# Hopfield Model

Neural networks – Assignment 03

By: Nadav Porat

Date: 10/5/20

# Number of memories and Overlap

The probability that the network will be unstable can be derived from the crosstalk, it goes according to the following formula:

$$P_{error} = P(C_i^{\nu} > 1) = \frac{1}{2}\operatorname{erfc}\frac{\sqrt{N}}{2p}$$

If we set N = 100 and allow $P_{error}$ to max out at 0.01, the maximum number of allowed memories to be stored is:

$$p_{max} \approx 19$$

* Assuming random patterns.

Overlap Function: In order to test the success of restoration, I define the following function:

$$overlap(A, B) = \frac{1}{N}\sum_i a_i b_i$$

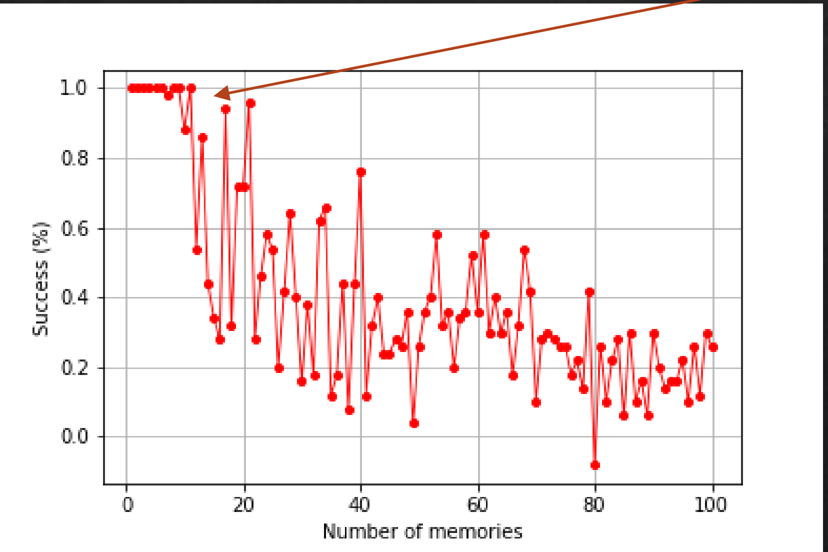
For A and B arrays of length N  with values +1 or -1 .

