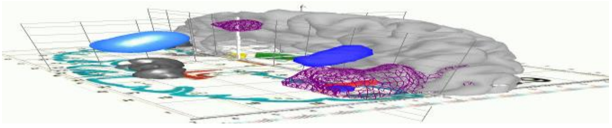


Neural Networks and Fuzzy Logic (630514)

Lecture 11

Supervised Learning in Neural Networks (Part 4)
Hopfield Networks

- Neural networks were designed by analogy with the brain. The brain's memory, however, works by association.
- Multilayer neural networks trained with the back-propagation algorithm are used for pattern recognition problems. However, ***to emulate the human memory's associative characteristics we need*** a different type of network: a ***recurrent neural network***.
- A recurrent neural network has **feedback loops** from its outputs to its inputs. The presence of such loops has a profound impact on the learning capability of the network.
- **John Hopfield** in 1982 formulated the physical principle of storing information in a dynamically stable network (**Content-Addressable Memory**).

Single-layer n-neuron Hopfield network

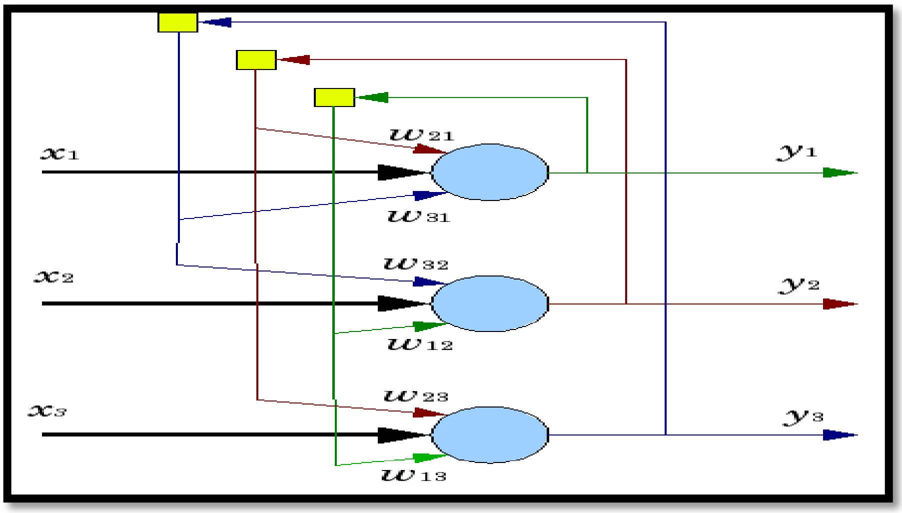
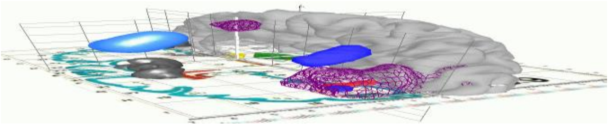


Figure 1

Common Properties and Architecture:

- Every neuron is connected to every other neuron.
- Fully connected with Symmetric weights: $w_{ij} = w_{ji}$.
- No self-loops : $w_{ii} = 0$.
- Each neuron has a single input from the outside world.
- In the Hopfield model the neurons have a binary output taking the values **–1** and **1**.
- Random updating for neurons.
- A Hopfield Network is a model of ***associative memory***. It is based on Hebbian learning but uses binary neurons.



- A Hopfield Network provides a formal model which can be analyzed for determining the ***storage capacity*** of the network.
- An associative memory can be thought as a set of attractors, each with its own basin of attraction.
- The space of all possible states of the network is called the ***configuration*** space.
- **Basins of attraction**: Division of the configuration space by stored patterns.
- Stored patterns should be attractors.

