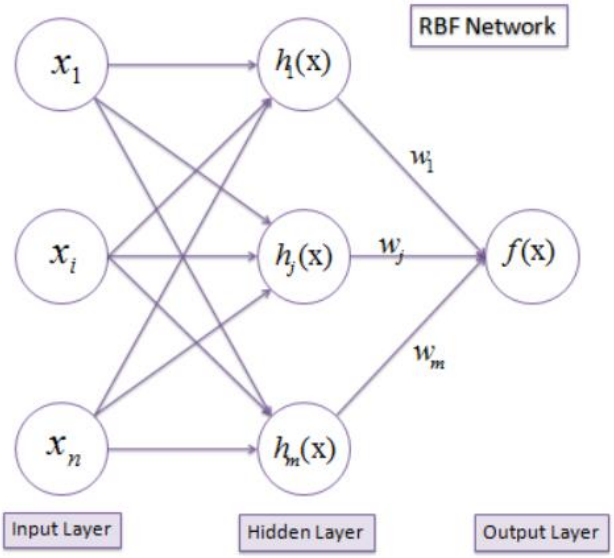
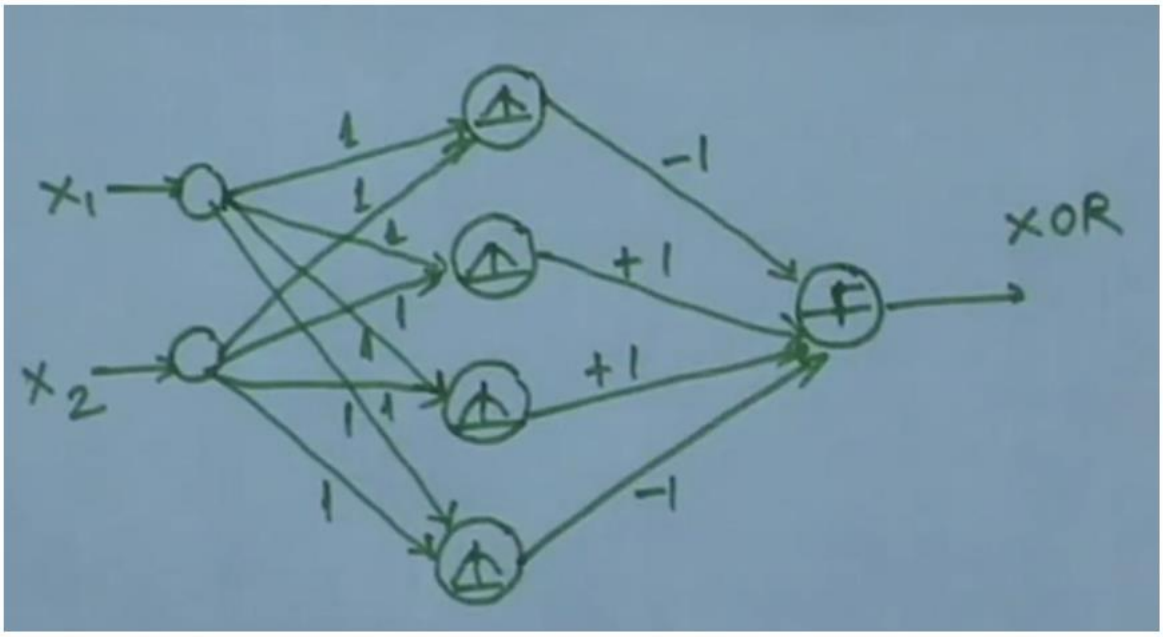


