Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 3
Hopfield networks

Benjelloun Hamza, Benchekroun Hamza and Ait lahmouch Nadir

February 19, 2021

1. Main objectives and scope of the assignment

Our goals from this assignment are:

- Study the Hopfield network addressable memory, and its attractors.
- Evaluate the storage capacity and stability depending on the distribution of patterns, and resistance to distortion.

2. Methods

In this Lab, we used **Python** and the **numpy** library for the matrix manipulations and operations.

3. Tasks and questions

3.1 Convergence and attractors

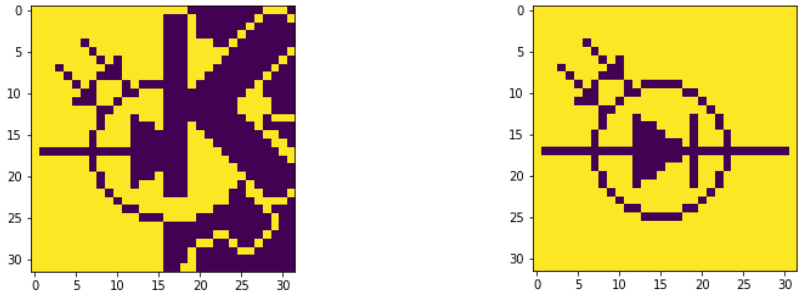
In this part, the diagonal matrix of the weight matrix is not set to 0.
The patterns x1, x2 and x3 we used to train the model are perfectly recalled, therefore they are attractors in this network.

With both **batch** and **seq** mode, only the second noisy pattern x2d is not recalled.
(Using zeros as the diagonal of the weight matrix yields different results, all the distorted patterns are recalled in **seq** mode, but only x1d converges to x1 in **batch** mode).
All the max of iteration in **seq** mode before convergence is 2.

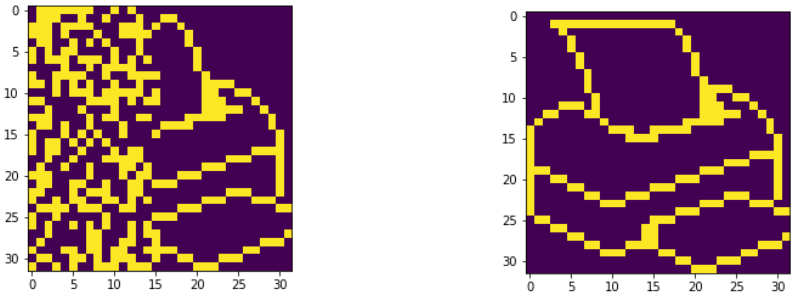
- There are 14 attractors (6 if we use zeros as the diagonal).
- When we use an even more dissimilar pattern, it's rare that the noisy pattern converges to the same pattern, it converges to other attractors.

3.2 Sequential update

- Here also, the three patterns used to learn the model are stable.
- Both of the distorted patterns are recalled. But for p11, sometimes it converges to other attractors.

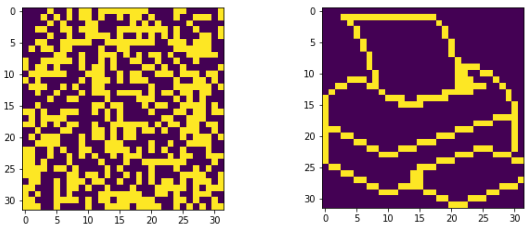


p11 (left) and output.

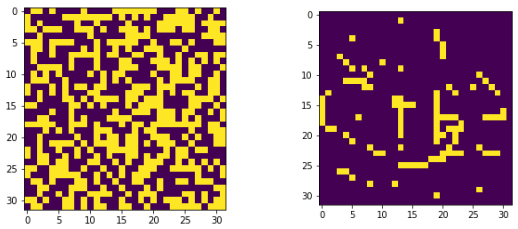


p10 (left) and output.

- With selecting the units randomly, most of the time, it takes two steps each with 1024 updates for the network to converge to a stable state. Sometimes it converges to one of the patterns we learned and sometimes to other attractors, like in the examples below.