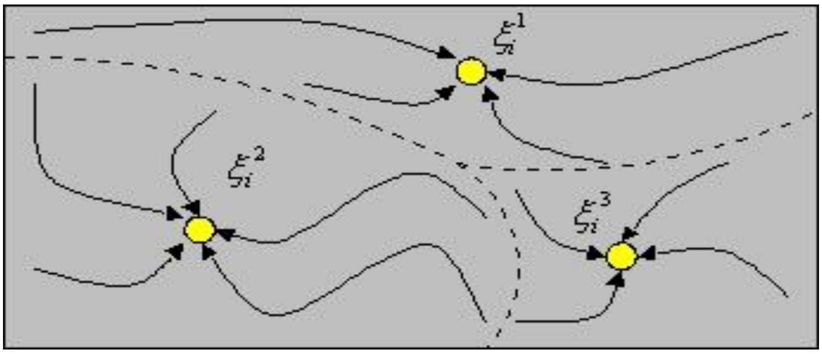


Figure 2

- Memories are attractors in state space as shown in the figure.

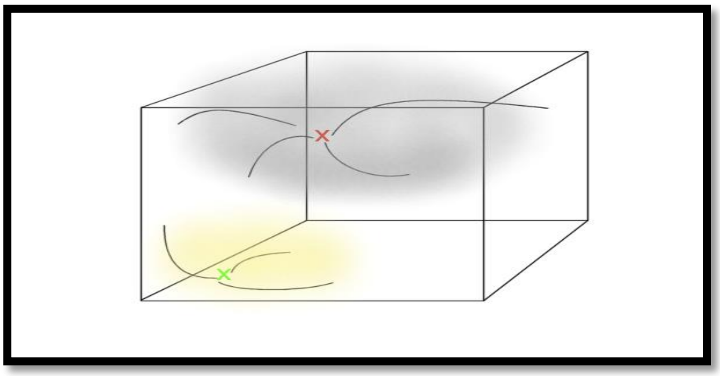
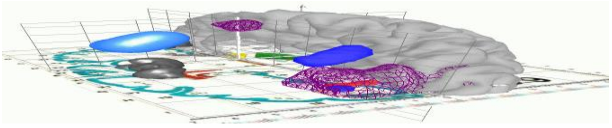


Figure 3

- The dynamics of the system carries starting points into one of the attractors as shown in the above figure.
- A Hopfield net is composed of **binary threshold units** with recurrent connections between them. Recurrent networks of **non-linear units** are generally very hard to analyze. They can behave in many different ways:
  - Settle to a stable state.
  - Oscillate.
  - Follow chaotic trajectories that cannot be predicted.
- But Hopfield realized that if the connections are **symmetric**, there is a **global energy function**.
- Each "**configuration**" of the network has energy.
- The binary threshold decision rule causes the network to settle to an energy minimum.
- Pattern recognizer.



- Hopfield networks have two applications. First, they can act as **associative memories**. Second, they can be used to solve **optimization problems**.

According to figure 1:

- The Hopfiled model starts with the **standard McCulloch-Pitts model** of a neurons with the sign activation function:

$$Y^{sign} = \begin{cases} +1, & \text{if } X > 0 \\ -1, & \text{if } X < 0 \\ Y, & \text{if } X = 0 \end{cases}$$

- The **current state** is determined by the current outputs of all neurons, y_1, y_2, . . ., y_n. Thus, for a single-layer n-neuron network, the state can be defined by the **state vector** as:

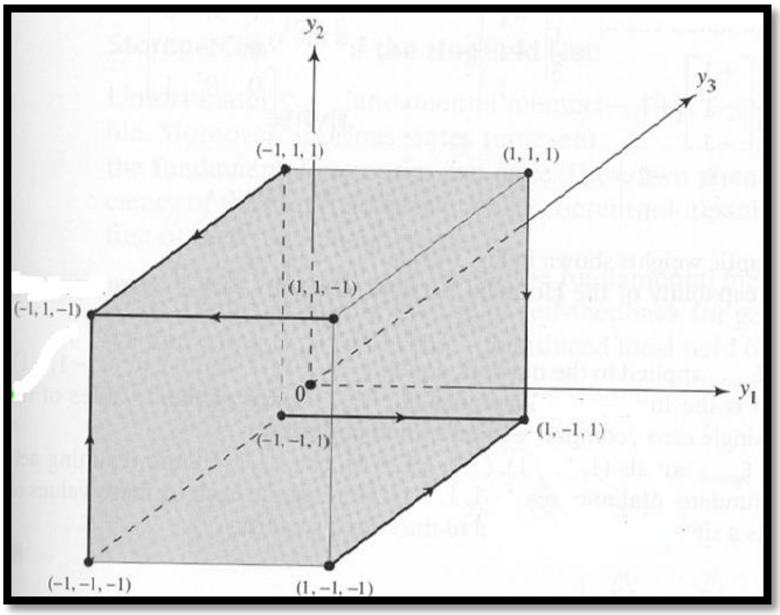
$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

