Artificial Neural Networks and Deep Architectures, DD2437

# Short report on lab assignment 3

## Hopfield networks

Alice Karnsund, Elin Samuelsson and Irene Natale

September 26, 2018

## 1 Main objectives and scope of the assignment

Our major goals in the assignment were

- to implement, train and explain the underlying principles of the Hopfield network
- to explain the attractor dynamics and the concept of energy
- to demonstrate how autoassociative networks can do pattern completion and noise reduction
- investigate the question of storage capacity and explain features that help increase it in associative memories

## 2 Methods

We used `Matlab` throughout in this lab without any use of toolboxes.

## 3 Results and discussion

### 3.1 Convergence and Attractors

The network was able to store the patterns $x_1, x_2, x_3$. We know this because we receive the same patterns back when applying the updating rule. When we apply the updating rule to $x_1d, x_2d, x_3d$ the answers are different, as shown in Fig. 1. The 1 bit distortion vector $x_1d$ still converges to its original $x_1$, while for the 2 bits distortion vectors $x_2d$ and $x_3d$ it depends. The second one

converges to $x_3$, while to other converges to another fixed points (closer to the input pattern).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -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 |
| -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 |
| 1 | 1 | 1 | 1 | 1 | -1 | 1 | 1 |

Figure 1: Left: output from $x_1d, x_2d, x_3d$. Right: all fixed points of the network.

If we try all possible input vectors, we obtain 14 different fixed points, i.e. 14 attractors. If the input vector differs a lot from any of $x_1, x_2, x_3$, for example when half of the input vector is different, the output does not converge to any of the training vectors, but to some other attractor.

## 3.2 Sequential Update

We train the network with the first three pictures. To see if these end up as fixed points, we should receive the same patterns back when we apply the update rule to these. This we did only after one iteration, meaning that convergence was instantaneous.

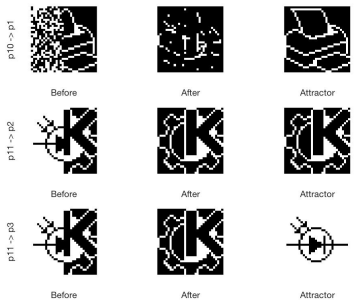


Figure 2: Middle column represents the results of the degraded patterns `p10` and `p11` after 7 iterations. The network succeeds to complete `p11` into `p2`.

In Figure 2 we see that the network will only succeed to complete `p11` as `p2`. The number of iterations needed to reach an attractor is roughly $log(N)$, therefor we used $round(log(N)) = 7$ iterations to obtain Figure 2, where all units are updated in each iteration. At this point the results had converged. The number of incorrect states in each result, compared to the attractor it aimed for, turned out to be: $[p10 \to p1, p11 \to p2, p11 \to p3] = [135, 0, 728]$.

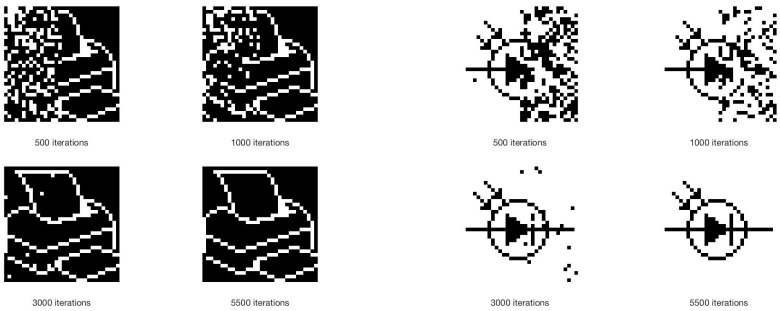


Figure 3: Units selected randomly when calculating the states for `p10` and `p11`.

In Figure 3, only one unit is updated in each iteration, thus the amount of iterations needed is much greater now. Comparing Figure 3 with row one and three in Figure 2 we see that, when the units are chosen randomly the network succeeds to complete `p10` and `p11` into `p1` and `p3`. Though it does not work to use the random version to obtain `p2` from `p11`. `p11` will always turn out as `p2` with the normal version and as `p3` with the random version.

