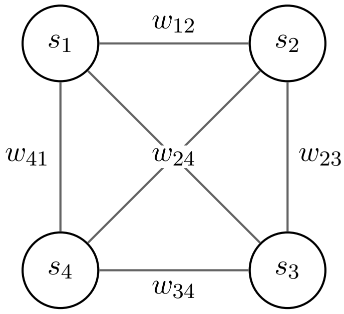
Hopfield Networks: Mathematical Problems & Solutions

Introduction

A discrete Hopfield network is a form of recurrent artificial neural network that serves as a content-addressable memory system with binary threshold nodes.

- **States:** Neurons have binary states $s_i \in \{+1, -1\}$.
- **Weights:** The connection weight between node i and j is w_{ij}. The matrix is symmetric ($w_{ij} = w_{ji}$) with no self-loops ($w_{ii} = 0$).
- **Energy Function:** The system minimizes the global energy:

$$E = -\frac{1}{2} \sum_i \sum_j w_{ij} s_i s_j$$



Problem 1: The Hebbian Learning Rule

Question:
Construct a Hopfield network to store the following two memory patterns (vectors) for a 4-neuron system:

$$\mathbf{p}^1 = [+1, +1, -1, -1]$$
$$\mathbf{p}^2 = [-1, +1, -1, +1]$$

Calculate the Weight Matrix **W** using the Hebbian rule: $w_{ij} = \sum_{\mu=1}^{P} p_i^\mu p_j^\mu$ for $i \neq j$, and $w_{ii} = 0$.

Solution:
We sum the outer products of the patterns.

$$w_{ij} = p_i^1 p_j^1 + p_i^2 p_j^2$$

Let's compute distinct pairs (matrix is symmetric):

- $w_{12} = (1)(1) + (-1)(1) = 1 - 1 = \mathbf{0}$
- $w_{13} = (1)(-1) + (-1)(-1) = -1 + 1 = \mathbf{0}$
- $w_{14} = (1)(-1) + (-1)(1) = -1 - 1 = \mathbf{-2}$
- $w_{23} = (1)(-1) + (1)(-1) = -1 - 1 = \mathbf{-2}$
- $w_{24} = (1)(-1) + (1)(1) = -1 + 1 = \mathbf{0}$
- $w_{34} = (-1)(-1) + (-1)(1) = 1 - 1 = \mathbf{0}$

The resulting Weight Matrix **W** is:

$$\mathbf{W} = \begin{pmatrix} 0 & 0 & 0 & -2 \\ 0 & 0 & -2 & 0 \\ 0 & -2 & 0 & 0 \\ -2 & 0 & 0 & 0 \end{pmatrix}$$

Problem 2: Testing Stability of Pattern 1

Question:
Verify if pattern $\mathbf{p}^1 = [+1, +1, -1, -1]$ is a stable state (a fixed point) of the network defined in Problem 1.
Hint: A state **s** *is stable if sign*$(\mathbf{Ws}) = \mathbf{s}$.

Solution:
We compute the net input vector $\mathbf{h} = \mathbf{W}\mathbf{p}^1$:

$$\begin{pmatrix} 0 & 0 & 0 & -2 \\ 0 & 0 & -2 & 0 \\ 0 & -2 & 0 & 0 \\ -2 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} +1 \\ +1 \\ -1 \\ -1 \end{pmatrix} = \begin{pmatrix} (0)(1) + (0)(1) + (0)(-1) + (-2)(-1) \\ (0)(1) + (0)(1) + (-2)(-1) + (0)(-1) \\ (0)(1) + (-2)(1) + (0)(-1) + (0)(-1) \\ (-2)(1) + (0)(1) + (0)(-1) + (0)(-1) \end{pmatrix} = \begin{pmatrix} +2 \\ +2 \\ -2 \\ -2 \end{pmatrix}$$

Now, apply the sign function (where $\text{sign}(x) = +1$ if $x \geq 0$, else -1):

$$\text{sign}(\mathbf{h}) = [+1, +1, -1, -1]$$

Since $\text{sign}(\mathbf{W}\mathbf{p}^1) = \mathbf{p}^1$, the pattern is **stable**.

Problem 3: Asynchronous Update Dynamics

Question:
Suppose the network is in a corrupted state $\mathbf{s} = [-1, +1, -1, -1]$ (Note: s_1 is flipped compared to $\mathbf{p}^1$).
Perform an asynchronous update on **Neuron 1**. Does the network recover the memory?