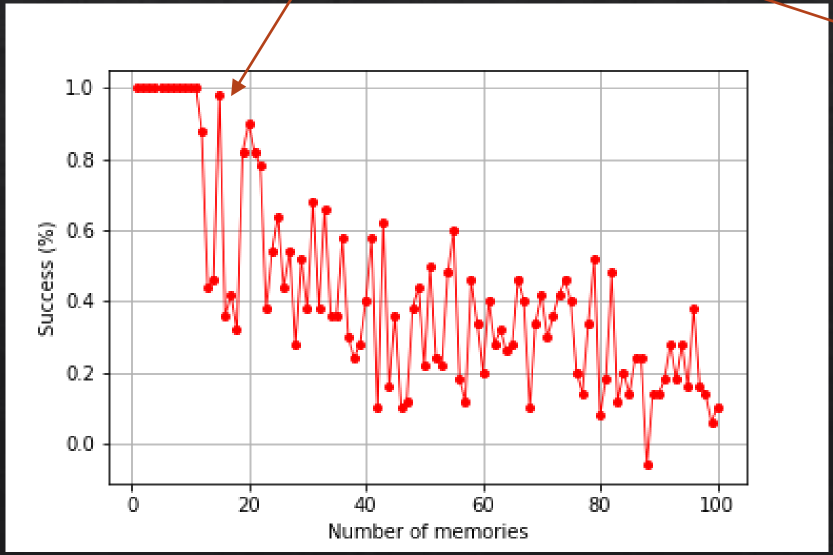
Test Network – 100 Cells

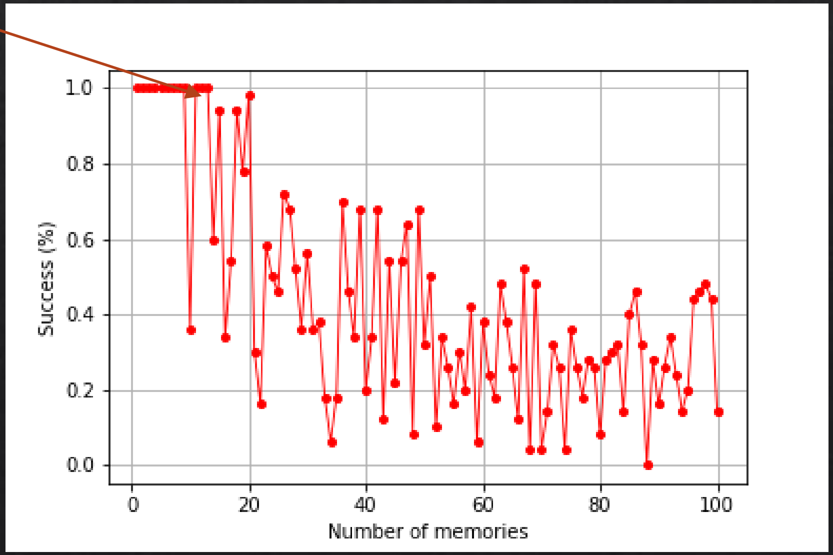
# Success vs Number of memories

In the following Networks I stored random-generated patterns as natural memories. Then I set the Network on a corrupted memory (35% of the Cells were chosen randomly and flipped) and measured the restoration success after a fixed amount of iterations. They all drop at about:

$$p = 17 \pm 2$$



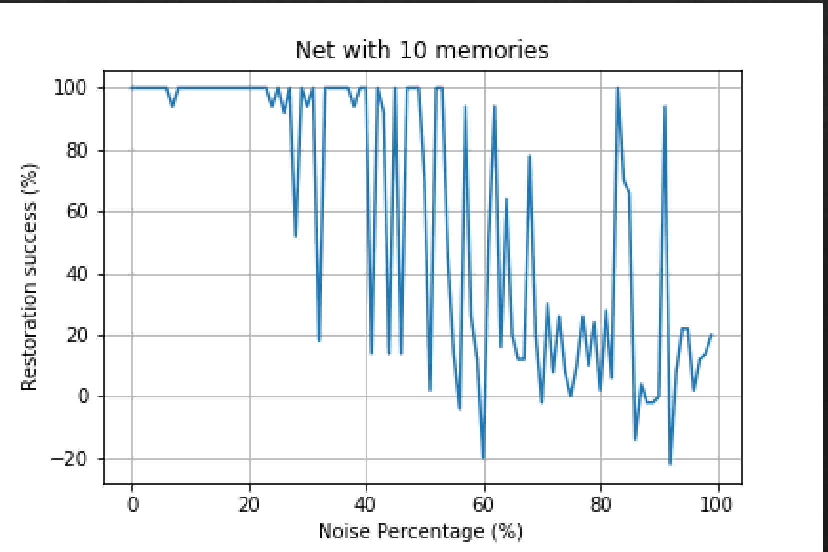



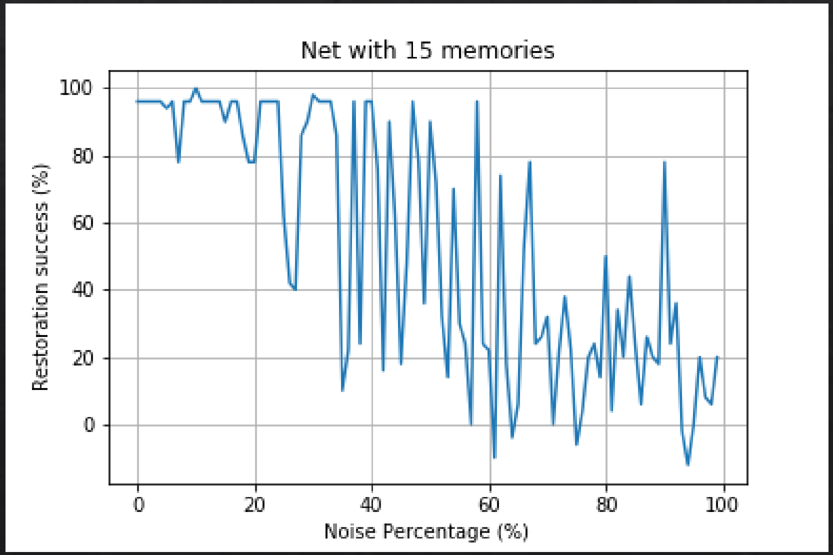
Test Network – 100 Cells

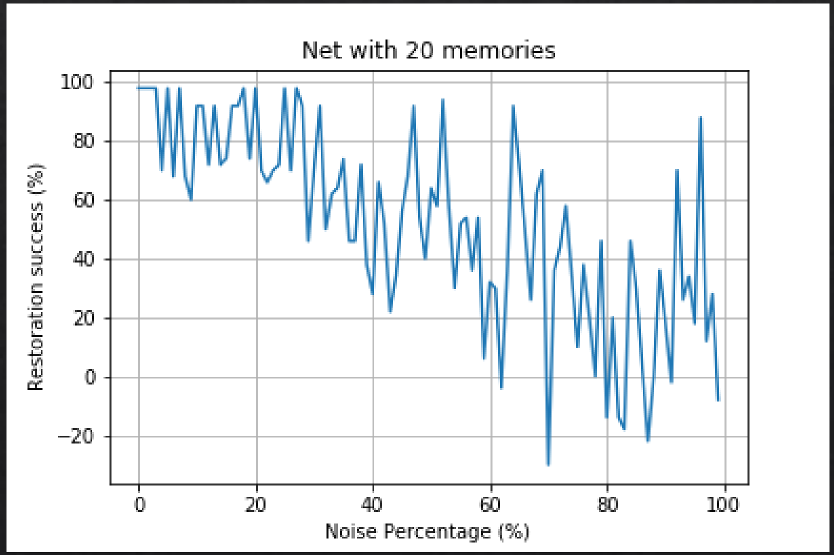
# Success vs Noise

In the following graphs, I inserted 10, 15, and 20 random memory patterns into the Network. Then chose one memory at random and corrupted it by adding varying degrees of noise. Then I measured the restoration success after a fixed amount of iterations.

We can clearly see that the restoration success drops as the noise increases. Also, the fewer the memories, the better the system can deal with noise, and the Network seems stable for small amounts of noise.



