HOPFIELD/ATTRACTOR NETWORK

1. Hopfield Network:

- **Core Concept**: A type of Recurrent Neural Network (RNN) invented by John Hopfield in 1982. Its primary purpose is to function as an associative memory (or content-addressable memory).
- Function: It stores patterns (like images or vectors) and can retrieve a complete pattern when given a partial or noisy version of it. The network "associates" the incomplete input with the closest stored memory.
- Goal: The network's dynamics are designed to evolve from an initial state (the noisy input) to a stable state that represents a stored memory. This process minimizes an associated "energy" function.

2. Architecture

- Structure: A single layer of fully interconnected neurons. Every neuron is connected to every other neuron.
- Neuron States: Neurons are binary threshold units, meaning their state is either +1 (active) or -1 (inactive). Sometimes 1 and 0 are used.
- Connection Weights:
  - Symmetric: The weight of the connection from neuron *i* to neuron *j* is the same as from *j* to *i* ($W_{ij}=W_{ji}$). This is critical for guaranteeing convergence.
  - No Self-Connections: A neuron is not connected to itself ($W_{ii}=0$).

3. Storing Patterns (Learning)

- Method: Learning involves setting the synaptic weights to embed the desired patterns. This is done using Hebbian Learning ("neurons that fire together, wire together").
- Process:
  - If two neurons in a pattern have the same state, their connection weight is strengthened (positive).
  - If they have different states, their connection weight is weakened (negative).
- Weight Matrix Formula: To store multiple patterns, the weight between neurons i and j is calculated by summing their states across all patterns.

$$W_{ij} = 1/M \; \Sigma \; (S_i^p \, S_j^p) \text{ where } i \neq j$$

  - S_i^p is the state (+1 or -1) of neuron *i* in pattern *p*.
  - M is the total number of patterns.

- ➔ One-Shot Learning: The entire weight matrix is calculated in a single operation, not through iterative training like backpropagation.

4. Retrieving Patterns (Recall/Convergence)

- ➔ Initialization: A noisy or partial pattern is presented to the network, setting the initial state of the neurons.
- ➔ Update Process: The network updates its neurons' states iteratively until it reaches a stable configuration. The update process is asynchronous, meaning neurons are updated one at a time (often in a random order).
- ➔ Update Rule:
  1. Calculate Activation: For a chosen neuron i, calculate its activation, which is the weighted sum of the states of all other neurons.

     $$\mathbf{ai(t)} = \Sigma\, W_{ij} S_j\,(t)$$

  2. Apply Threshold: The neuron's new state is determined by the sign of its activation.

     $$\mathbf{si(t+1) = +1 \text{ if } ai(t) \geq 0, -1 \text{ if } ai(t) < 0}$$

     Essentially, if the weighted sum of its neighbors' inputs is positive, the neuron becomes active (+1), otherwise it becomes inactive (-1).
- ➔ Convergence: This process continues until no more neurons change their state. This final, stable state is the retrieved memory.

5. The Energy Function

- ➔ Concept: A Hopfield Network has an associated energy function (a Lyapunov function). The network's state always evolves to minimize this energy.
- ➔ Landscape: Stored memories correspond to local minima in the energy landscape. The recall process is like a ball rolling downhill on this landscape until it settles in a valley (a stored memory).
- ➔ Energy Formula:

  $$\mathbf{E} = -1/2 \sum W_{ij}\, S_i S_j$$

- ➔ Guarantee of Convergence: The symmetric weights (wij=wji) and the asynchronous update rule guarantee that the energy never increases. Therefore, the network must eventually reach a stable, low-energy state.

6. Characteristics & Limitations

- Strengths:
  - Associative Memory: Excellent for pattern completion and noise reduction.
  - Robustness: Can retrieve memories from highly distorted inputs.

- Limitations:
  - Limited Storage Capacity: Can only reliably store approximately $0.14 \times N$ patterns, where N is the number of neurons. Overloading the network leads to errors.
  - Spurious Attractors: The network can converge to stable states that are not any of the stored patterns. These are false memories that are mixtures or combinations of the true patterns.
  - Binary Patterns: The classic model is designed for binary (+1/-1) inputs.

7. Application to MNIST

- ➔ Goal: Given a distorted handwritten digit from the MNIST dataset, the network should output the correct, clean digit.
- ➔ Setup:
  - Neurons: An MNIST image is 28x28 pixels, so the network will have $28 \times 28 = 784$ neurons. Each neuron corresponds to one pixel.
  - Binarization: Pixel values (0-255) must be converted to binary states (+1 for white/lit, -1 for black/unlit) using a threshold.
  - Storing: Store one or more "prototype" images for each digit (0-9) by calculating the weight matrix using the Hebbian rule.
  - Testing: Present a corrupted MNIST digit to the network and let it iterate until it converges to the closest stored prototype.

References

- Hopfield Networks: Neural Memory Machines | Towards Data Science
- Modern Hopfield network and associative memory | by Farshad Noravesh | Medium