Random pic and its output



Random pic and its output

3.3 Energy

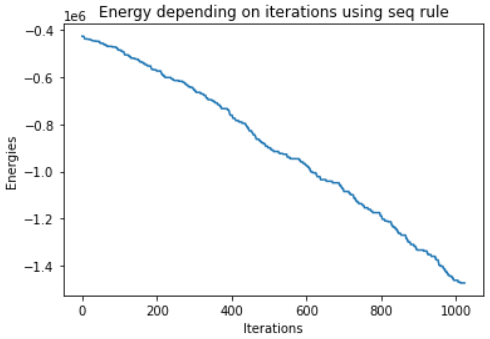
We found 4 attractors, p1, p2, p4 and the pic we saw in the section above.
Here's a table of the energy at these attractors:

Pattern	Energy
p1	-1473936
p2	-1398416
p3	-1497344
4th attractor	-1634316

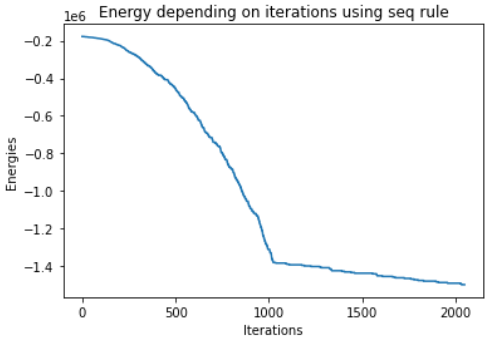
And here's the energies of the distorted patterns.

Distorted pattern	Energy before recall	Energy after recall
p10	-425964	-1473936
p11	-177664	-1497344

Let's now follow how the energy changes from iteration to iteration using the seq update.



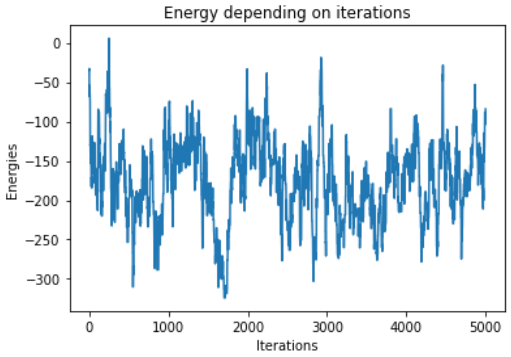
Energy for p10

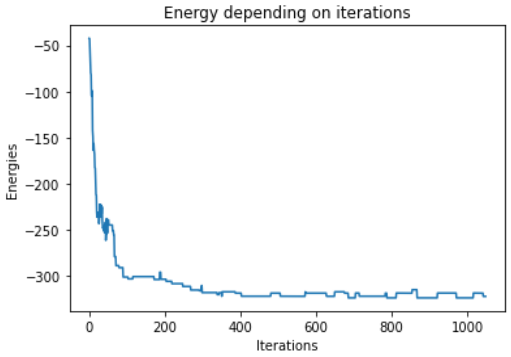


Energy for p11

We can see that the energy is decreasing regularly before reaching a stable state.

When we are using a weight matrix by setting the weights to normally distributed random numbers, the network doesn't converge and the energy is unstable *(left pic below)*. However, when we turn this matrix into a symmetric one, energy decreases iteration after iteration *(right below pic)*.





And this can be explained by the energy formula. Let E be the energy at some iteration, and E' this same energy after a node p_i has been updated to p'_i.
we have:

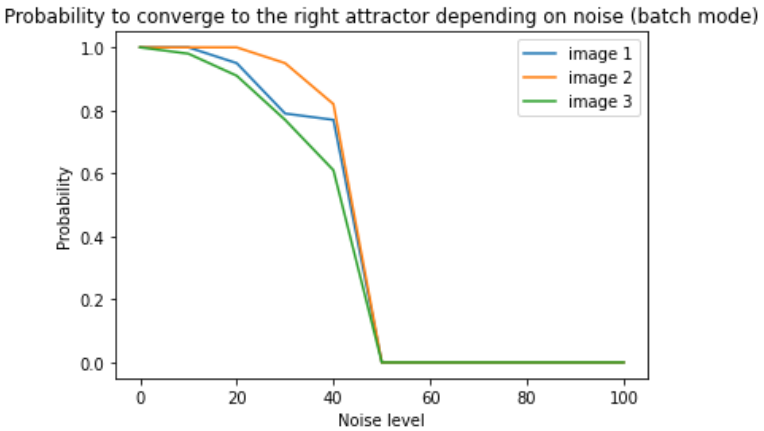
$$E - E' = -(p_i - p'_i)\sum_{j=1}^{N}(w_{j,i} + w_{i,j})p_i$$

