weight matrix Energy evolution. Right, symmetric initialised weight matrix Energy evolution.

3.4 Distortion Resistance

For this section patterns p1, p2 and p3 have been distorted and then introduced to the Hopfield net. Each experiment have been repeated 100 times in order to get averaged results. As seen in Figure 5 the NN is able to recover slightly distorted patterns. Patterns with less than 20% of distortion can be recovered over 90% of times, whereas patterns with distortions above 40% can never be recovered. It is also interesting to see how when patterns are highly distorted (over 70%) is very likely to obtain their symmetrics, which are also attractors as is well-known. Other interesting feature of both graphs in Figure 5 is that pattern p2 is slightly more resistant to distortion than p1 and p3. This is very likely to be due to is less correlated with other patterns and so less likely to converge into a spurious pattern.

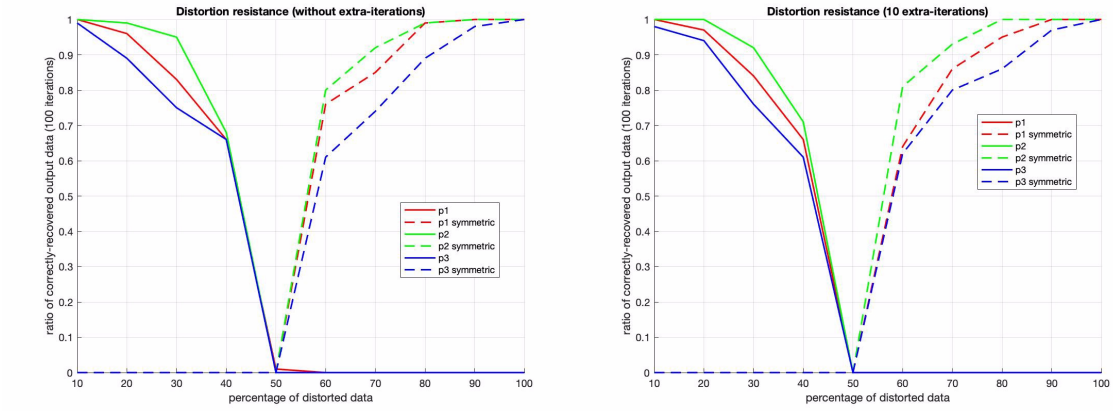

Figure 5: Average ratio of correctly-recovered patterns for different amount of distortion after 100 experiments with no extra iterations (left) and with 10 extra iterations (right)

It is also interesting to see in Figure 5 that adding extra-iterations does not improve the performance of the network in terms of distortion resistance.

3.5 Capacity

The theoretical capacity of a Hopfield network is 0.138d, where d is the number of neurons of the network. However, if the pattern p4 is introduced to the matrix W, the number of recovered patterns p1,p2 and p3 is always 0 for distortions from 10% to 100%. This is very likely to happen

because p4 is highly correlated with p1,p2 and p3. If random patterns are added to W instead of the pattern p4, the drop in performance is more gradual, specially for p2 which is the most resistant pattern. In Figure 6 this more gradual drop in performance can be observed. However, when more than 10 random patterns are introduced to the NN, none of the patterns can be recovered. This discordance between the theory and the experiments can be due to for reaching the theoretical capacity patterns are supposed to be orthogonal or quasi-orthogonal, whereas patterns under study are highly correlated each other.

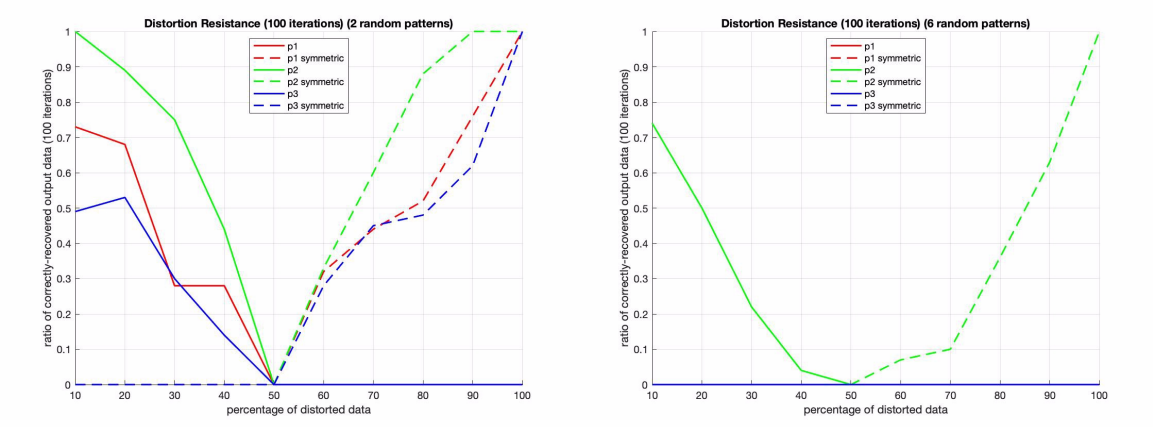

Figure 6: Average ratio of correctly-recovered patterns for different amount of distortion after 100 experiments when 2 random patterns are added to W (left) and when 6 random patterns are added to W (right)

The next experiment consisted of training a 100-neurons Hopfield net with 300 patterns and see the evolution when a new pattern is added to W. Firstly, without removing self-connections (Figure 7 left) it can be observed that the number of stored patterns start decreasing after storing 13-15 patterns and start increasing again when so many patterns are stored. That increasing is produced by the effect of self-connections, since they enforce the network to remain in the current state. However, the ratio of recovered patterns drops to 0 after the theoretical capacity is reached (around 14 patterns in this case) and never increases again, as expected. If self-connections are removed (Figure 7 right) the number of stored patterns does not increase once it reaches the minimum.

