Hopfield Network

Table of Contents

What is a Hopfield Network?

- ➔ A Hopfield network consists of a set K of completely linked neurons without direct recurrences.
- ➔ All the nodes in a Hopfield network are both inputs and outputs, and they are fully interconnected.
- ➔ A recurrent neural network (RNN) invented by John Hopfield in 1982.

Used for:

- Associative memory: Retrieving stored patterns from partial or noisy input.
- Optimization problems: Finding the best solution among many possibilities.

Key features:

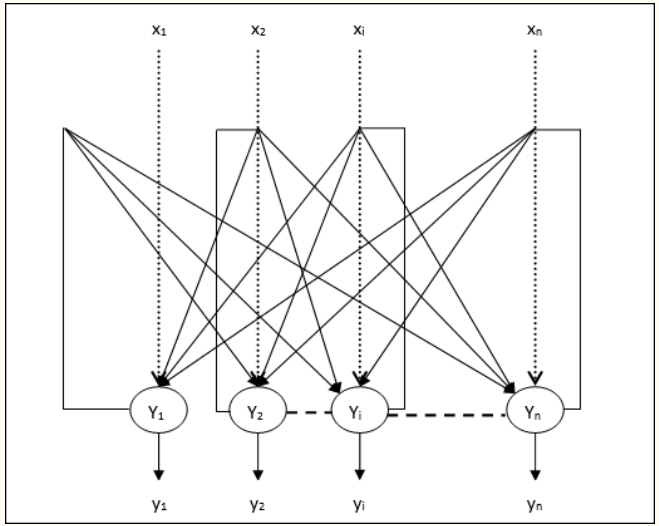
- Symmetric connections between neurons.
- Asynchronous updates of neuron states.
- Energy function that guides the network's dynamics.

Structure of Hopfield Network

Network Architecture

- A network of interconnected neurons.
- Each neuron is connected to every other neuron.
- Connections have weights (w_{ij}) that can be positive or negative.
- Weights are symmetric with no self-connections: $w_{ij} = w_{ji}$ and $w_{ii} = 0$.
- Neurons have two states: +1 (firing) or -1 (resting).

Structure of Hopfield Network...



- This model consists of neurons with one inverting and one non-inverting output.
- The output of each neuron should be the input of other neurons but not the input of self.
- Weight/connection strength is represented by $\mathbf{w_{ij}}$.
- Connections can be excitatory as well as inhibitory. It would be excitatory, if the output of the neuron is same as the input, otherwise inhibitory.
- Weights should be symmetrical, i.e. $\mathbf{w_{ij} = w_{ji}}$

The output from $\mathbf{Y_1}$ going to $\mathbf{Y_2}$, $\mathbf{Y_i}$ and $\mathbf{Y_n}$ have the weights $\mathbf{w_{12}}$, $\mathbf{w_{1i}}$ and $\mathbf{w_{1n}}$ respectively.

Similarly, other arcs have the weights on them.

Training Algorithm

During training of discrete Hopfield network, weights will be updated. As we know that we can have the binary input vectors as well as bipolar input vectors. Hence, in both the cases, weight updates can be done with the following relation

For Binary input patterns (*NB: Bipolar inputs like [-1,1] also available*)

For a set of binary patterns **s*p*, p = 1 to P**
Here, **s*p* = s_1p, s_2p,..., s_ip,..., s_np**

Weight Matrix is given by

$$W_{ij} = \sum_{p=1}^{P} [2V^p_i - 1][2V^p_j - 1] \text{ for } i \neq j$$

$$\text{but with } W_{ij} = 0.$$

Activation

$V_i in = \sum W_{ji} V_j$

$V_i = 1$ if $V_i in >= 0$

else $V_i = 0$

*Note that if you only have **one pattern**, this equation deteriorates to:*

$$W_{ij} = \sum [2v_i - 1][2v_j - 1] \text{ for } i \neq j$$

Learning in Hopfield Networks

- Hebbian Learning: The weights are typically determined through Hebbian learning principles, allowing the network to store patterns.
- Weight Update Rule: For a given pattern, the weights can be updated as follows:

$$W_{ij} = \sum_{p=1}^{P} V_i^p V_j^p$$

where V^P represents the vector of the p^{th} stored pattern.

Energy Function

- **Concept:** Hopfield Networks utilize an energy function to evaluate the state of the network.
- **Purpose:** The energy function is designed to decrease over time, leading the network towards a stable state (or "attractor").

Updating Rule

- The state of a neuron is updated asynchronously:
- Uses the sign function to determine the new state.

Pattern Storage and Retrieval

- **Stores** binary patterns in the weight matrix.
- **Asynchronous Updating:** Neurons are updated one at a time, allowing the network to converge to a stable state.
- **Pattern Completion:** Given a partial input, the network can retrieve the stored pattern via the dynamics defined by its energy function.
- Acts as an associative memory system.

Applications of Hopfield Networks

- Pattern recognition and associative memory.
- Optimization problems (e.g., the traveling salesman problem).
- Image restoration and reconstruction.
- Content-addressable memory systems.

Summary

- Hopfield Networks exemplify an important class of neural network architectures providing insights into associative memory and retrieval mechanisms.
- Hopfield Networks are energy-based recurrent networks.
- Governed by energy minimization and Hebbian learning.
- Useful in pattern recognition and optimization.
- Have limitations such as limited capacity and local minima.
- **Future Directions**: Ongoing research aims to enhance their capabilities and address limitations, paving the way for advanced neural network designs.

Example

Suppose we wish to store or learn the set of states/patterns in a Hopfield network, where each pattern consists of a set of (**0 1 1 0 1**) and (**1 0 1 0 1**) attractors in the network, and then return the one that is the most similar to a given input starting at the state (**11111**).

Therefore, how to train a Hopfield network with a fixed node updating order of nodes

3, 1, 5, 2, 4, 3, 1, 5, 2, 4, etc.

In a Hopfield network, all the nodes are inputs to each other, and they're also outputs. As stated above, how it works in computation is that you put a distorted pattern onto the nodes of the network, iterate a bunch of times, and eventually it arrives at one of the patterns we trained it to know and stays there. So, what you need to know to make it work are:

- ➔ How to "train" the network? **Store a given pattern in the network**
- ➔ How to update a node in the network? **First add a weight matrix of the stored patterns weight-by-weight basis**
- ➔ How the overall sequencing of node updates is accomplished? **randomly select a neuron and update it.**
- ➔ How can you tell if you've reached one of the trained patterns? **If you go through 5 consecutive neurons without changing any of their values, then you're at an attractor, so you can stop.**

References

- Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8), 2554-2558.
- Haykin, S. (1998). Neural Networks: A Comprehensive Foundation. Prentice Hall.
- https://www.tutorialspoint.com/artificial_neural_network/artificial_neural_network_hopfield.htm