But in symmetric matrices, it's:

$$E - E' = -2(p_i - p'_i)\sum_{j=1}^{N} w_{j,i}p_i$$

Therefore, with non symmetric matrices, we cannot regroup the two terms and then we can't know the sign of the difference, while with symmetric ones the difference is positive because $(p_i - p'_i)$ and $\sum_{j=1}^{N} w_{j,i}p_i$ are of different signs.

3.4 Distortion resistance

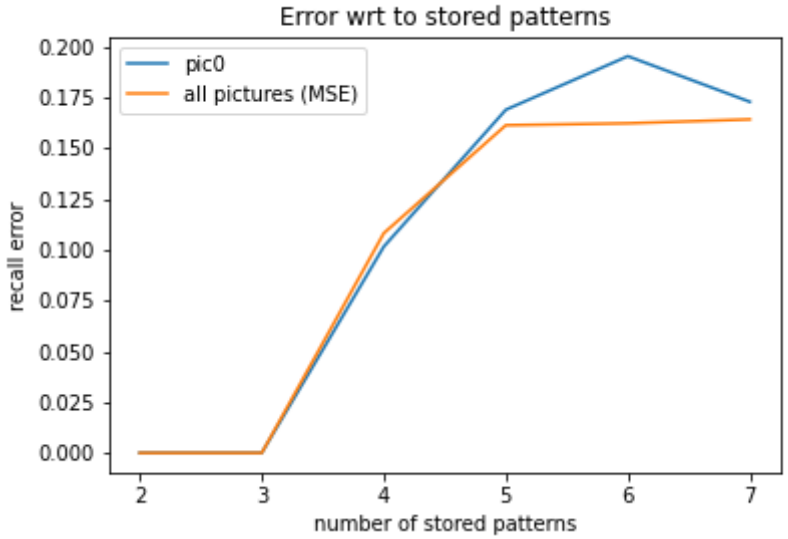


The figure shows that the network can remove noise up to 10% for the first and the third pictures, and up to 25% for the second picture. It seems that the second picture is more resistant to noise and the third image is less resistant.

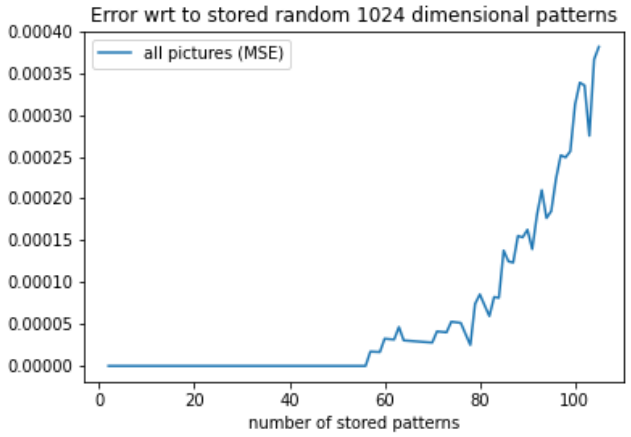
For a noise greater than 50% the network converges to a spurious attractor. It is normal because the picture doesn't look like the original one.

3.5 Capacity

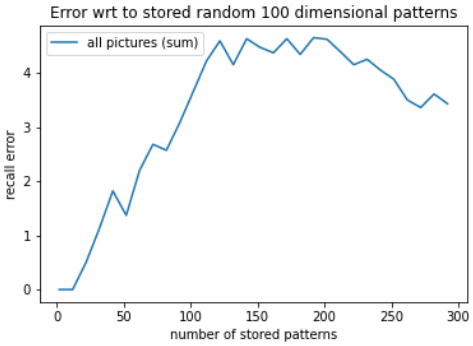
In this part, we use the Mean Squared Error on the Recall error for each of the patterns. We start by testing the capacity on the given images, the recall error is different than 0 starting from 4 patterns, which is because the images are similar.