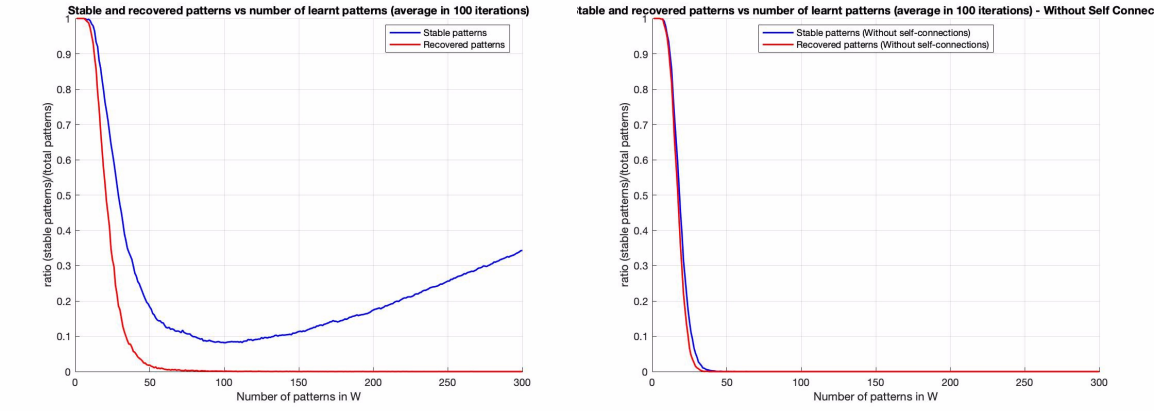

Figure 7: Average ratio of stable and recovered patterns vs number of stored patterns with self-connections (left) and without self-connections (right)

If patterns are biased (there are more +1 than -1) the capacity decreases (from around 13 patterns to around 6 patterns). This could happen because now stored patterns are more likely to be more correlated.

3.6 Sparse Patterns

The aim of this section is to investigate the effect of adding a bias to the training, since patterns are not usually balanced and the capacity decreases as seen in Section 3.5. A correct selection of the bias according to the activity of training patterns improves the network performance. In Figure 8 (left) it can be observed the number of stored patterns for different biases when the activity is 0.1 (10% ones and 90% zeros). In the previous case a correct bias would be between 12 and 20. In Figure 8 (right) the performance for different biases when there are lower activity levels can be seen. The main feature of this graph is that the less the activity, the lower the bias and also the range of good biases is reduced. For example, if the activity is 1% there is only one bias that allows to store the 300 patterns.

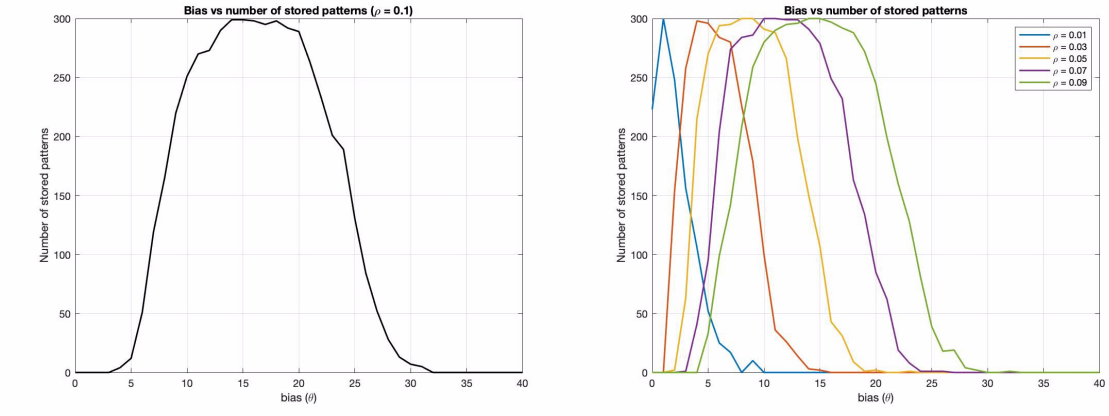

Figure 8: Bias vs number of stored patterns for activity = 10% (left) and for lower activities (right)

4 Final remarks *(max 0.5 page)*

This lab definitely improved our knowledge on Hopfield networks and let us play around with the networks. We felt we sufficiently met the learning objectives and enjoyed the lab.