RAMA UNIVERSITY

www.ramauniversity.ac.in

FACULTY OF EGINEERING AND TECHNOLOGY

Soft Computing

LECTURE -09

Umesh Kumar Gera
Assistant Professor
Computer Science & Engineering

OUTLINE

- Concept of Hopfield Network
- Hopfield Network updating rule
- Hopfield Network as a Dynamical system
- Energy function evaluation
- Multiple Choice Question
- References

INTRODUCTION KOHONEN SELF ORGANIZING MAPS

Background of Kohonen Self-Organizing Maps

❑ Pioneered in 1982 by Finnish professor and researcher Dr. Teuvo Kohonen, a self-organising map is an unsupervised learning model, intended for applications in which maintaining a topology between input and output spaces is of importance.

❑ The notable characteristic of this algorithm is that the input vectors that are close — similar — in high dimensional space are also mapped to nearby nodes in the 2D space.

❑ It is in essence a method for dimensionality reduction, as it maps high-dimension inputs to a low (typically two) dimensional discretised representation and conserves the underlying structure of its input space.

❑A valuable detail is that the entire learning occurs without supervision i.e. the nodes are self-organising.

❑ They are also called feature maps, as they are essentially retraining the features of the input data, and simply grouping themselves according to the similarity between one another.

❑ This has a pragmatic value for visualizing complex or large quantities of high dimensional data and representing the relationship between them into a low, typically two-dimensional, field to see if the given unlabelled data has any structure to it.

STRUCTURE OF KOHONEN SELF ORGANIZING MAPS

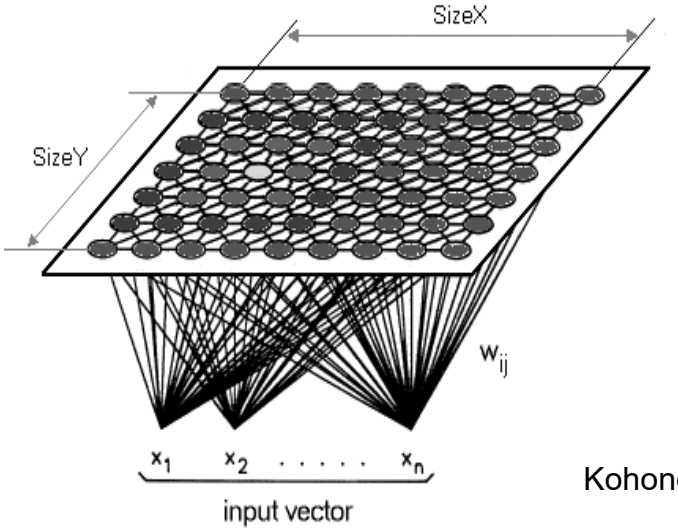
Structure of Kohonen Self-Organizing Maps

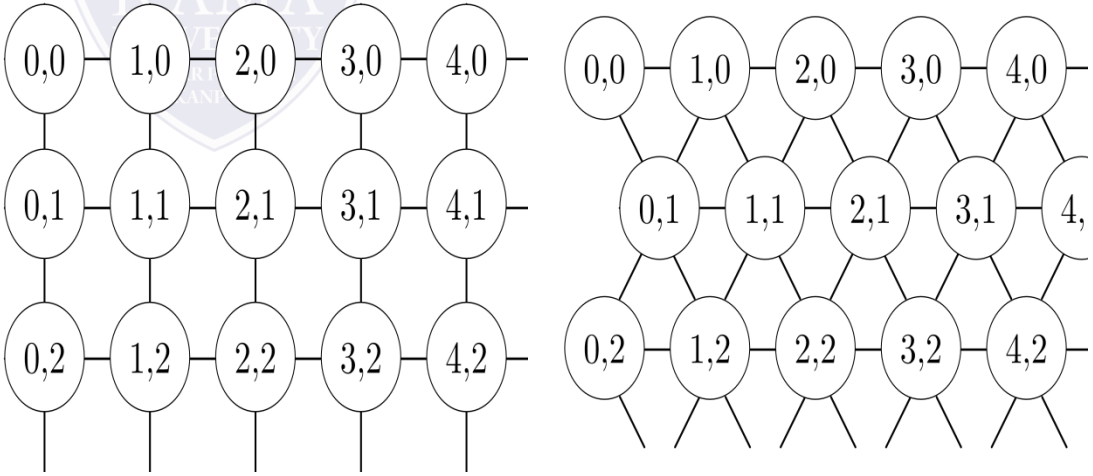
❑ A Self-Organizing Map (SOM) differs from typical ANNs both in its architecture and algorithmic properties.

