Unit-5

Unsupervised Learning Network Paradigms

By Dr. Gopal Chandra Jana, Assistant Professor, DCSE, SSCSE

Course Page: https://www.gcjana.in/courses/shardauniversity/2502/CSA203/

Topic to be Covered: Unit 5 - Unsupervised Learning Network Paradigms

- Self-Organizing Feature Maps (SOM)
- Structure of SOM
- Training of SOM
- Topology & Neighborhood Function
- Distance Functions in SOM
- Learning Rate vs Neighborhood
- Applications of SOM
- Adaptive Resonance Theory (ART)
- Structure of ART Network
- Learning Process in ART
- ART Extensions
- Hopfield Network
- Associative Networks
- Restricted Boltzmann Machine (RBM

1. Self-Organizing Feature Maps (SOM) - (Also called **Kohonen Self-Organizing Map**, 1970s)

Concept

A **Self-Organizing Map (SOM)** is an **unsupervised neural network** used for:

- Clustering
- Dimensionality reduction
- Visualization of high-dimensional data

➢ It maps high-dimensional input data into a **low-dimensional (usually 2D) grid** while preserving **topological relationships**.

Key Idea

- Similar inputs → mapped close together
- Dissimilar inputs → mapped far apart

2. Structure of SOM

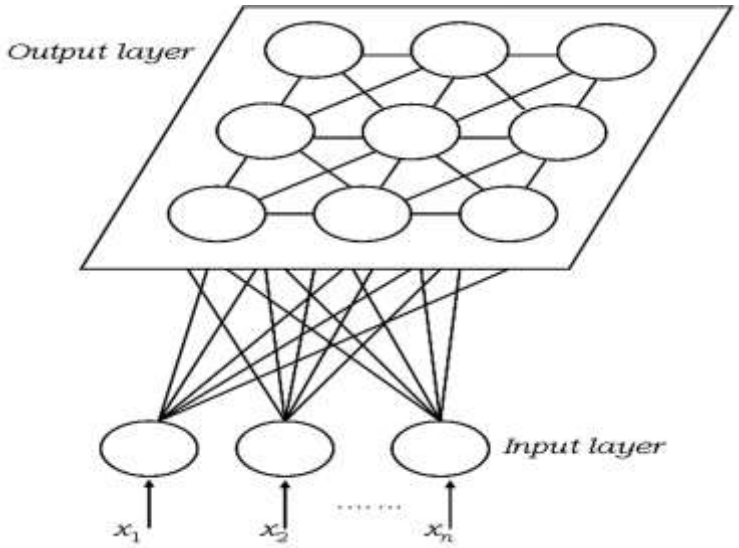
Architecture

• Input Layer: n-dimensional vector

• Output Layer: 2D grid of neurons

Each neuron has:

• Weight vector (same dimension as input)

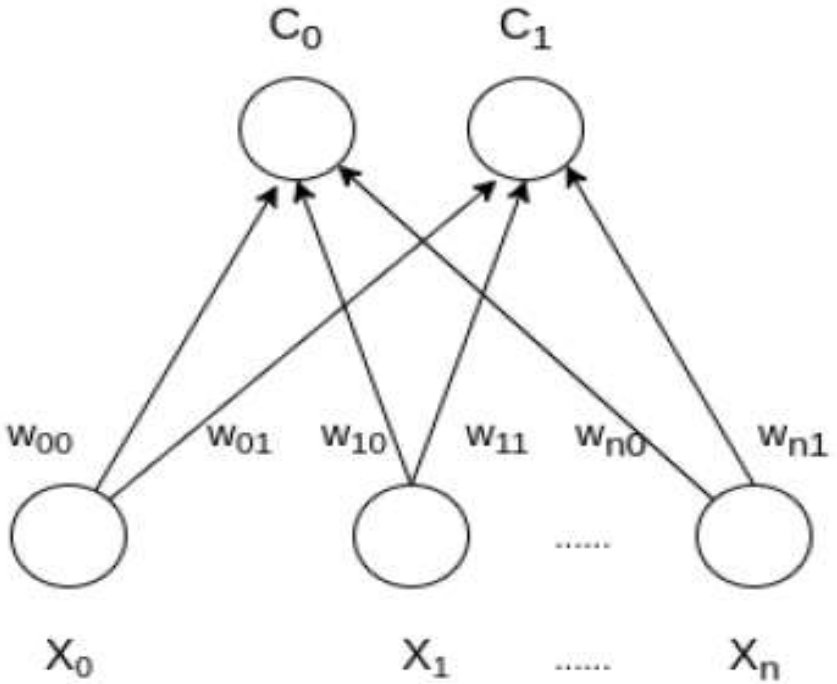


Components

1. Input vector: $X = [x_1, x_2, . . . x_n]$

2. Weight vector: $W_i = [w_{i1}, w_{i2}, . . . w_{in}]$

3. Grid structure (rectangular or hexagonal)



3. Training of SOM

❖ **Steps**

Step 1: Initialization

• Random weights

Step 2: Competition

• Find **Best Matching Unit (BMU)** using distance

➢ Common metric:
• Euclidean Distance

Step 3: Cooperation
• Neighbors of BMU are identified

Step 4: Adaptation
Update weights:

$$W_i(t+1) = W_i(t) + \alpha(t) \cdot h_{ci}(t) \cdot \big(X - W_i(t)\big)$$

Formula: $w_{ij}^{(new)} = w_{ij}^{(old)} + \alpha \cdot (x_i - w_{ij}^{(old)})$

where

- α is the learning rate
- x_i is the input feature.

❑ **Learning Phases**

• Ordering phase (large neighborhood)

• Convergence phase (small neighborhood)

4. Topology & Neighborhood Function

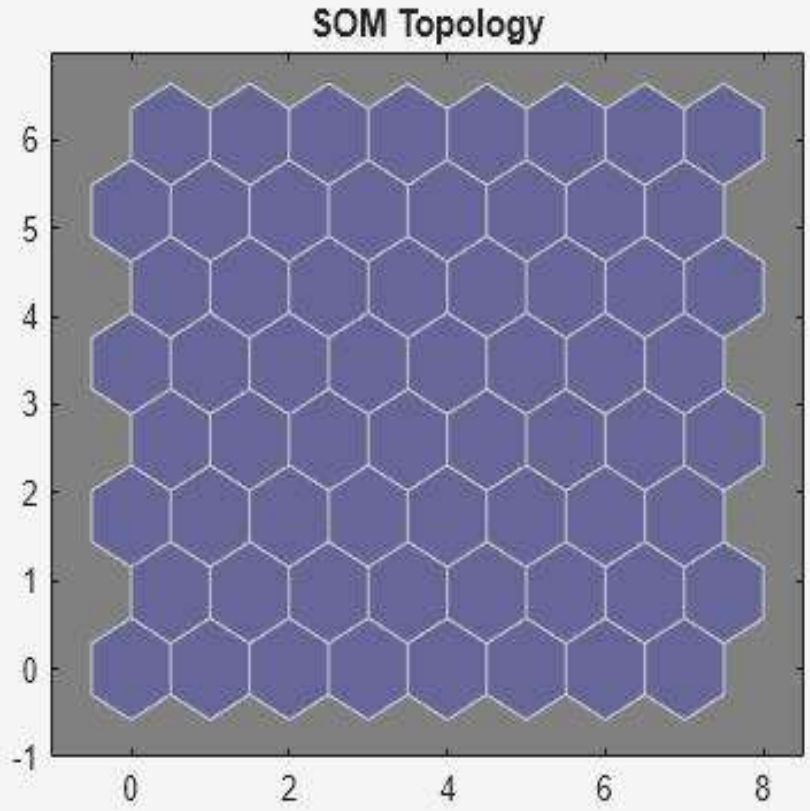
❑ Topology Types

- Rectangular grid
- Hexagonal grid

❑ Neighborhood Function $h_{ci}(t)$

Common types:

1. Gaussian
2. Bubble
3. Mexican Hat



4. Topology & Neighborhood Function

❑ Topology Types

- Rectangular grid
- Hexagonal grid

❑ Neighborhood Function $h_{ci}(t)$

Common types:

1. Gaussian
2. Bubble
3. Mexican Hat

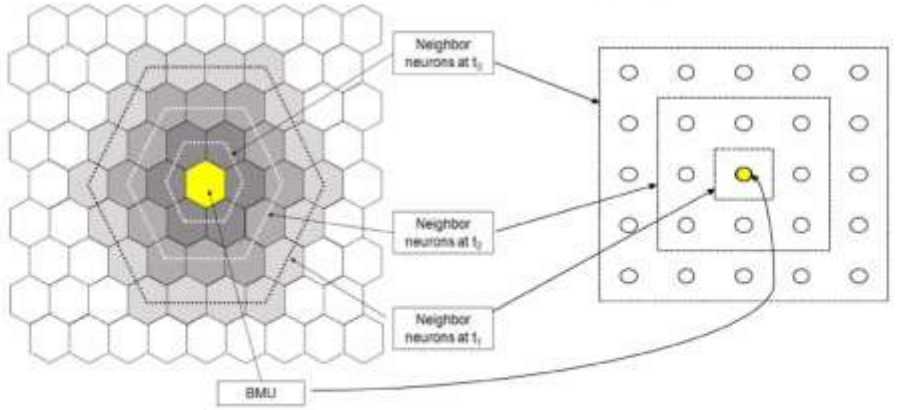
Self Organizing Maps(SOM)

