Project 3 – Hopefield neural network
Juraj Mašlej

Contents:

1. Noise correction from input patterns

We used input patterns and noise levels specified in task. We selected n random pixels and flipped their values.

1.1. Cycle and fixed point detection

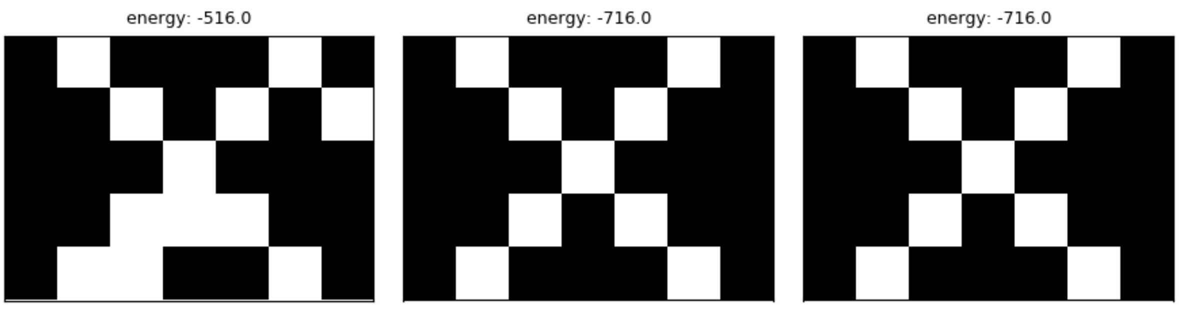
Cycle : we generated k-steps, if last step already appeared in k-2 states, we detected cycle.
Fixed point: last two steps of network are equal.

1.2. Noise correction

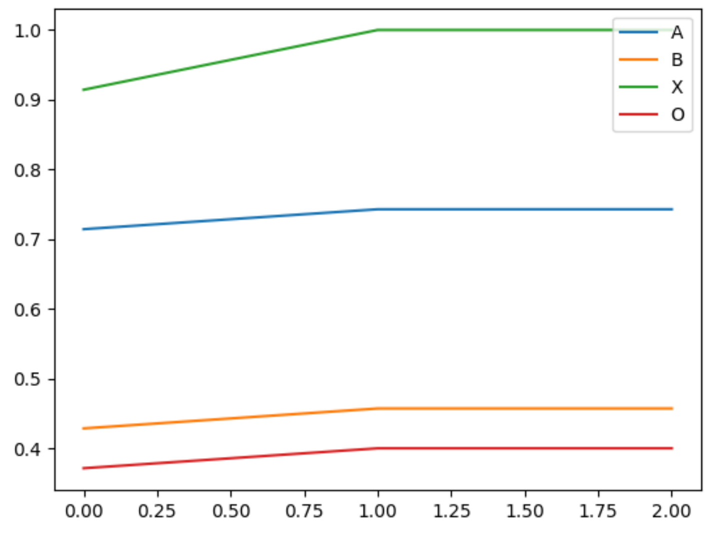
For noise 0 (no noise in patterns), network did not destroy patterns, thus performed correctly.
Noise 7(flipping 7 positions): patterns A,X and O were corrected correctly, noised pattern B did not reach correct state.
Noise 14: only noised pattern O reached correct pattern.
Noise 21: no pattern reached it correct, no-noise state.

All graphs are included in zipfile.
Naming: “A14states.png” = graph of states, original pattern was “A”, level of noise is 14, corresponding graph of overlaps is called “A14.png”
Note: energy is not plotted as graph, but marked with visualizing states, as it shows that energy lowers while getting closer to patterns.

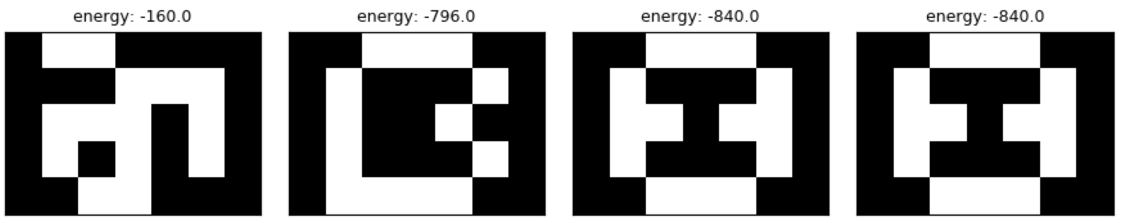
Original pattern X, noise 7, True attractor



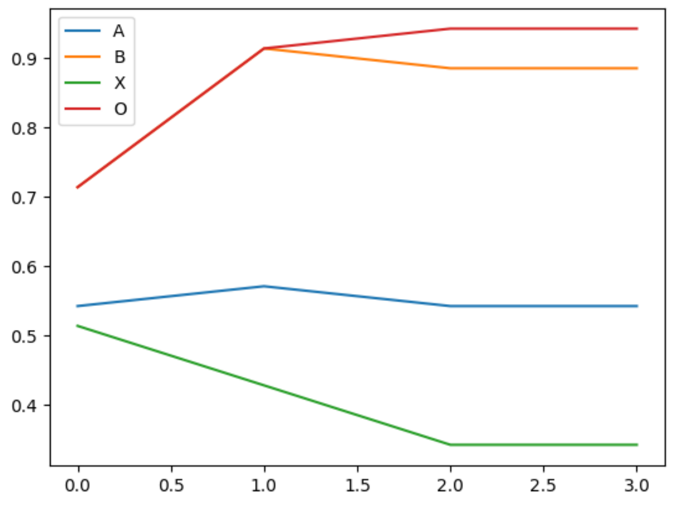
Graph of overlaps for previous network computation
States on x-axis, overlap on y-axis.