❑ Firstly, its structure comprises of a single-layer linear 2D grid of neurons, instead of a series of layers.

❑ All the nodes on this grid are connected directly to the input vector, but not to one another, meaning the nodes do not know the values of their neighbors, and only update the weight of their connections as a function of the given inputs.

❑ The grid itself is the map that organizes itself at each iteration as a function of the input of the input data. As such, after clustering, each node has its own (i, j) coordinate, which allows one to calculate the Euclidean distance between 2 nodes by means of the Pythagorean theorem.

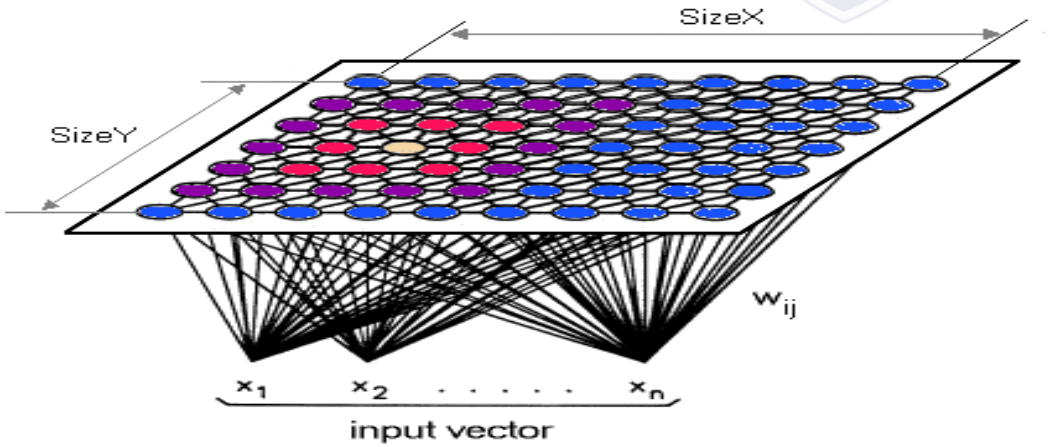




Kohonen network's nodes can be in a rectangular (left) or hexagonal (right) topology.

Property of Kohonen Self-Organizing Maps

- A Self-Organising Map, additionally, uses competitive learning as opposed to error-correction learning, to adjust it weights.
- This means that *only a single node* is activated at each iteration in which the features of an instance of the input vector are presented to the neural network, as all nodes compete for the right to respond to the input.
- The chosen node — the Best Matching Unit (BMU) — is selected according to the similarity, between the current input values and all the nodes in the grid.
- The node with the smallest Euclidean difference between the input vector and all nodes is chosen, *along with its neighboring nodes* within a certain radius, to have their position slightly adjusted to match the input vector.
- By going through all the nodes present on the grid, the entire grid eventually matches the complete input dataset, with similar nodes grouped together towards one area, and dissimilar ones separated.



A Kohonen model with the BMU in yellow, the layers inside the neighborhood radius in pink and purple, and the nodes outside in blue.

VARIABLES OF KOHONEN SELF ORGANIZING MAPS

Kohonen Self-Organizing Maps variables

- t is the current iteration
- n is the iteration limit, i.e. the total number of iterations the network can undergo
- λ is the time constant, used to decay the radius and learning rate
- i is the row coordinate of the nodes grid
- j is the column coordinate of the nodes grid
- d is the distance between a node and the BMU
- w is the weight vector
- w_ij(t) is the weight of the connection between the nodes i,j in the grid, and the input vector's instance at the iteration t
- x is the input vector
- x(t) is the input vector's instance at iteration t
- α(t) is the learning rate, decreasing with time in the interval [0,1], to ensure the network converges.
- β_ij(t) is the neighborhood function, monotonically decreasing and representing a node i, j's distance from the BMU, and the influence it has on the learning at step t.
- σ(t) is the radius of the neighborhood function, which determines how far neighbor nodes are examined in the 2D grid when updating vectors. It is gradually reduced over time.