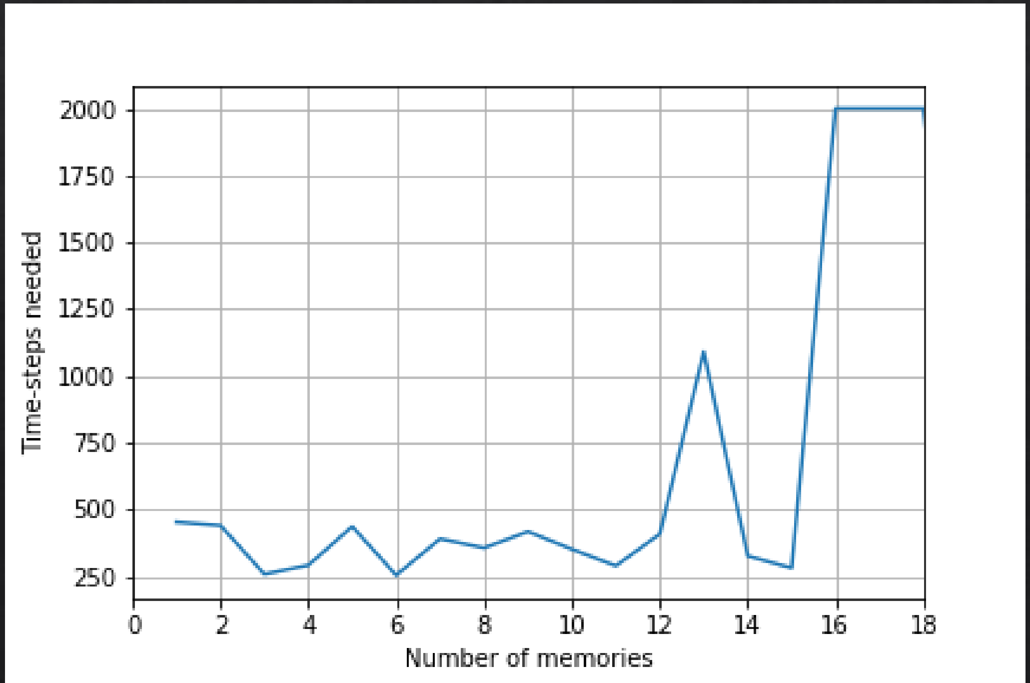



# Time vs Number of memories

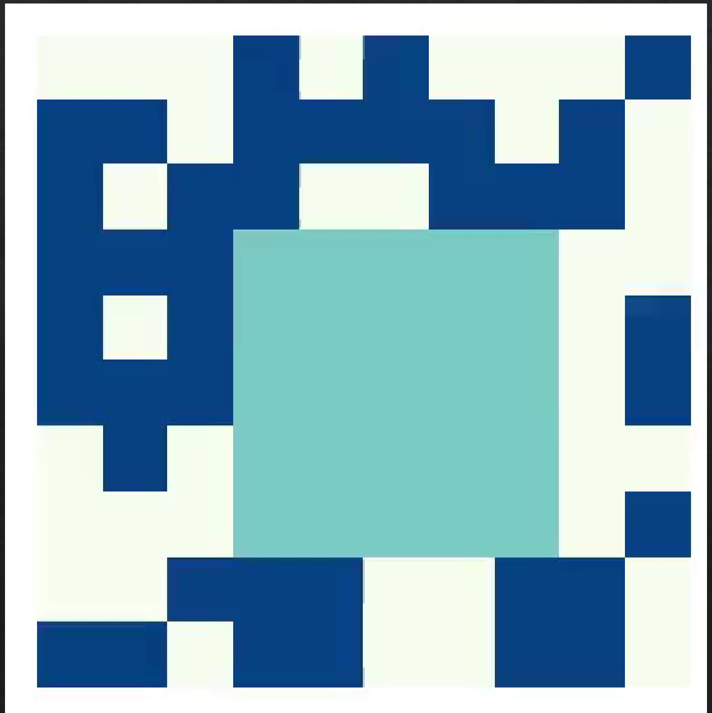
In the same network, the time-steps needed to restore a corrupted pattern grows as a function of the number of memories. The usual time-steps can be estimated around ~350.

* In the graph, 2000 time-steps is the limit of measurement.



# Deterministic distortion

I took one random pattern from the system and corrupted it deterministically. Meaning I changed a 4 by 4 square in the middle and set it as the initial state of the Net. The system's response:

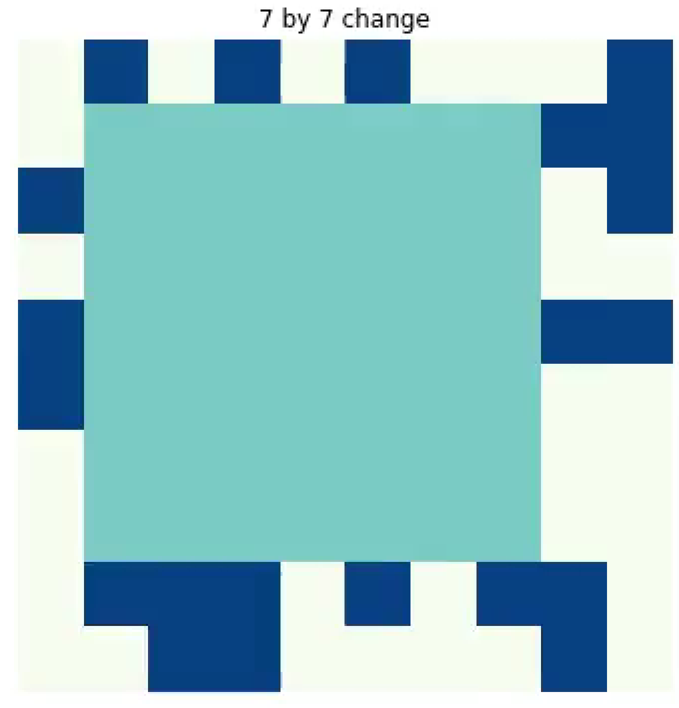


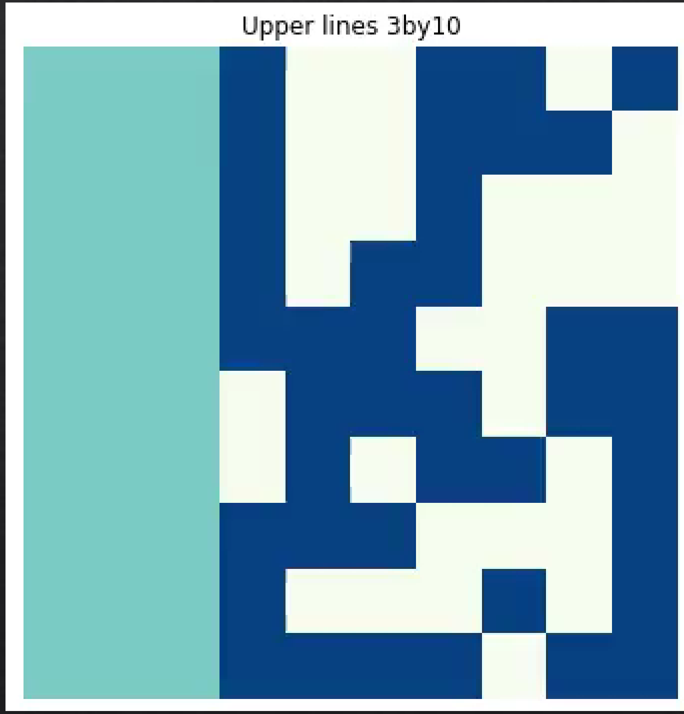
*Light blue – Cells with wrong values.

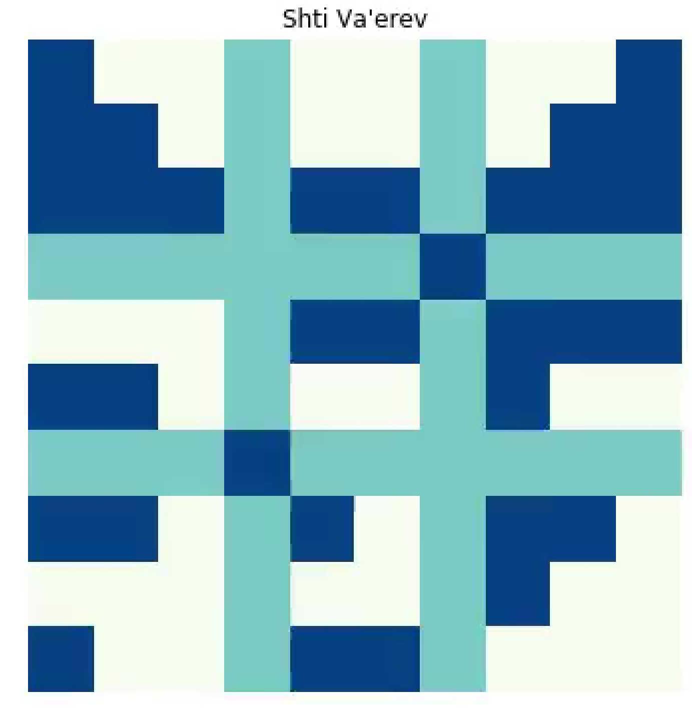
*Dark blue – Cells with correct values

# Deterministic distortion

More cool Vids:



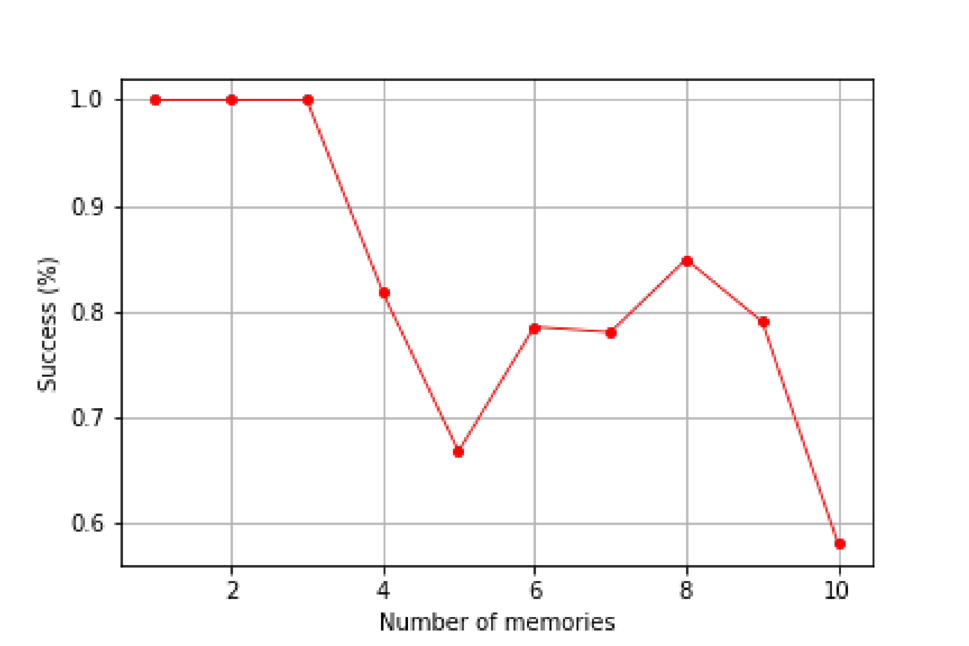



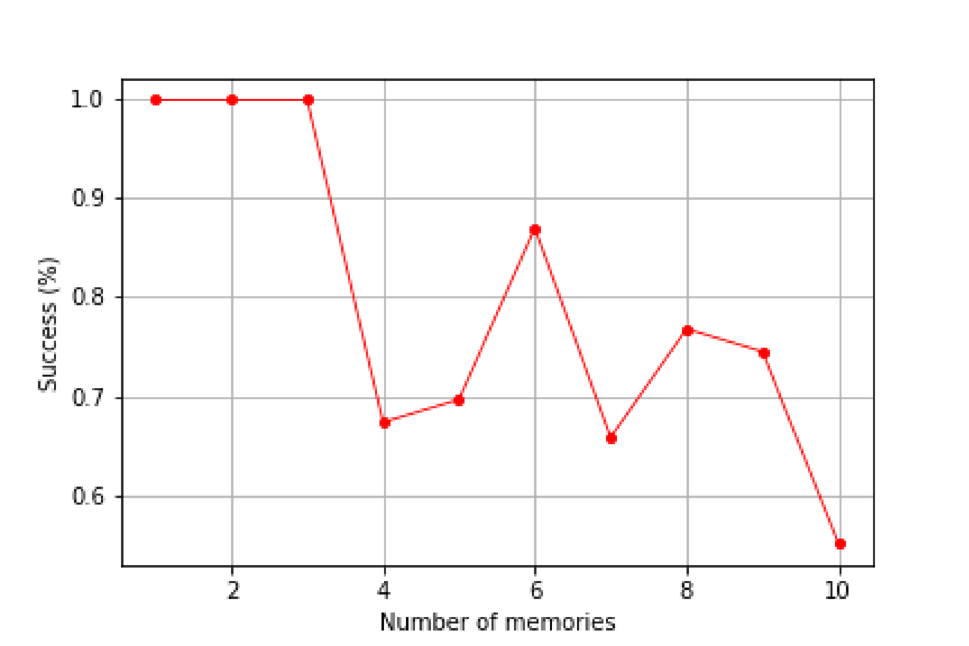




I can conclude that deterministic change has no meaning (with random patterns), only the amount of noise has meaning.

# MNIST Network

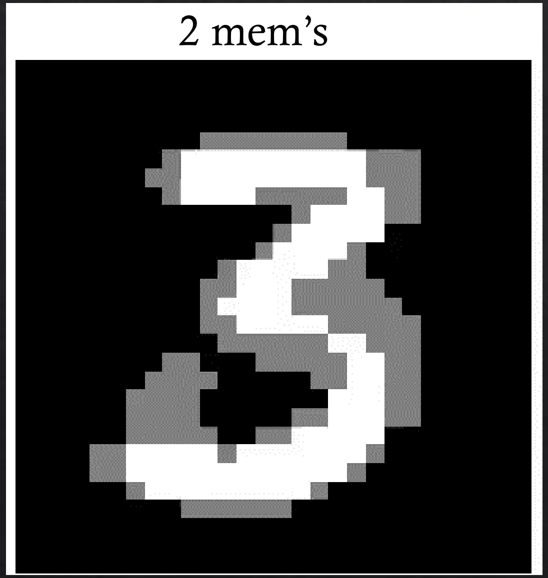
The MNIST Network behaves vastly different. Because of high correlation between pixel representation of decimal digits, the Network is barley stable with more than ~2 memories stored in (corruptness is at 20%).