## 3.3 Energy

For the three different attractors `p1`, `p2` and `p3` the energies are:
$E_{attractors} = [-1439.4, -1365.6, -1462.3]$

For the two distorted patterns, `p10` and `p11`, the energies are:
$E_{distorted} = [-415.9805, -173.5000]$

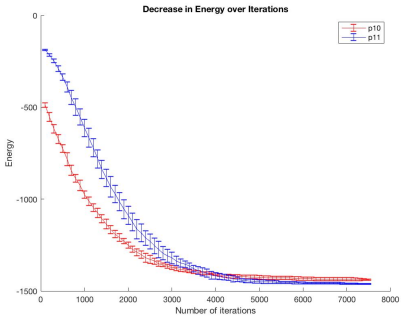


Figure 4: Decrease in energy for the patterns `p10` and `p11` as they approaches attractors. The results are obtained over 10 iterations.

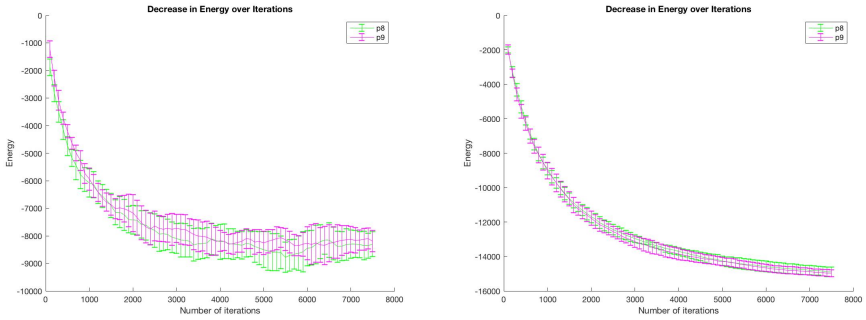


Figure 5: Decrease in energy for the patterns `p8` and `p9` when (1) W is normally distributed random numbers and (2) when W is a symmetric version of the normally distributed W. The results are obtained over 10 iterations.

As shown to the left in Figure 5, when W is normally distributed random numbers, the network never converges, it cycles between different states forever. But when W is symmetric, to the right in Figure 5, the network will converge. In this case we used `p8` and `p9` as arbitrary starting states.

## 3.4 Distortion Resistance

Figure 6 Left shows that the network is able to recover pictures with less than 50% noise. When the noise increases further, we approach the same pattern but with inverted colours. Thus, the network can no longer restore the original picture, but the inverted one. An example of this is shown in Figure 6 Right. The second picture p2 has the best noise tolerance.

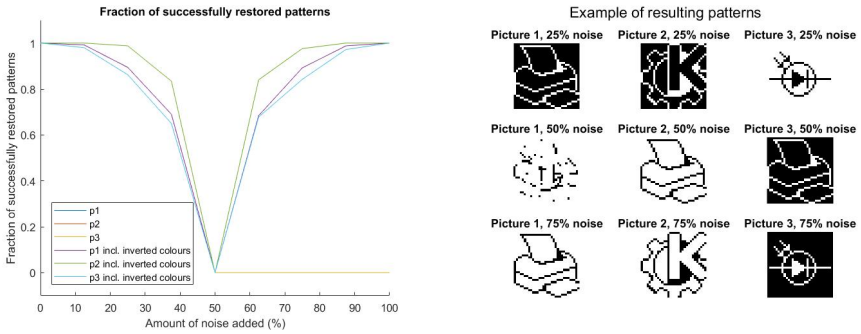


Figure 6: Left: The fraction of successfully restored pictures over 500 iterations, with and without including the negatives of the pictures, when training with 3 pictures. Right: An example of the resulting patterns for different levels of noise, when training with 3 pictures. The title stands for the input and the corresponding picture for the output.

There are nine possible attractors, shown in Figure 7 left. They are the pictures themselves and their negatives, one messy picture and its negative, and one completely black picture. When restoring x, the network sometimes need more than one update for convergence, but the number of successfully restored patterns is equally good regardless of one or more updates.

## 3.5 Capacity

Figure 7 Right shows that by adding p4 to the weight matrix, the network abruptly loses its ability to restore any picture. However, when replacing the picture data with random data of the same size, the network can restore all data with noise less than 50%.

