PSYC 51.09: Problem Set 5

Introduction

This problem set is intended to solidify the concepts you learned about in this week's lectures and readings. *After attempting each question on your own*, you are encouraged to work together with your classmates in small groups, consult with ChatGPT or other tools, and/or to post and answer questions on the course's Canvas site.

Please upload your problem set to Canvas (as a Word or PDF file) before the due date. No late submissions will be accepted.

Readings and ungraded questions

Links to the readings and materials referenced below may be found on the course GitHub page.

1. Read Chapter 5 of *Foundations of Human Memory* (if you have not already done so). What were your thoughts on the reading? **(Ungraded)**
2. *Optional.* If you'd like to learn about *deep neural networks* (an extension of the Hopfield networks we learned about in class and in Chapter 5) watch this YouTube video: `https://tinyurl.com/kvbw872`. What'd you think? **(Ungraded)**
3. *Optional.* If you'd like to learn about how network patterns in our brains reflect our ongoing thoughts, read Owen et al. (2021). Thoughts? **(Ungraded)**
4. *Optional.* If you'd like to learn more about how we can intentionally forget, read Manning et al. (2016). **(Ungraded)**
5. *Optional.* Sievers and Momennejad (2019) propose an approach for "deleting" specific targeted memories by presenting tailored sequences of stimuli. Can you think of any interesting applications and/or implications of this work? **(Ungraded)**
6. *Optional*: submit a multiple-choice question based on the materials covered in this week's lectures, readings, and this problem set. You should calibrate the difficulty so that 60–70% of your classmates answer it correctly on an exam. If your question is chosen and you hit your target, you will receive and extra credit point on that exam. **(Ungraded)**

Graded questions

For this problem set, your job is to create your own neural network model of memory (a Hopfield network). Below are two memories, $\mathbf{m}_1$ and $\mathbf{m}_2$ that you will store in your network. Use the techniques we discussed in class (and in the book), along with the provided equations, to answer the following questions. Show your work!

$$\mathbf{m}_1 = \begin{pmatrix} +1 \\ -1 \\ -1 \\ +1 \\ -1 \\ -1 \end{pmatrix} \quad \mathbf{m}_2 = \begin{pmatrix} -1 \\ -1 \\ +1 \\ -1 \\ -1 \\ +1 \end{pmatrix} \quad \mathbf{x}_1 = \begin{pmatrix} -1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \mathbf{x}_2 = \begin{pmatrix} +1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Learning rule:

$$W(i,j) = \sum_{k=1}^{L} a_k(i) a_k(j)$$

Dynamic rule:

$$a(i) = \text{sign}\left(\sum_{j=1}^{N} W(i,j) a(j) \right)$$

1. Create a weight matrix, using Hebbian learning, that contains both $\mathbf{m}_1$ and $\mathbf{m}_2$ as stable memories.

2. For each of the partial cues, $\mathbf{x}_1$ and $\mathbf{x}_2$, the activity of the first two neurons is known. Use **asynchronous updating** to calculate the activities of the remaining four neurons (in whatever order you want). Can the network retrieve both memories? Hint: update neurons 3, 4, 5, and 6 (in any order). Then continue updating those 4 neurons until none of the values change to show that the network has stabilized.