ALGORITHM OF KOHONEN SELF ORGANIZING MAPS

Kohonen Self-Organizing Maps Algorithm steps

1. Initialize each node's weight w_ij to a random value
2. Select a random input vector x_k
3. Repeat point 4. and 5. for all nodes in the map:
4. Compute Euclidean distance between the input vector x(t) and the weight vector w_ij associated with the first node, where t, i, j = 0.
5. Track the node that produces the smallest distance t.
6. Find the overall Best Matching Unit (BMU), i.e. the node with the smallest distance from all calculated ones.
7. Determine topological neighborhood β_ij(t) its radius σ(t) of BMU in the Kohonen Map
8. Repeat for all nodes in the BMU neighborhood: Update the weight vector w_ij of the first node in the neighborhood of the BMU by adding a fraction of the difference between the input vector x(t) and the weight w(t) of the neuron.
9. Repeat this whole iteration until reaching the chosen iteration limit t=n

Step 1 is the initialization phase, while step 2–9 represent the training phase.

WORKING OF HOPFIELD NETWORK

Formula used in Kohonen Self-Organizing Maps

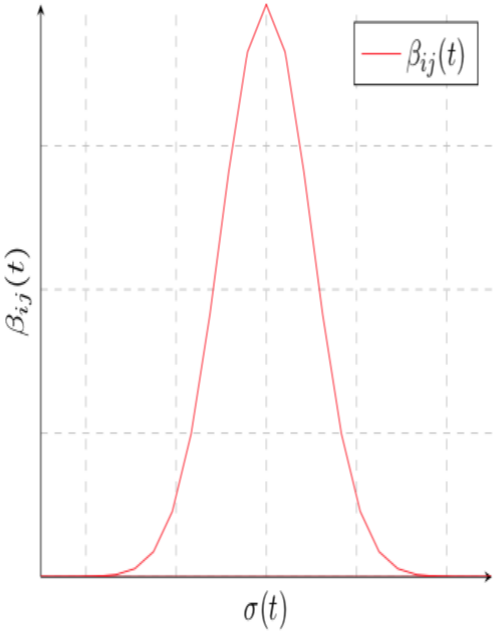
The updates and changes to the variables are done according to the following formulas:

The weights within the neighborhood are updated as:

$$w_{ij}(t+1) = w_{ij}(t) + \alpha_i(t)[x(t) - w_{ij}(t)], \quad \text{or}$$
$$w_{ij}(t+1) = w_{ij}(t) + \alpha_i(t)\beta_{ij}(t)[x(t) - w_{ij}(t)]$$

The first equation tells us that the new updated weight w_ij (t + 1) for the node i, j is equal to the sum of old weight w_ij(t) and a fraction of the difference between the old weight and the input vector x(t). In other words, the weight vector is 'moved' closer towards the input vector. Another important element to note is that the updated weight will be proportional to the 2D distance between the nodes in the neighborhood radius and the BMU.

Furthermore, the same equation 3.1 does not account for the influence of the learning being proportional to the distance a node is from the BMU. The updated weight should take into factor that the effect of the learning is close to none at the extremities of the neighborhood, as the amount of learning should decrease with distance. Therefore, the second equation adds the extra neighborhood function factor of βij(t), and is the more precise in-depth one.



