Lec 8

RADIAL BASIS FUNCTION NETWORK

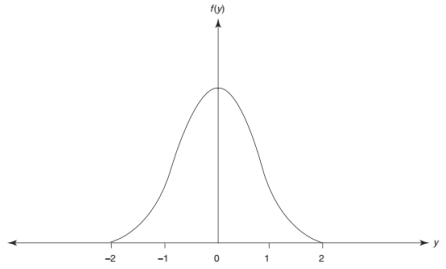
Dr. Mohammad Mhawish

- In this lecture, we take a completely different approach. Specifically, we solve the problem of classifying nonlinearly separable patterns by proceeding in a *hybrid* manner, involving two stages:
- The first stage transforms a given set of nonlinearly separable patterns into a new set for which, under certain conditions, the likelihood of the transformed patterns becoming linearly separable is high; the mathematical justification of this transformation is traced to an early paper by Cover (1965).
- The second stage completes the solution to the prescribed classification problem by using least-squares estimation

- using a *radial-basis function (RBF) network*, the structure of which consists of only three layers:
- The input layer is made up of source nodes (sensory units) that connect the network to its environment.
- The second layer, consisting of *hidden units*, applies a nonlinear transformation from the input space to the hidden (feature) space. For most applications, the dimensionality of the only hidden layer of the network is high; this layer is trained in an unsupervised manner using stage 1 of the hybrid learning procedure.
- The output layer is *linear*, designed to supply the response of the network to the activation pattern applied to the input layer; this layer is trained in a supervised manner using stage 2 of the hybrid procedure.

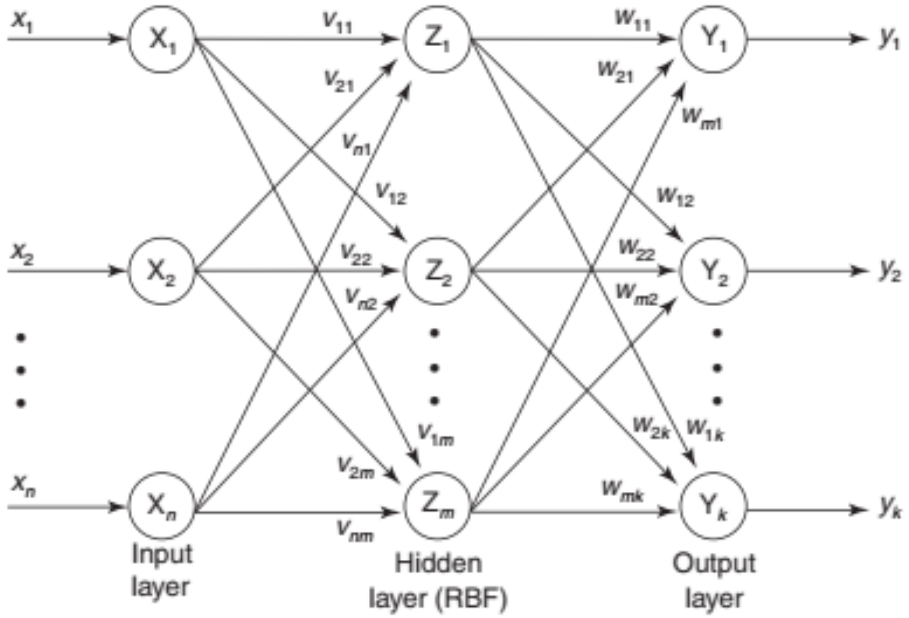
- **RADIAL-BASIS-FUNCTION NETWORKS**
  specifically, we have three layers:
  1. *Input layer*, which consists of *m0* source nodes, where *m0* is the dimensionality of the input vector **x**.
  2. *Hidden layer*, which consists of the same number of computation units as the size of the training sample, namely, *N*; each unit is mathematically described by a radial basis function

