Computational Neuroscience

AY 2022-23

Laboratory Assignment 2 (LAB2)
Hopfield network

General Information

Solve the following assignment, whose completion is required to access the oral examination. Upload the assignments all-together in the Moodle platform of the course (once you have completed all the labs, not only this single one) as a compressed folder including one subfolder for each laboratory.

The subfolder for this lab should be called "LAB2_2" and should include the scripts and the other files as requested in the assignment below. You can organize the code as you wish, implementing all the helper functions that you need, provided that these are included in the subfolder and are appropriately called in the scripts.

Bonus track assignments are meant to be for those who finish early, but they are not formally required for completing the Lab Assignment.

Detailed instructions for submissions are given in the lecture note "Intro to the course".

The recommended programming language is MATLAB, but you are allowed to use Python (e.g., with NumPy, or TensorFlow).
If you use MATLAB, you will provide:

- the source files as a collection of the scripts (.m files) used to solve the assignment, or as a live script (.mlx file) with all the necessary code.
- the output data as a collection of binary .mat files, or in a single .mat file.

If you use Python, you will provide:

- the source files in one of these forms:
  - a collection of the scripts (.py files) used to solve the assignment, with a main script that calls all the necessary other scripts;
  - a Jupyter nootebook (.ipynb file) with all the necessary code.
- the output data as a collection of pickle or json files, or in a single pickle/json file
  - in this case it is required to include a script to load the saved data in the chosen format.

In both cases, the required figures can be provided in .pdf files or in a graphics format (e.g., png, jpg). When multiple images are required, you are allowed to generate a single file including multiple subplots (or pages, if appropriate).

Supporting Material

Lecture’s slides

Hopfield networks lecture “Part 2 Lecture 2”

Matlab documentation

Matlab User’s Guide https://www.mathworks.com/help/index.html
Matlab documentation using the *help* command

Python documentation

https://www.learnpython.org/
https://numpy.org/
https://matplotlib.org/

Assignment 1: Small Image Dataset

The assignment consists in the following points:

1) Download the data for this assignment, you can find it in the file lab2_2_data.zip.
2) Implement a function that, given an input vector *p* whose elements are +1s and -1s, and a scalar value *d* (within 0.05 and 1) returns a perturbed/distorted version of the input vector, flipping (i.e., changing the sign) of the d*100 % its elements. You can find an example of such a function, implemented in MATLAB, in “distort_image.m” file (in the archive download at point 1)). You can interpret the input *p* as ain input image, and *d* as the percentage of distortion to apply.
3) Implement all the code that is required for the different stages in the operation of a Hopfield network, i.e. the storage phase (learning) and the retrieval phase (initialization, iteration until convergence, outputting).
   Note: in the code also include the computation of the overlap functions (with respect to the training patterns) and of the energy function.
4) Use the 3 input vectors p0, p1 and p2 (from the corresponding csv files in the archive downloaded in point 1)) to train the Hopfield network.
5) Generate distorted versions of the 3 patterns using the function obtained at point 2). For each pattern p_i, generate 3 distorted versions, with percentage of distortion 0.05, 0.1 and 0.25. E.g. (using the provided MATLAB file), d_i_1 = distort_image(p_i,0.05); d_i_2 = distort_image(p_i,0.1); d_i_3 = distort_image(p_i,0.25);

   In this step, you will have generated a total number of 9 patterns.

6) Feed the trained Hopfield network with the 9 input patterns generated in the previous point. For each of the 9 input patterns:
   - plot the energy as a function of time;
   - plot the overlap computed for each one of the training patterns, as a function of time; suggestion: use the same figure for plotting all the overlaps
   - plot the reconstructed image and compute a measure of discrepancy (choose the measure of discrepancy you prefer) between the reconstructed image and the optimal corresponding memory (add this result as title caption to the figure).