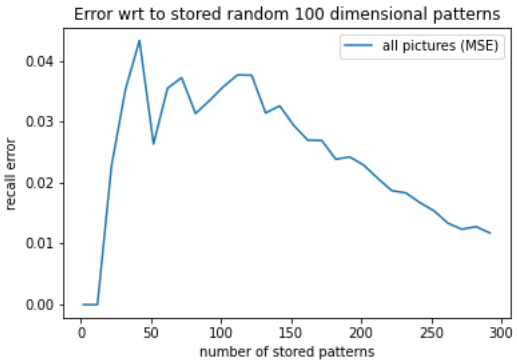


We generate random patterns in this 1024 dimensional space, and we get the following expected results:

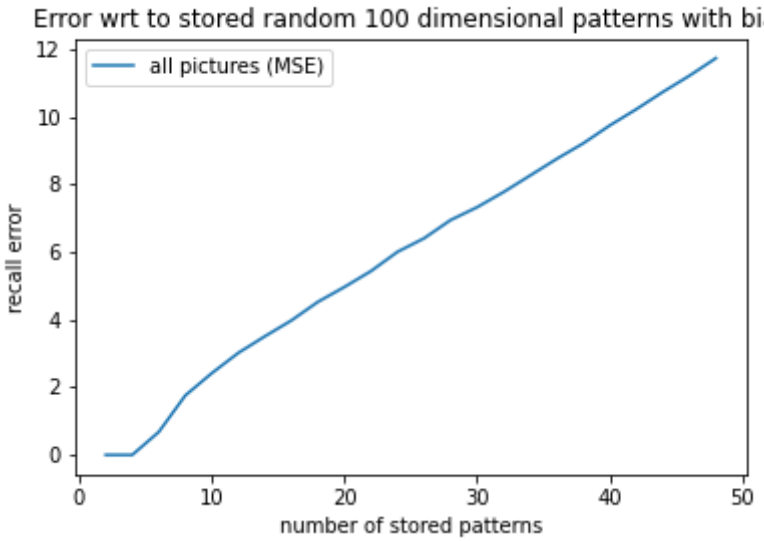


Switching to a 100 dimensional space, with an obvious smaller capacity, we use the MSE metric defined previously, and the sum of recall errors for each pattern too. The capacity with zero recall is 12.

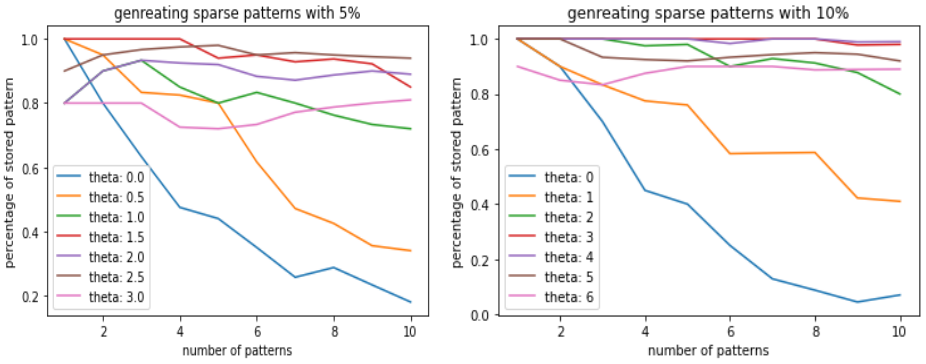




Adding bias decreases the capacity to 4, which is predictable since we approach the first case of the images in 1024 dimensional space.



3.6 Sparse patterns



The bias has a big influence on the capacity of the network to store patterns especially when we have sparse patterns (drawn from a skewed distribution). As we see in the figures for an activity of 10% the best bias is between 3 and 4 and for an activity of 5% the best bias is between 2 and 3. We conclude that the smaller the activity the smaller the bias needed.

5. Final remarks

We learned a lot in this lab about hopfield networks. They are useful at storing patterns and denoising them. However, patterns of interest are often biased and limit the storage capacity, easily producing spurious patterns. But we also learned that adding bias to the network solves the problem of capacity reduction.