- Common neighborhood function

$$h_{ci}(t) = \exp\left(-\frac{\|r_c - r_i\|^2}{2\sigma^2(t)}\right)$$

σ(t): defines the radius of the neighborhood around the BMU

- **Note:** $\|r_c - r_i\|$ is measured in the output space



4. Topology & Neighborhood Function

❑ Topology Types

- Rectangular grid
- Hexagonal grid

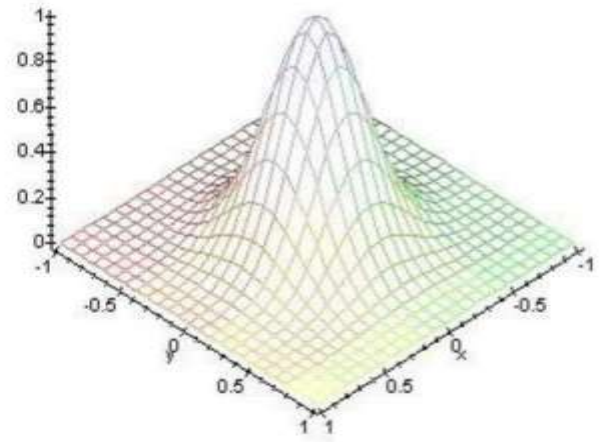
❑ Neighborhood Function $h_{ci}(t)$

Common types:

1. Gaussian
2. Bubble
3. Mexican Hat

SOM: Gaussian neighborhood function

$$h_{ci} = \exp\left(-\frac{\|r_c - r_i\|^2}{2\sigma_t^2}\right)$$



4. Topology & Neighborhood Function

❑ Topology Types

- Rectangular grid
- Hexagonal grid

❑ Neighborhood Function $h_{ci}(t)$

Common types:

1. Gaussian
2. Bubble
3. Mexican Hat

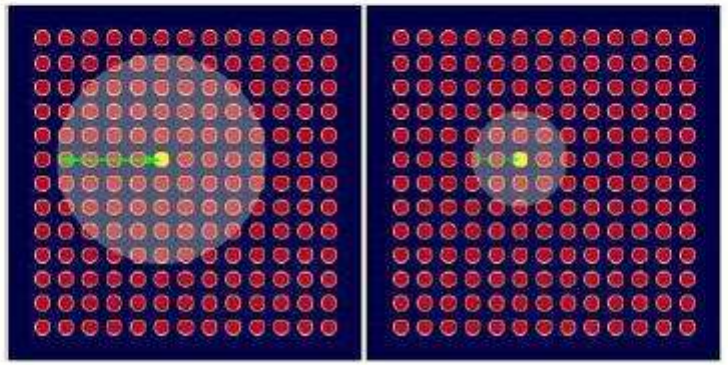
$$\sigma(t) = \sigma_0 \exp(-\frac{t}{\lambda})$$

σ_0 = *the width of lattice at time zero*

t = *the current time step*

λ = *the time constant*

The value of λ depends on σ_0 and the chosen number of iterations for algorithm.



Size of the neighborhood around the BMU shrinks

4. Topology & Neighborhood Function

❑ Topology Types

- Rectangular grid
- Hexagonal grid

❑ Neighborhood Function $h_{ci}(t)$

Common types:

1. Gaussian
2. Bubble
3. Mexican Hat

5. Distance Functions in SOM

Common distance measures:

- Euclidean distance
- Manhattan distance
- Cosine similarity

6. Learning Rate vs Neighborhood

❑ **Key Relationship**

• Learning rate $\alpha(t)$ ↓over time

• Neighborhood radius ↓ over time

❑ **Early stage:**

• Large learning rate

• Large neighborhood → global ordering

❑ **Later stage:**

• Small learning rate

• Small neighborhood → fine tuning

7. Applications of SOM

- Data visualization
- Image compression
- Pattern recognition
- Clustering in bioinformatics
- Market segmentation

8. Adaptive Resonance Theory (ART)

Concept

Proposed by **Stephen Grossberg**

ART solves:

Stability-Plasticity dilemma

•Stability → retain old knowledge

•Plasticity → learn new patterns

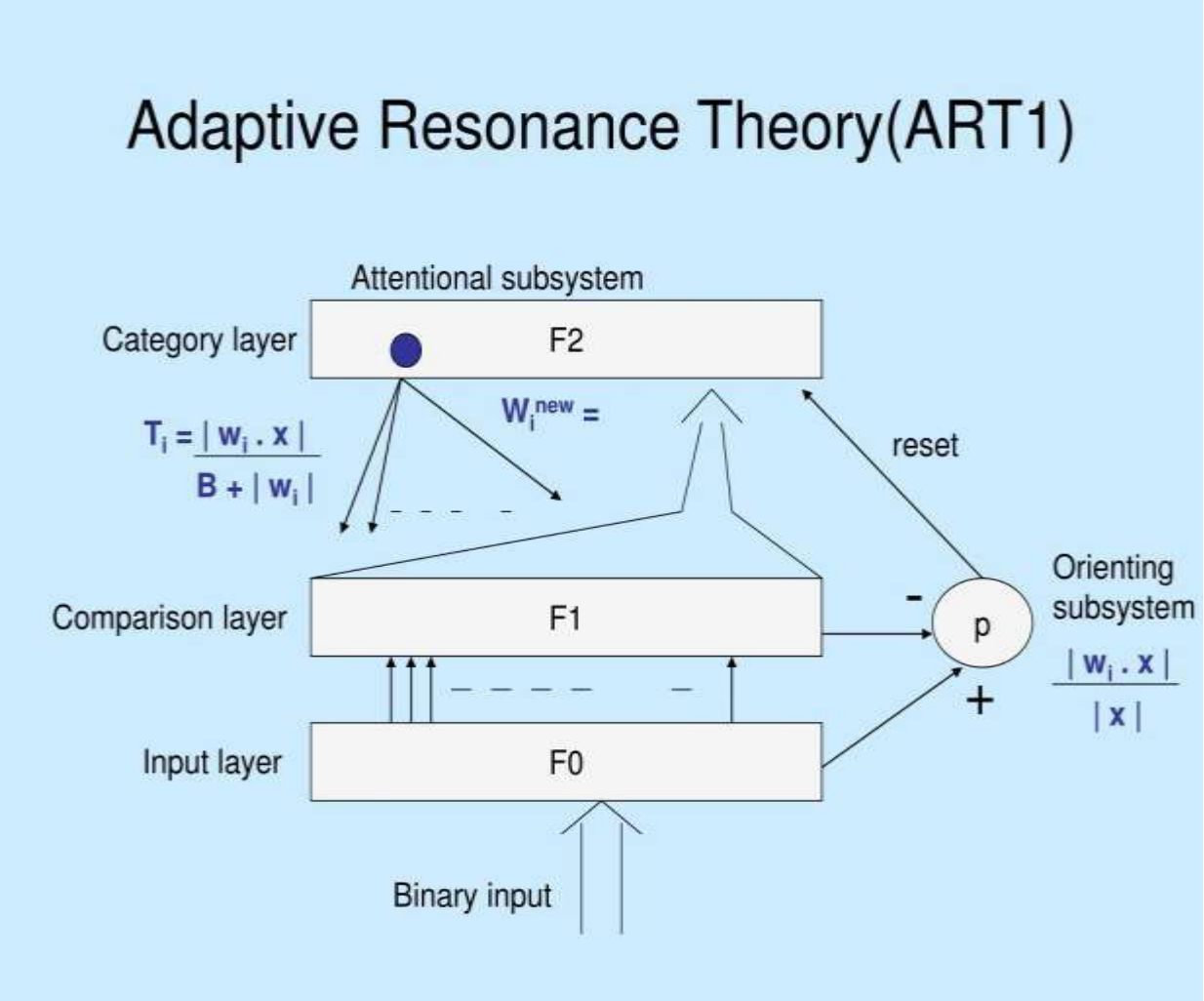
9. Structure of ART Network

Layers

1.F1 Layer → Input representation

2.F2 Layer → Category layer

3.Vigilance parameter (ρ)