Original pattern B, noise 14, attractor is spurious



Overlap for previous network states:
x-axis: network states, 1(marked as 0) to 4(marked as 3)
y-axis: level of overlap, 0.5 = 50%

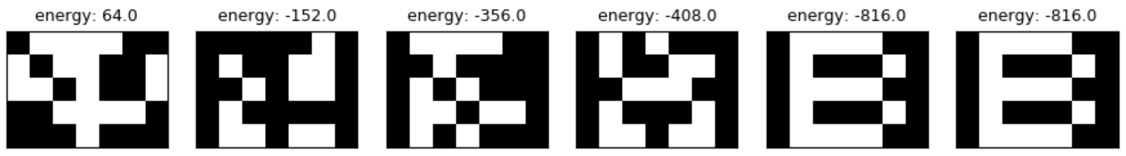


2. 5 random binary inputs

Inputs were generated randomly. All graphs added in zip file.

Naming: "states_random_1.png" network states for first random input, corresponding overlap graph "_random0.png"

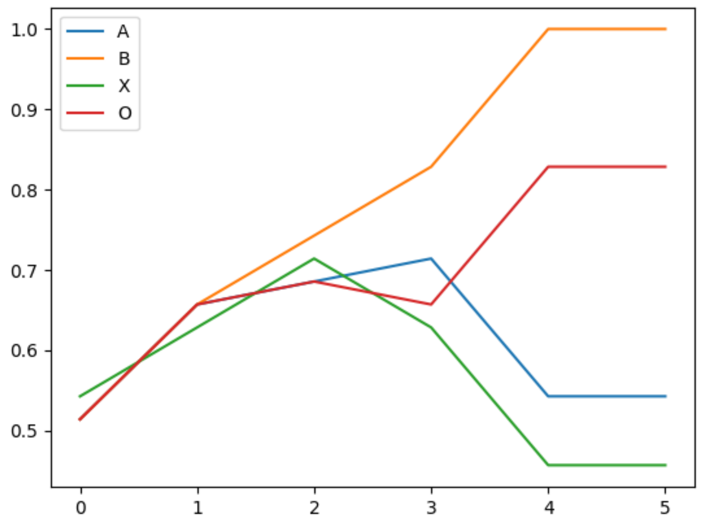
Example of network states when input was randomly generated:



Overlap graph for network states:

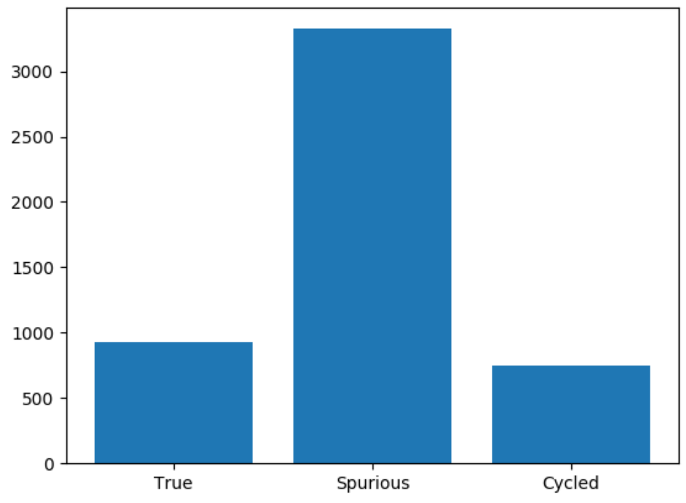
x-axis: states

y-axis: overlaps

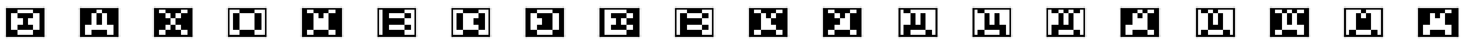


3. 5000 random binary inputs

Exact values for types of attractors: True = 925, Spurious = 3325, Cycle = 750



Most frequent attractors: we merged attractors different only in sign, code in atractor_equality(a1, all_others_already_seen), fullsize graph attached in zip. Most frequent attractor is on the left side, second most frequent is “A”.



4. Code base and file structure – important

What does code printout:

1. Actual run : generating from patterns, generating random input….
2. Distribution of attractors as list [True, Spurious, Cycle]
   Most of the attractors are spurious ones.
3. In run with 5000 random inputs, frequency of most common atractor, usually between 1800 and 1950-times.
4. Distribution of attractors for run with 5000 random inputs.

Which graphs are where:

1. Generating from patterns : “x_noise_in_patterns” folders.
2. 5 random input : “5 random inputs” folder.
3. Graphs for 5000 inputs run : “run5000” folder.

State of code:

New plotting is commented out for speed. Only for 5000 binary inputs we plot new graphs directly into directory.