   Note: plots of the energy and overlap functions should be produced considering the evolution of the state of the network per each state update (i.e., the resolution is on the neuron update, not on the epochs)

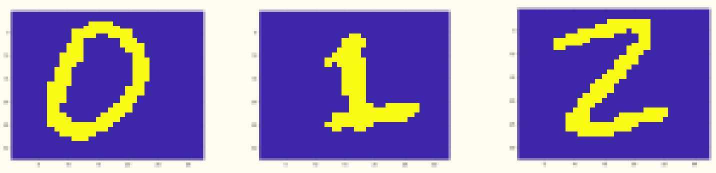
The output of the assignment should then consist in the following

- The source code.
- The plots generated in point 6) (please use a significant name for each file, e.g. “distorted_0_1_energy.png”, “distorted_0_1_overlap.png”, “distorted_0_1_reconstructed.png”)

Further notes

Some hints/notes on the manipulation of the images in the input patterns (for this assignment):

- each input pattern represents a 32x32 bitmap (+1/-1)
- you can reshape a 1024 vector to a 32×32 matrix (e.g., in MATLAB by img = reshape(vec,32,32);)
- you can then visualize the data as an image (e.g., in MATLAB with command imagesc(img);)



Bonus Track Assignment # 1 – Synthetic data

Use the code for the implementation of the Hopfield network (written for Assignment1) and train a new Hopfield network to store the following input patterns

p1 = [-1, -1, +1, -1, +1, -1, -1, +1];

p2 = [-1, -1, -1, -1, -1, +1, -1, -1];

p3 = [-1, +1, +1, -1, -1, +1, -1, +1];

Use an implementation without the input bias.

Check if the network has effectively stored the 3 patterns as fixed points.

Now consider the following distorted inputs

p1d=[+1, −1, +1, −1, +1, −1, −1, +1];

p2d=[+1, +1, −1, −1, −1, +1, −1, −1];

p3d=[+1, +1, +1, −1, +1, +1, −1, +1];

Feed them to the network and apply the activation update until convergence.

P1) Did all the patterns converge to the appropriate stored memory? Store the activations of the network neurons (as a function of time) in a matrix (e.g. Nneurons x Ntimes). Once the network has converged, plot such activations (as a function of time) on the same figure. Note that for these plots, you have the freedom to choose your preferred view, provided that it is explained sufficiently (e.g., through comments in the code and legends and titles in the images).

The output of the assignment should then consist in the following

- The source code.
- The matrices of neurons' activations generated at point P1) (i.e., 3 matrices);
- The plots generated at point P1) (i.e., 3 plots); please, use significant names for the files.

Bonus Track Assignment # 2 – How much can you remember?

In this bonus track exercise we study how many digits our Hopfield Network of Assignment 1 can store and recover.

To do so, you can train the Hopfield Network incrementally. The update rule is the following:

$$W^{(new)} = W^{(old)} + \frac{1}{N}\xi\,\xi^T ,$$

where ξ is the new pattern to be stored. Remember to add the *input bias* when computing the activations of a neuron, and to remove self-connections among neurons.

Load the 10 digits from the *lab2_2_alldigits.zip* archive file (there is one csv file for each of the input patterns). Then, proceed as follows:

1) Learn the first 3 digits, and generate their noisy version with a percentage of distortion of 0.05.
2) Now repeat:
    a. Compute the average discrepancy of all the patterns used so far.
    b. Compute the overlap for each of them over "time" with respect to the original image.
    c. Add the next pattern (proceed in order) to the Hopfield Network, and generate its distorted version.

Plot the average discrepancy with as a function of the number of stored patterns. Is there a drop in reconstruction performance at some point? How bad is it?

Finally, plot the overlaps you computed every time you added a pattern. Is there any of them which is better recalled than the others?

The output of the assignment should then consist in the following

- The source code.
- One plot for the average discrepancy
- 8 figures with the plots of the overlaps (the first one with digits {1,2,3}, the last one with digits {1,2,…10})