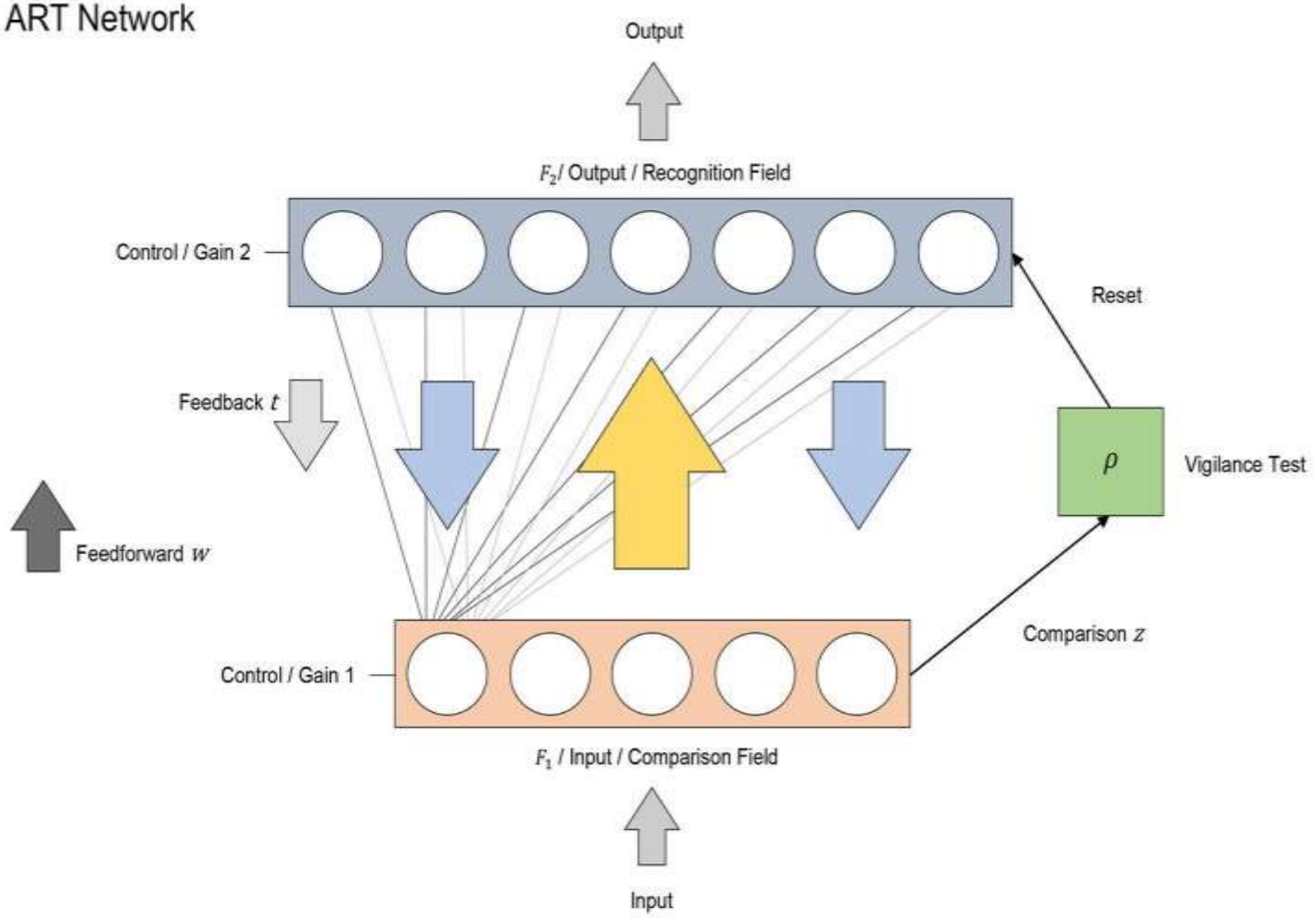
9. Structure of ART Network

Layers

1.F1 Layer → Input representation

2.F2 Layer → Category layer

3.Vigilance parameter (ρ)



10. Learning Process in ART

Two Types of Learning

Bottom-Up Learning
- Input → category
- Feature extraction

Top-Down Learning
- Category → expectation
- Pattern matching

Matching Process
- Compare input with stored pattern
- If match ≥ vigilance → learn
- Else → reset and search new category

11. ART Extensions

Types

- ART1 → Binary inputs
- ART2 → Continuous inputs
- ART3 → Biologically inspired
- Fuzzy ART → Fuzzy logic based