Figure 8 Left shows that the number of stable patterns increase with number of patterns added to the network. On the other hand, the number of successfully recovered patterns from 5% noise quickly reaches a maximum and then decreases down to zero again. This implies that training on a large number of patterns leads to poor generalization performance. By removing the self-connections, and thus the spurious patterns, the stable patterns exhibit the same behaviour.

As seen in Figure 8 Right, the mean maximum number of retriavable patterns is 9.42, which is not too far from the theoretical value of 13.8. Further, having biased patterns decreases the network's performance regarding both stable and retrievable patterns. This can explain the previous difference between picture data and random data. The pictures are biased towards one colour, and are thus harder to recover. This is supported by the observation of p2 being the most noise tolerant picture in section 3.4, since p2 has a more even distribution between 1 and -1 than p1 and p3.

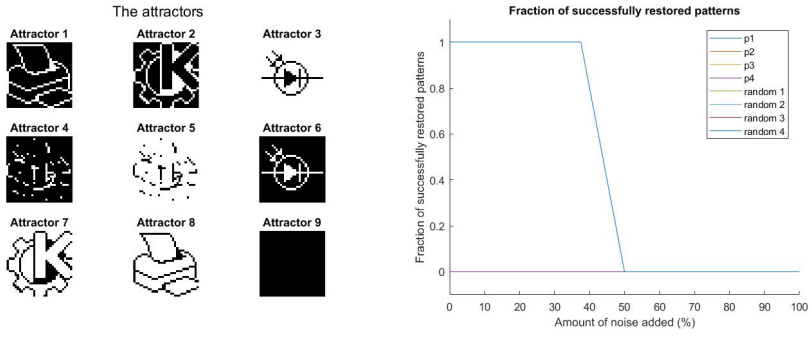


Figure 7: Left: The attractors, when training with 3 pictures. Right: The fraction of successfully restored pictures over 500 iterations, for picture and random data, when training with 4 patterns.

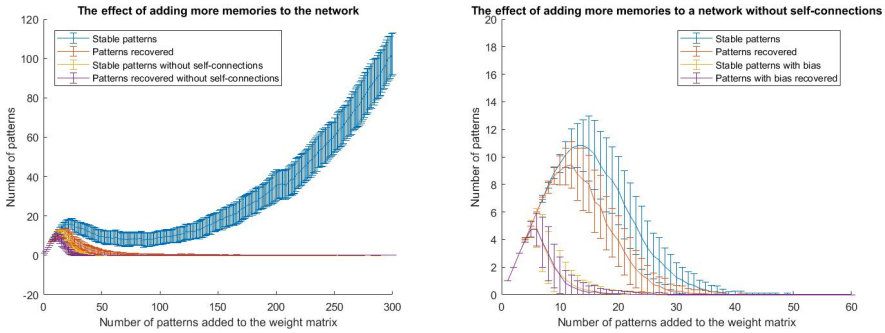


Figure 8: Left: The effect of removing self-connections when adding more memories to the network. A 5% noise was used, and the test was repeated 100 times. Right: The effect of having biased patterns when adding more memories to a network without self-connections. Again, a 5% noise was used, and the test was repeated 100 times.

### 3.6 Sparse Patterns

From Fig. 9 it is clear that the optimal value for the bias must be around 17.50. The network stores less patterns if the bias is lower or above this value. It is also clear that the percentage of active neurons in the training patterns influences the performance. In fact, for all the considered values of the bias, input patterns with less active neurons are easier to store during the training.

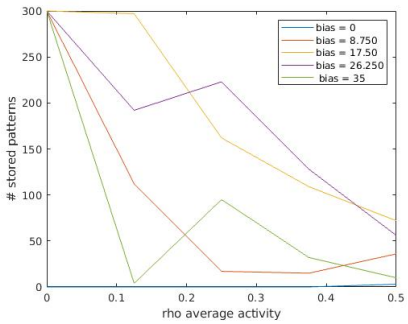


Figure 9: Number of stored patterns with varying bias and average activity.

## 4 Final remarks *(max 0.5 page)*

In our opinion the lab was well structured and we have been guided to a reasonable understanding of auto-associative networks.