Solution:
The update rule for neuron i is: $s_i(t+1) = \text{sign}\left(\sum_j w_{ij} s_j(t)\right)$.
Calculate the input for Neuron 1 ($i = 1$):

$$h_1 = \sum_{j=1}^{4} w_{1j} s_j = w_{12} s_2 + w_{13} s_3 + w_{14} s_4$$

Using weights from Problem 1 and current state $\mathbf{s}$:

$$h_1 = (0)(+1) + (0)(-1) + (-2)(-1)$$

$$h_1 = 0 + 0 + 2 = +\mathbf{2}$$

Update state s_1:

$$s_1(t+1) = \text{sign}(+2) = +\mathbf{1}$$

The new state becomes $[+1, +1, -1, -1]$, which is exactly $\mathbf{p}^1$. The network has successfully corrected the error.

Problem 4: Energy Calculation

Question:
Calculate the Energy E of the stable state $\mathbf{p}^1 = [+1, +1, -1, -1]$.

$$E = -\frac{1}{2} \sum_i \sum_j w_{ij} s_i s_j$$

Alternatively, in vector form: $E = -\frac{1}{2} \mathbf{s}^T \mathbf{W} \mathbf{s}$.

Solution:
We calculated $\mathbf{h} = \mathbf{W}\mathbf{s} = [+2, +2, -2, -2]^T$ in Problem 2. Now compute the dot product $\mathbf{s}^T \mathbf{h}$:

$$\mathbf{s} \cdot \mathbf{h} = (1)(2) + (1)(2) + (-1)(-2) + (-1)(-2)$$

$$= 2 + 2 + 2 + 2 = 8$$

Finally, apply the $-\frac{1}{2}$ factor:

$$E = -\frac{1}{2}(8) = -\mathbf{4}$$

Problem 5: Stability of Pattern 2

Question:
Without re-calculating the entire matrix multiplication, verify if the second pattern $\mathbf{p}^2 = [-1, +1, -1, +1]$ is stable. Check the alignment of the input field for each neuron.

Solution:
We need to check if h_i has the same sign as p_i^2.

$$\mathbf{h} = \mathbf{W}\mathbf{p}^2 = \begin{pmatrix} 0 & 0 & 0 & -2 \\ 0 & 0 & -2 & 0 \\ 0 & -2 & 0 & 0 \\ -2 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} -1 \\ +1 \\ -1 \\ +1 \end{pmatrix}$$

Calculating row by row:

- $h_1 = -2(p_4^2) = -2(1) = -2$. Sign is -1. (Matches p_1^2)
- $h_2 = -2(p_3^2) = -2(-1) = +2$. Sign is $+1$. (Matches p_2^2)
- $h_3 = -2(p_2^2) = -2(1) = -2$. Sign is -1. (Matches p_3^2)
- $h_4 = -2(p_1^2) = -2(-1) = +2$. Sign is $+1$. (Matches p_4^2)

Since all signs match $\mathbf{p}^2$, the pattern is **stable**.

Problem 6: Spurious States (The "Anti-Pattern")

Question:
Consider the state $\mathbf{s}_{spur} = [-1, -1, +1, +1]$, which is the exact negation of $\mathbf{p}^1$ (i.e., $-\mathbf{p}^1$). Is this "reversed" image also a stable state of the network? Explain mathematically.

Solution:
Yes, it is stable. In standard Hopfield networks with zero thresholds, if $\mathbf{s}$ is a stored pattern, then $-\mathbf{s}$ is also a stable state (often called a "spurious state").

Mathematical Proof: Let $\mathbf{s}_{spur} = -\mathbf{p}^1$. The input field is $\mathbf{h} = \mathbf{W}(-\mathbf{p}^1) = -(\mathbf{W}\mathbf{p}^1)$. From Problem 2, we know $\mathbf{W}\mathbf{p}^1$ results in a vector with the same signs as $\mathbf{p}^1$. Therefore, $-(\mathbf{W}\mathbf{p}^1)$ will have signs exactly opposite to $\mathbf{p}^1$, which matches $-\mathbf{p}^1$. Thus, $\text{sign}(\mathbf{W}\mathbf{s}_{spur}) = \mathbf{s}_{spur}$.