12. Hopfield Network

Concept

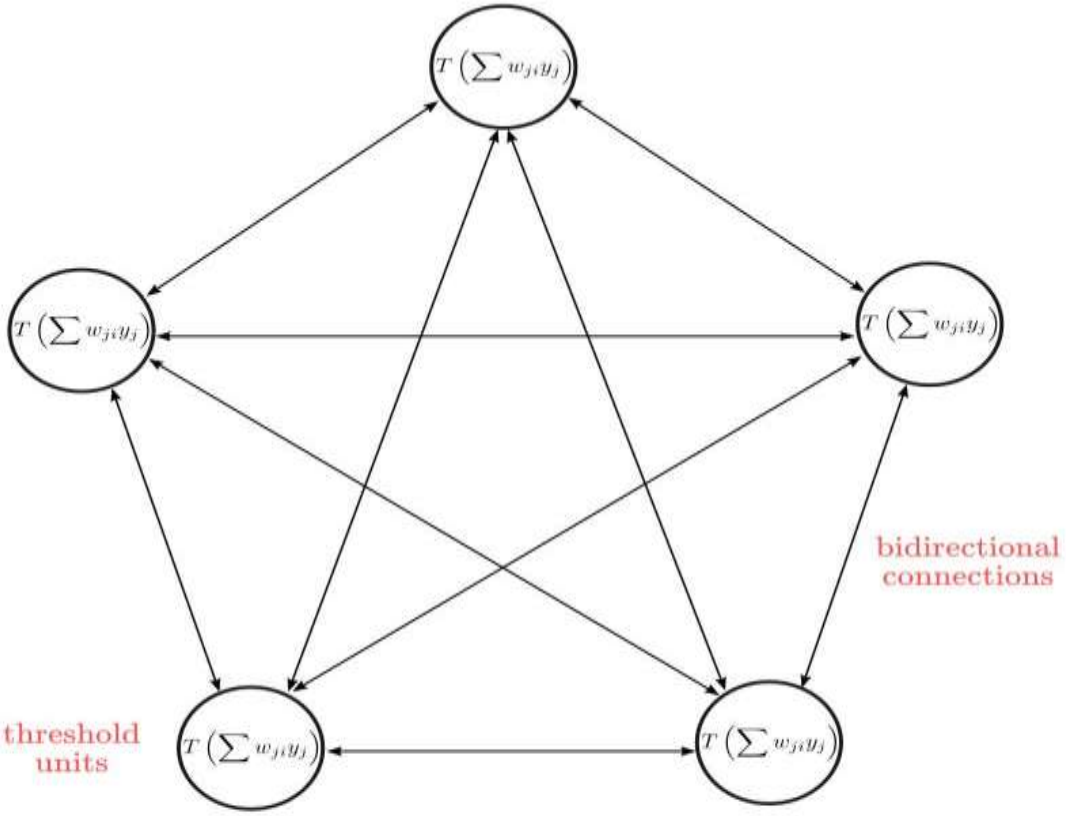
A **Hopfield Network** is:

- Recurrent neural network
- Used for associative memory

Features

- Fully connected network
- Symmetric weights
- Energy function minimization

$$E = -\frac{1}{2}\sum_i \sum_j w_{ij} x_i x_j$$



energy function

$$E = -\sum_{i<j} w_{ji} y_j y_i - \sum_i b_i y_i$$

weight update

$$\Delta w_{ji} = y_j y_i$$

13. Associative Networks

Types

❑ **Homogeneous Associative Network**

- Input = Output type
- Example: Auto-associative memory

❑ **Heterogeneous Associative Network**

- Input ≠ Output
- Example: Pattern translation

❑ **Applications**

- Pattern completion
- Noise removal
- Memory recall

14. Restricted Boltzmann Machine (RBM)

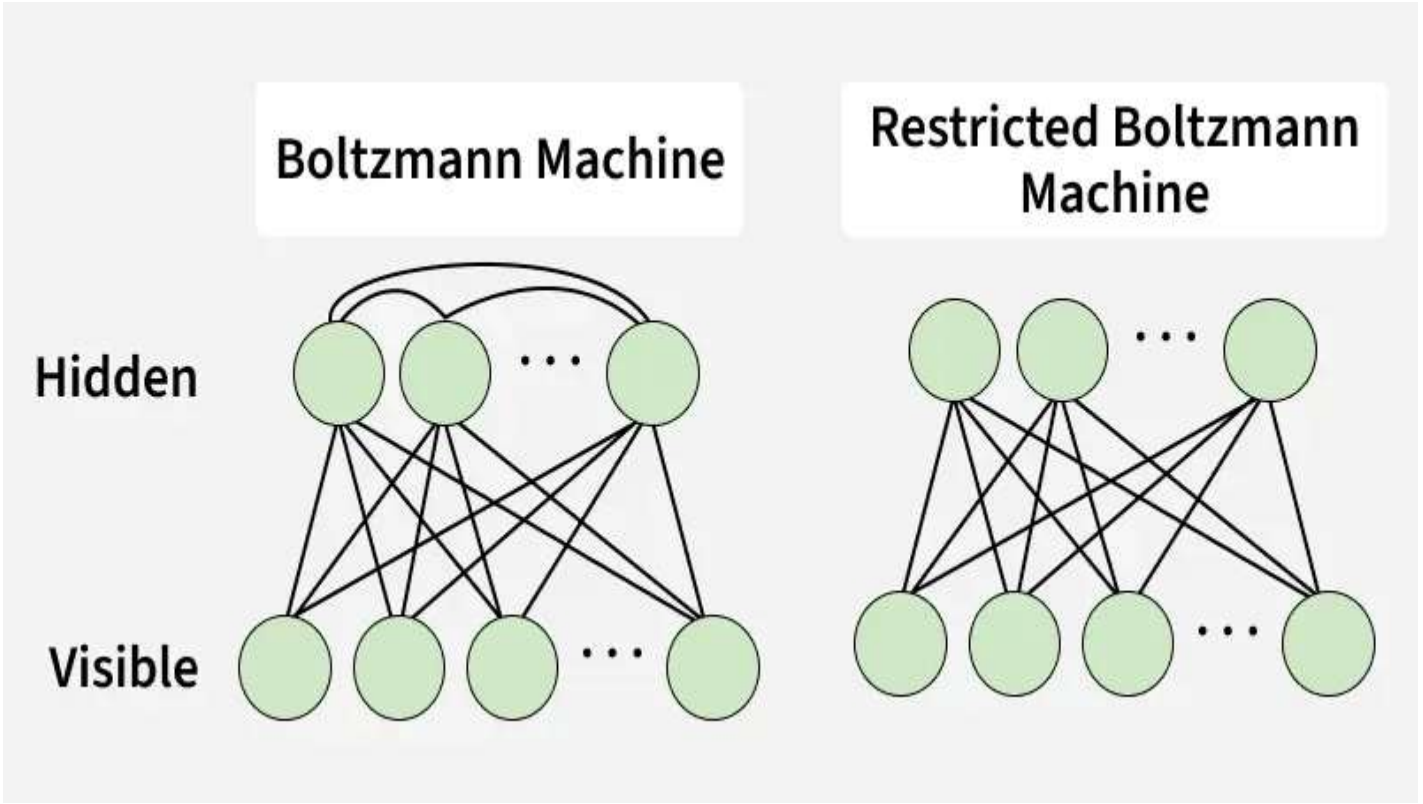
Concept

A **Restricted Boltzmann Machine** is:

- Generative model
- Two-layer network

Structure

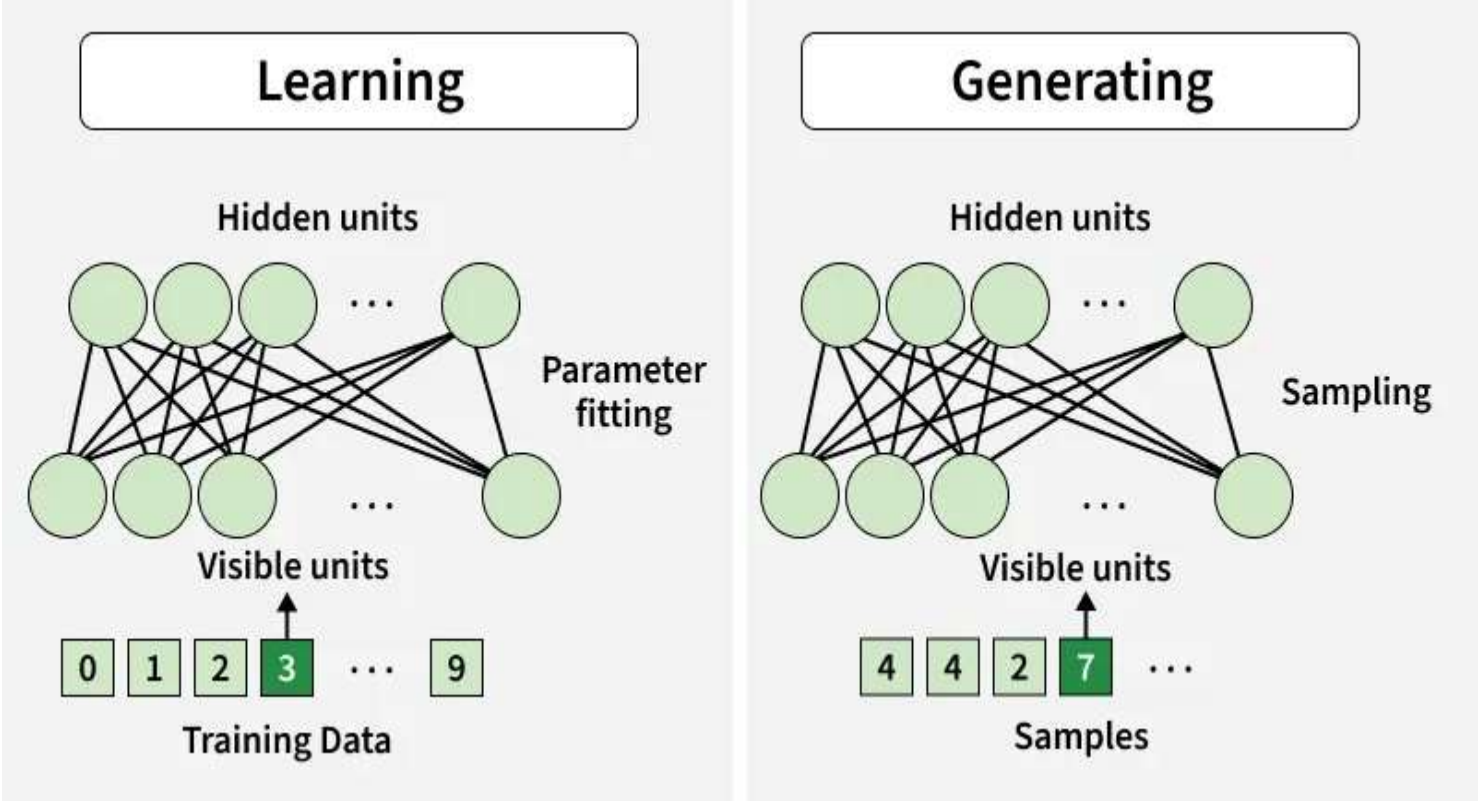
- Visible layer (input)
- Hidden layer
- No intra-layer connections



