CS F433 Computational Neuroscience
Programming Assignment

(Weightage: 10% of final grade; 30 marks)

Deadline: Thursday, October 5 2023, 10pm.

In this programming assignment, you will write code to train your own Hopfield Network from scratch, to store memories, following which you will run some experiments on it.

You will use 60x60px black & white images in the PBM format. PBM is an ASCII format to store binary images, which are also convenient to read and write to by writing your own I/O code to do file handling. Look up the details on the format. PBM files can be converted/displayed by most image processing/handling programs or online.
Build a dataset of 25 PBM images of size 60x60px. Feel free to download images from the internet or use your own images; resize them to size 60x60px and convert them to the ASCII PBM format.

1. [10 marks] Set up code to read memories from PBM files and write the current state of the Hopfield Network onto a PBM file midway during updates. Use your dataset of 25 PBM images (which you will also print in your report) to initialize and train a Hopfield Network via the Hebbian Learning rule discussed in class.
2. [10 marks] Write code to corrupt a given memory from a PBM file by either (a) flipping each pixel with probability p, or (b) by "cropping it", by keeping a part of the image inside a bounding box to be identical to the original memory, and turn every pixel outside to all black or all white.
   Write code to update your Hopfield net (a) synchronously, (b) asynchronously, starting from an initial memory. In your report, take 2 initializations, (1) An image of a stored memory corrupted with probability p=0.3 and (2) An image with a bounding box of 40x40px. For the same initializations, report a sequence of asynchronous updates and synchronous updates respectively, upto convergence.
3. [10 marks] For probability p=0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8 and 0.9 respectively, take 20 initializations for each probability value, and starting from the initialization, run the network to convergence via synchronous updates, while keeping track of number of update steps used to do so. For each run, determine if convergence was to the corresponding stored memory that was corrupted. Report for each p value above, via a bar chart, the fraction of initializations that converged to the "correct" uncorrupted memory. Secondly, for each p value above, wherein the initialization converged to the correct uncorrupted memory, plot a separate histogram for the number of update steps needed for each convergence.

You may use any programming language of your choice, provided you do not use any library function for the training/simulation part of the code. You may of course use libraries or tools for e.g. I/O and plotting.

You need to submit a report (as a PDF file) with the above findings and also submit your code via a Google Form that will be provided. In your report, start by providing a brief description

(<1 page) about how you went about the implementation/simulation, and how you estimated the quantities/curves that you were asked to. After this, present the plots that you were asked above. Be sure to label the axes properly and report units. Finally, include all the code you have written in the Appendix. Be sure to comment your code well. You will also need to submit the code separately, in a form in which it can be run.

Note: This is an individual assignment and not a group assignment. While you may discuss the assignment with others in class or outside, **all code written and submitted must completely be your own, and this should also be the case with all your experiments and reports thereof**. Do not share your code with others or submit by modifying someone else's code or experiments.
Plagiarism is the act of representing another person's work as your own. It is a serious form of academic misconduct and among other things it corrupts the academic atmosphere in our courses.
All acts of plagiarism will lead to an enquiry and potential referral to a disciplinary committee. In the past, we have run sophisticated code comparison tools, to this end, and those indulging in plagiarism have faced serious consequences. Please don't do it.