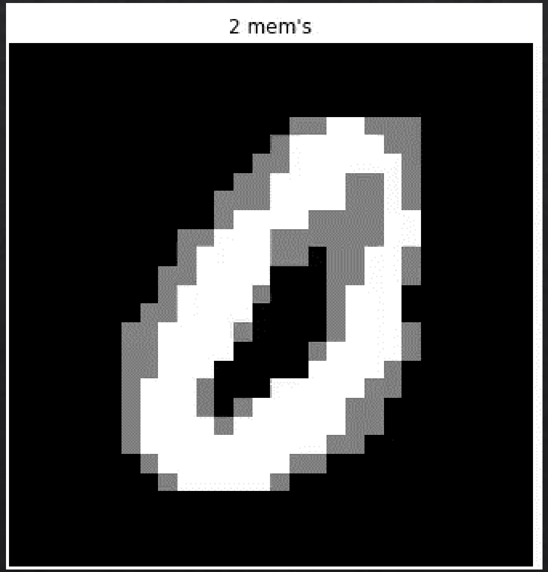


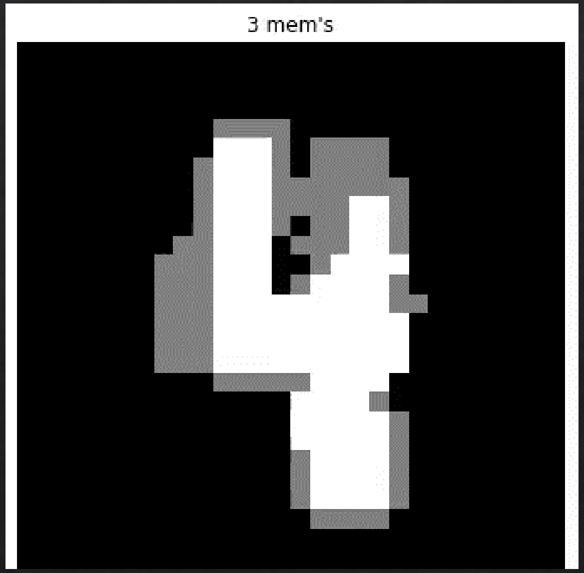



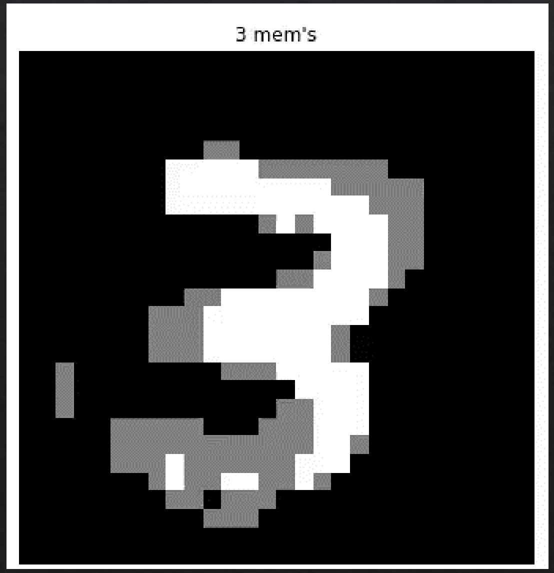
# Retrieving a digit from a different pattern

Using the MNIST database, I inserted a few random digits into the system. Then I set the Network on a different pattern corresponding to the digits it has memorized. Since the MNIST Network is barley stable with less than 3 memories, I tested it with 2-3 memories stored. The results are dependent on how similar the digit patterns are, and the amount of memories.



# Shifted digit patterns

Since the MNIST Network preformed poorly with more than three memories, I used a different set of digits in order to continue testing. The digits I used are bigger (32*32) and sharper, therefore provide better results:





* These photos and the function that implements them are taken from this guy “duskybomb” on GitHub.

# Shifted digit patterns

We can see that as I increase the memories stored in the Network, the restoration success drop.

- It is worth to note that both randomly corrupting a digit and shifting it increase its Hamming distance from the original digit. But shifting can also **decrease** the Hamming distance to another digit. So shifting can be worse than randomly corrupting.