14. Restricted Boltzmann Machine (RBM)

A Restricted Boltzmann Machine (RBM) is a generative stochastic neural network consisting of two layers a visible layer and a hidden layer.

The term restricted means there are no connections within the same layer only between visible and hidden units.



https://www.geeksforgeeks.org/machine-learning/restricted-boltzmann-machine/

14. Restricted Boltzmann Machine (RBM)

RBM Architecture

- **Visible units (v):** Represent the input data

$$v = (v_1, v_2, \ldots, v_n)$$

- **Hidden units (h):** Capture latent features

$$h = (h_1, h_2, \ldots, h_m)$$

- **Weights (W):** Connections between visible and hidden units. W_{ij} connects $v_i \; and \, h_j$.

- **Biases:** Visible bias b_i and hidden bias c_j

14. Restricted Boltzmann Machine (RBM)

Energy Function:

$$E(v, h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} v_i w_{ij} h_j$$

Where,

- v_i :state of visible unit i
- h_j :state of hidden unit j
- W_{ij} :weight between visible unit i and hidden unit j

Lower energy leads to higher probability.

14. Restricted Boltzmann Machine (RBM)

Learning Process of Restricted Boltzmann Machine:

The learning process of an RBM aims to reduce the reconstruction error. This is achieved by iteratively updating the weights so that the reconstructed data becomes closer to the original data distribution.

1. Reconstruction Error

Reconstruction error is define by:

$$v^{(0)} - v^{(1)}$$

Where,

- $v^{(0)}$:original input (visible units)
- $v^{(1)}$:reconstructed input

The goal of learning is to minimize this error over successive training iterations by adjusting the weights W.

14. Restricted Boltzmann Machine (RBM)

Learning Process of Restricted Boltzmann Machine:

2. Forward Pass

In the forward pass, we compute the probability of activating hidden units given the visible input $v^{(0)}$

$$P\left(h_j = 1 \mid v^{(0)}\right) = \sigma\left(c_j + \sum_{i=1}^{n} W_{ij}\, v^{(0)}\right)$$

3. Backward Pass

In the backward pass, the RBM reconstructs the input using the hidden activations

$$P(v_i = 1 \mid h) = \sigma\left(b_i + \sum_{j=1}^{m} W_{ij}\, h_j\right)$$

14. Restricted Boltzmann Machine (RBM)

Learning Process of Restricted Boltzmann Machine:

4. Joint Probability Distribution (Gibbs Distribution)

The joint probability of a visible–hidden configuration is:

$$P(v, h) = \frac{1}{Z}\exp\big(-E(v, h)\big)$$

where the partition function is:

14. Restricted Boltzmann Machine (RBM)

Learning Process of Restricted Boltzmann Machine:

5. Generative Learning Perspective

RBM performs reconstruction, not classification or regression.

- It does not map inputs to labels

- Instead, it learns the probability distribution of the input data

Hence, RBM is a generative model, unlike discriminative models used in classification.

14. Restricted Boltzmann Machine (RBM)

Learning Process of Restricted Boltzmann Machine:

6. Error Minimization Using KL-Divergence

The difference between distributions represents the learning error and is measured using Kullback–Leibler (KL) divergence:

$$D_{KL}(p \parallel q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

Where,

- $p(x)$:true data distribution
- $q(x)$:model's reconstructed distribution

KL-divergence measures how much information is lost when $q(x)$ approximates $p(x)$

14. Restricted Boltzmann Machine (RBM)

Learning Process of Restricted Boltzmann Machine:

7. Weight Update Rule (Contrastive Divergence)

To reduce the KL-divergence RBM updates weights using:

$$\Delta W_{ij} = \eta \left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \right)$$

where

- η: Learning Rate
- $\langle v_i h_j \rangle_{\text{data}}$: expectation from real data
- $\langle v_i h_j \rangle_{\text{model}}$: expectation from reconstructed data

Ref: https://www.geeksforgeeks.org/machine-learning/restricted-boltzmann-machine/