WORKING OF HOPFIELD NETWORK

The radius and learning rate are both similarly and exponentially decayed with time.

$$\sigma(t) = \sigma_0 \cdot \exp(\frac{-t}{\lambda}), \quad \text{where } t = 1, 2, 3 \dots n$$

$$\alpha(t) = \alpha_0 \cdot \exp(\frac{-t}{\lambda}), \quad \text{where } t = 1, 2, 3 \dots n$$

The neighborhood function's influence β_i(t) is calculated by:

$$\beta_{ij}(t) = \exp\left(\frac{-d^2}{2\sigma^2(t)}\right), \quad \text{where } t = 1, 2, 3 \dots n$$

The Euclidean distance between each node's weight vector and the current input instance is calculated by the Pythagorean formula.

$$||\vec{x} - \vec{w_{ij}}|| = \sqrt{\sum_{t=0}^{n} [\vec{x}(t) - \vec{w}_{ij}(t)]^2}$$

WORKING OF HOPFIELD NETWORK

The BMU is selected from all the node's calculated distances as the one with the smallest.

$$d = \min(||\vec{x} - \vec{w_{ij}}||) = \min(\sqrt{\sum_{t=0}^{n}[\vec{x}(t) - \vec{w}_{ij}(t)]^2})$$

WORKING OF HOPFIELD NETWORK

Energy function evaluation:

Hopfield networks have an energy function that diminishes or is unchanged with asynchronous updating.

For a given state X ∈ {−1, 1} N of the network and for any set of association weights W_{ij} with $W_{ij} = w_{ji}$ and $w_{ii} = 0$

let,

$$E = -1/2 \sum_{i,j=1}^{N} WijXiXj$$

Here, we need to update Xm to X'm and denote the new energy by E' and show that.

E'-E = (Xm-X'm) ∑i≠mWmiXi.

Using the above equation, if Xm = Xm' then we have E' = E

If Xm = -1 and Xm' = 1 , then Xm - Xm' = 2 and hm= ∑iWmiXi ? 0

Thus, E' - E ≤ 0

Similarly if Xm =1 and Xm'= -1 then Xm - Xm' = 2 and hm= ∑iWmiXi < 0

Thus, E - E' < 0.

Training the network: One pattern (K_i=0)

Suppose the vector $x^{\rightarrow} = (x_1,\ldots,x_i,\ldots,x_N) \in \{-1,1\}^N$ is a pattern that we like to store in the Hopfield network.

To build a Hopfield network that recognizes $x^{\rightarrow}$, we need to select connection weight W_{ij} accordingly.

If we select $W_{ij} = \eta X_i X_j$ for $1 \le i, j \le N$ (Here, $i \neq j$), where $\eta > 0$ is the learning rate, then the value of X_i will not change under updating condition as we illustrate below.

We have

$$hi = \sum_{j=1}^{N} WijXj = \eta \sum_{j \neq i} XiXjXj = \eta \sum_{j \neq i} Xi = \eta(N-1)X_i$$

It implies that the value of Xi, whether 1 or -1 will not change, so that $x\rightarrow$ is a fixed point.

Note that - $x\rightarrow$ also becomes a fixed point when we train the network with $x\rightarrow$ validating that Hopfield networks are sign blind

MULTIPLE CHOICE QUESTION

6. Which of the following is true for neural networks?

(i) The training time depends on the size of the network.

(ii) Neural networks can be simulated on a conventional computer.

(iii) Artificial neurons are identical in operation to biological ones.

a) All of the mentioned

b) (ii) is true

c) (i) and (ii) are true

d) None of the mentioned

7. What are the advantages of neural networks over conventional computers?

(i) They have the ability to learn by example

(ii) They are more fault tolerant

(iii)They are more suited for real time operation due to their high 'computational' rates

a) (i) and (ii) are true

b) (i) and (iii) are true

c) Only (i)

d) All of the mentioned

8. Which of the following is true?

Single layer associative neural networks do not have the ability to:

(i) perform pattern recognition

(ii) find the parity of a picture

(iii)determine whether two or more shapes in a picture are connected or not

a) (ii) and (iii) are true

b) (ii) is true

c) All of the mentioned

d) None of the mentioned

9. Which is true for neural networks?

a) It has set of nodes and connections

b) Each node computes it's weighted input

c) Node could be in excited state or non-excited state

d) All of the mentioned

10. What is Neuro software?

a) A software used to analyze neurons

b) It is powerful and easy neural network

c) Designed to aid experts in real world

d) It is software used by Neurosurgeon

REFERENCES

- https://www.sanfoundry.com/neural-networks-questions-answers-backpropagation-algorithm/
- https://www.javatpoint.com/artificial-neural-network-hopfield-network