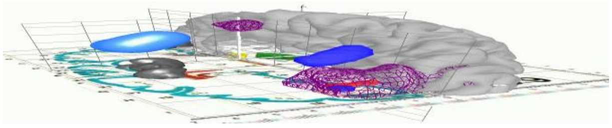
- **Synaptic weights** between neurons are represented in matrix form as follows:

$$\mathbf{W} = \sum_{m=1}^{M} \mathbf{Y}_m \mathbf{Y}_m^T - M\,\mathbf{I}$$

- Where **M** is the number of **states** to be memorized by the network, $\mathbf{Y}_m$ is the n-dimensional binary vector, **I** is **n x n** identity matrix, and superscript **T** denotes matrix transposition.

Possible states for the three-neuron Hopfield network





- The stable state-vertex is determined by the weight matrix **W**, the current input vector **X**, and the threshold matrix $\boldsymbol{\theta}$. If the input vector is partially incorrect or incomplete, the initial state will converge into the stable state-vertex after a few iterations.

Example:

- Suppose, that our network is required to memorize two states, **(1, 1, 1)** and **(-1, -1, -1).** Thus,

$$\mathbf{Y}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \mathbf{Y}_2 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{Y}_1^T = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \quad \mathbf{Y}_2^T = \begin{bmatrix} -1 & -1 & -1 \end{bmatrix}$$

where $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are the three-dimensional vectors.

- The **3 x 3** identity matrix **I** is

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- **weight matrix** calculation:

$$\mathbf{W} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \end{bmatrix} - 2 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 2 & 2 \\ 2 & 0 & 2 \\ 2 & 2 & 0 \end{bmatrix}$$

- The network is tested by the sequence of input vectors, $\mathbf{X}_1$ and $\mathbf{X}_2$, which are equal to the output (or target) vectors $\mathbf{Y}_1$ and $\mathbf{Y}_2$, respectively using the following equation:

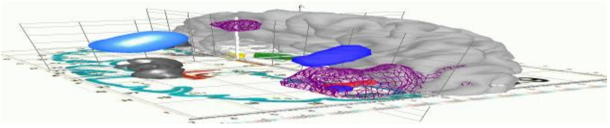
$$Y_m = sign(WX_m - \theta), \qquad m = 1, 2, \ldots, M$$

Where $\boldsymbol{\theta}$ is the threshold. The testing process include the following steps:

1) Activate the network by applying the input vector **X**.
2) Calculate the actual output vector **Y**.
3) Compare the result with the initial input vector **X**.

$$\mathbf{Y}_1 = sign\left\{ \begin{bmatrix} 0 & 2 & 2 \\ 2 & 0 & 2 \\ 2 & 2 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \right\} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\mathbf{Y}_2 = sign\left\{ \begin{bmatrix} 0 & 2 & 2 \\ 2 & 0 & 2 \\ 2 & 2 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \right\} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$



- The remaining six states are all unstable. However, stable states (also called **fundamental memories**) are capable of attracting states that are close to them.
- The fundamental memory **(1, 1, 1)** attracts unstable states **(-1, 1, 1), (1,-1, 1)** and **(1, 1, -1).** Each of these unstable states represents a single error, compared to the fundamental memory **(1, 1, 1).**
- The fundamental memory **(-1, -1, -1)** attracts unstable states **(-1, -1, 1), (-1, 1, -1) and (1, -1, -1).**
- Thus, the Hopfield network can act as an **error correction network**.