Problem 7: Hamming Distance

Question:
Calculate the Hamming Distance between the corrupted state $\mathbf{s} = [-1, +1, -1, -1]$ used in Problem 3 and the target memory $\mathbf{p}^1 = [+1, +1, -1, -1]$. *Note: Hamming distance is the number of bit positions in which the two vectors differ.*

Solution:
Compare $\mathbf{s}$ and $\mathbf{p}^1$ element by element:

- Position 1: $s_1 = -1$, $p_1^1 = +1$ (Mismatch)
- Position 2: $s_2 = +1$, $p_2^1 = +1$ (Match)
- Position 3: $s_3 = -1$, $p_3^1 = -1$ (Match)
- Position 4: $s_4 = -1$, $p_4^1 = -1$ (Match)

There is exactly **1 mismatch**. Therefore, the Hamming Distance is **1**.

Problem 8: Energy Landscape Dynamics

Question:
In Problem 3, the network updated from the corrupted state $\mathbf{s}_{old} = [-1, +1, -1, -1]$ to the stable state $\mathbf{s}_{new} = [+1, +1, -1, -1]$. Calculate the energy of the corrupted state $\mathbf{s}_{old}$ and verify that the energy decreased after the update.

Solution:
1. Energy of $\mathbf{s}_{old}$:

$$\mathbf{h}_{old} = \mathbf{W}\mathbf{s}_{old} = \begin{pmatrix} 0 & 0 & 0 & -2 \\ 0 & 0 & -2 & 0 \\ 0 & -2 & 0 & 0 \\ -2 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} -1 \\ 1 \\ -1 \\ -1 \end{pmatrix} = \begin{pmatrix} -2(-1) \\ -2(-1) \\ -2(1) \\ -2(-1) \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ -2 \\ 2 \end{pmatrix}$$

Dot product $\mathbf{s}_{old} \cdot \mathbf{h}_{old} = (-1)(2) + (1)(2) + (-1)(-2) + (-1)(2) = -2 + 2 + 2 - 2 = 0$.

$$E_{old} = -\frac{1}{2}(0) = \mathbf{0}$$

2. Energy of $\mathbf{s}_{new}$ (Pattern 1): From Problem 4, we know $E_{new} = -\mathbf{4}$.
Conclusion: The energy decreased from 0 to -4. This confirms the Hopfield energy minimization property.

Problem 9: Storage Capacity Estimation

Question:
A fundamental limitation of Hopfield networks is their storage capacity. For a network with $N = 100$ neurons, estimate the maximum number of random patterns (P_{max}) that can be stored and retrieved with a small error probability. Use the standard approximation $P_{max} \approx 0.138N$.

Solution:
Using the statistical capacity formula derived by Amit, Gutfreund, and Sompolinsky:

$$P_{max} \approx 0.138 \times N$$

Given $N = 100$:

$$P_{max} \approx 0.138 \times 100 = 13.8$$

Therefore, the network can reliably store approximately **13 to 14** random patterns. Storing more patterns will likely result in "blackout" where the network cannot retrieve memories correctly due to excessive crosstalk noise.

Problem 10: Synchronous Update Instability

Question:
Unlike asynchronous updates, **synchronous** updates (updating all neurons at once) do not guarantee energy minimization and can lead to oscillation. Consider a state $\mathbf{s}_{sync} = [-1, -1, -1, -1]$ for the network in Problem 1. Perform one synchronous update step. What is the next state? If you perform a second step, what happens?

Solution:
Step 1: Calculate input **h** for the current state $\mathbf{s} = [-1, -1, -1, -1]$.

$$\mathbf{h} = \mathbf{Ws} = \begin{pmatrix} -2(-1) \\ -2(-1) \\ -2(-1) \\ -2(-1) \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 2 \\ 2 \end{pmatrix}$$

Apply sign function simultaneously:

$$\mathbf{s}_{next} = [\text{sign}(2), \text{sign}(2), \text{sign}(2), \text{sign}(2)] = [+\mathbf{1}, +\mathbf{1}, +\mathbf{1}, +\mathbf{1}]$$

Step 2: Now update from $[+1, +1, +1, +1]$.

$$\mathbf{h} = \mathbf{W}[1, 1, 1, 1] = \begin{pmatrix} -2(1) \\ -2(1) \\ -2(1) \\ -2(1) \end{pmatrix} = \begin{pmatrix} -2 \\ -2 \\ -2 \\ -2 \end{pmatrix}$$

Apply sign function:

$$\mathbf{s}_{next+1} = [-\mathbf{1}, -\mathbf{1}, -\mathbf{1}, -\mathbf{1}]$$

Conclusion: The network is trapped in a limit cycle of length 2 (oscillation), blinking between all $+1$ and all -1. It does not settle into a stable state.