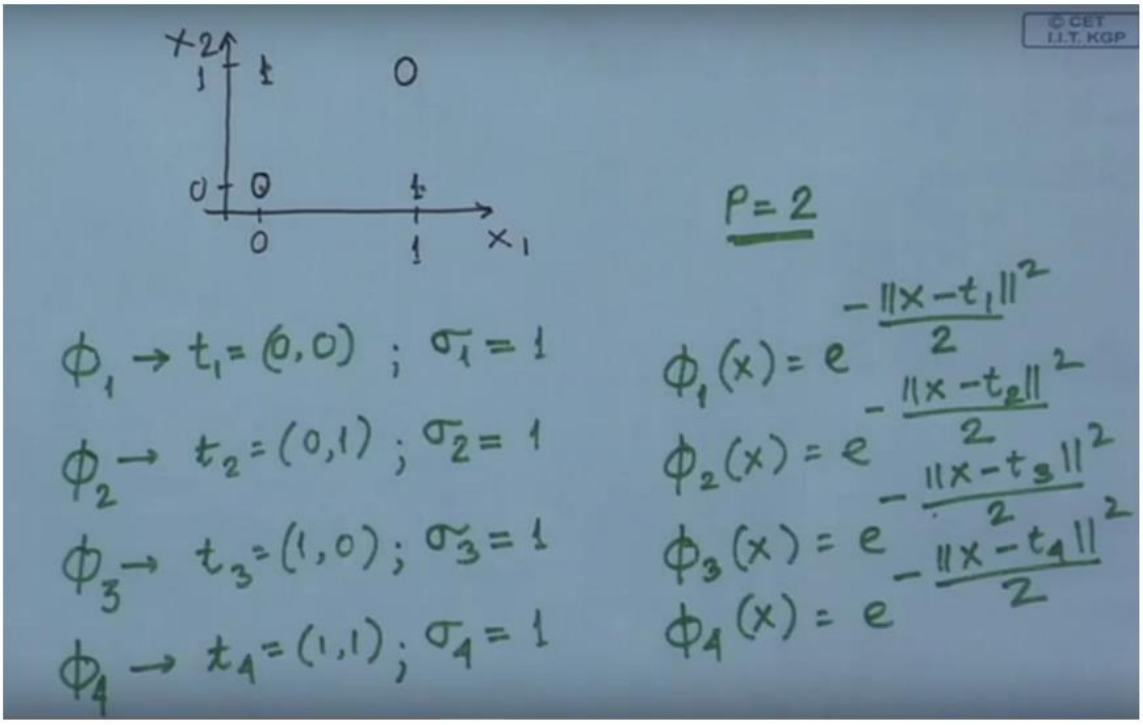
$$f(\mathrm{x}) = \sum_{j=1}^{m} w_j h_j(\mathrm{x})$$

$$h(x) = \exp\left(-\frac{(x-c)^2}{r^2}\right)$$

Classification only happens on the second phase, where linear combination of hidden functions are driven to output layer.



Architecture of XOR RBNN



Transformation function with receptors and variances.

Input	ϕ_1	ϕ_2	ϕ_3	ϕ_4	$\sum w_i \phi_i$	Output
0 0	1.0	0.6	0.6	0.4	−0.2	0
0 1	0.6	1.0	0.4	0.6	0.2	1
1 0	0.6	0.4	1.0	0.6	0.2	1
1 1	0.4	0.6	0.6	1.0	−0.2	0
	−1	+1	+1	−1		

Output → linear combination of transformation function is tabulated.

Only Nodes in the hidden layer perform the radian basis transformation function.

- Output layer performs the linear combination of the outputs of the hidden layer to give a final probabilistic value at the output layer.
- So the classification is only done only @ ***(hidden layer → output layer)***

*1. Training in RBNN is **faster** than in Multi-layer Perceptron (MLP) → takes **many interactions** in MLP.*
*2. We can easily interpret what is the meaning / **function of the each node** in hidden layer of the RBNN. This is **difficult** in MLP.*
*3. (what should be the # of nodes in hidden layer **&** the # of hidden layers) this **parameterization** is difficult in MLP. But this is not found in RBNN.*
*4. **Classification** will take more time in RBNN than MLP.*

Key differences between RBF and MLP are:-

MLP:

1. The activation function can be any non linear function which can serve the purpose.
2. The final layer in MLP also uses the activation function before linearly combining it.
3. There can be more than one hidden layer in MLP.

RBF:

1. The activation function is a function of the euclidean distance of input vector and a certain vector.
2. The final layer of RBF don't use activation function, it rather linearly combines the output of the previous neuron.
3. There is only one hidden layer in RBF and one output layer.
4. The final layer have only one neuron.

Typically RBF and MLP share a common neuron model.

Feature of network architecture	**Neural network type**	
	MLP	**RBF**
Signal transmission	Feed-forward	Feed-forward
Process of building the model	One stage	Two different independent stages: • First stage: the probability distribution is established by means of radial basis functions • Second stage: the network learns the relations between input x and output y Note: The lag is only visible in RBF in the output layer
Threshold	Yes	No
Type of parameters	Weights and thresholds	• Location and width of basis function • Weights binding basis functions with output
Functioning time	Faster	Slower (bigger memory and size required)
Learning time	Slower	Faster