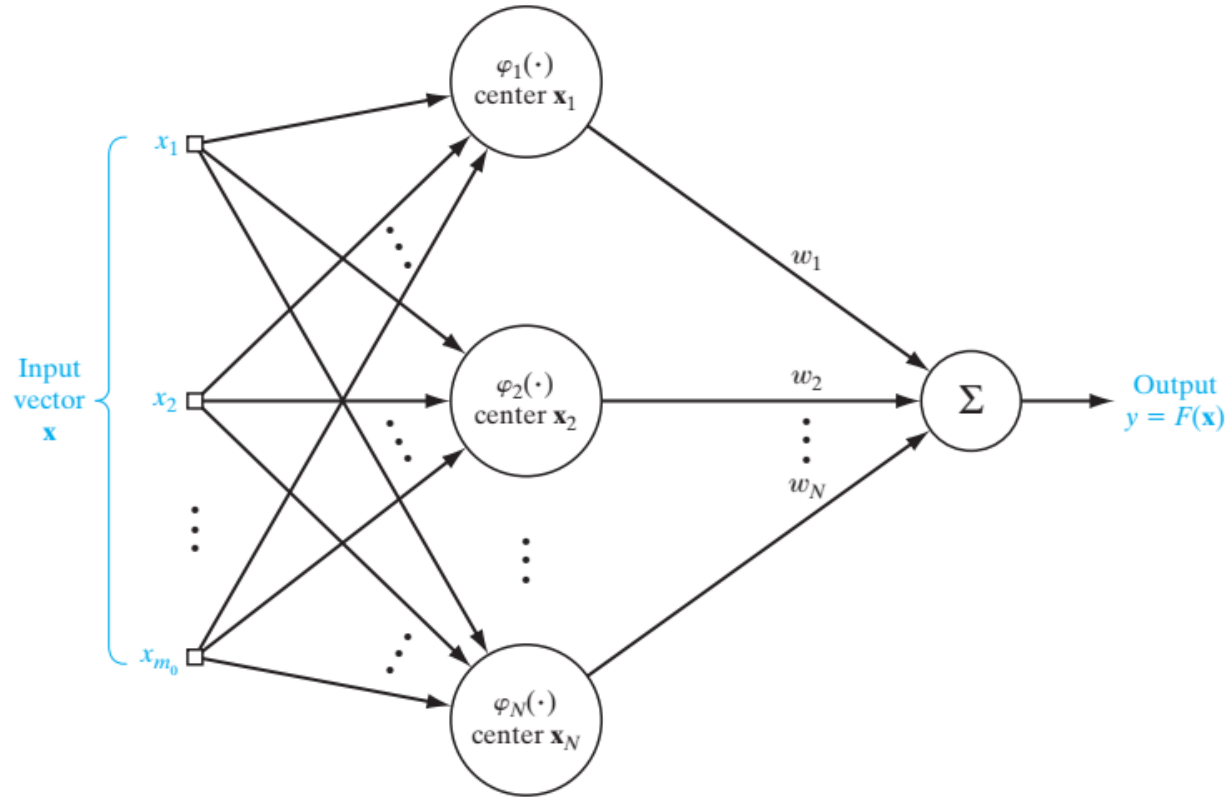
$$\varphi_j(\mathbf{x}) = \varphi(\mathbf{x} - \mathbf{x}_j)$$
$$= \exp\left(-\frac{1}{2\sigma_j^2}\|\mathbf{x} - \mathbf{x}_j\|^2\right), \qquad j = 1, 2, ..., N$$



where σ_j is a measure of the *width* of the jth Gaussian function with center $\mathbf{x}_j$.

- *3. Output layer*, which,in the RBF structure of ,consists of a single computational unit. Clearly, there is no restriction on the size of the output layer, except to say that typically the size of the output layer is much smaller than that of the hidden layer.


x_1
x_2
x_n
X_1
X_2
X_n
Input layer
v_{11}
v_{21}
v_{n1}
v_{12}
v_{22}
v_{n2}
v_{1m}
v_{2m}
v_{nm}
Z_1
Z_2
Z_m
Hidden layer (RBF)
w_{11}
w_{21}
w_{m1}
w_{12}
w_{22}
w_{m2}
w_{2k}
w_{1k}
w_{mk}
Y_1
Y_2
Y_k
Output layer
y_1
y_2
y_k


Input vector $\mathbf{x}$
x_1
x_2
x_{m_0}
$\varphi_1(\cdot)$ center $\mathbf{x}_1$
$\varphi_2(\cdot)$ center $\mathbf{x}_2$
$\varphi_N(\cdot)$ center $\mathbf{x}_N$
w_1
w_2
w_N
Σ
Output $y = F(\mathbf{x})$

if hidden layer > input ?

How we can choose the center?

Training Algorithm

Step 0: Set the weights to small random values.

Step 1: Perform Steps 2–8 when the stopping condition is false.

Step 2: Perform Steps 3–7 for each input.

Step 3: Each input unit (x_i for all $i = 1$ to n) receives input signals and transmits to the next hidden layer unit.

Step 4: Calculate the radial basis function.

Step 5: Select the centers for the radial basis function. The centers are selected from the set of input vectors. It should be noted that a sufficient number of centers have to be selected to ensure adequate sampling of the input vector space.

Step 6: Calculate the output from the hidden layer unit:

$$v_i(x_i) = \frac{\exp\left[-\sum_{j=1}^{r}(x_{ji} - \hat{x}_{ji})^2\right]}{\sigma_i^2}$$

where $\hat{x}_{ji}$ is the center of the RBF unit for input variables; σ_i the width of ith RBF unit; x_{ji} the jth variable of input pattern.

Step 7: Calculate the output of the neural network:

$$y_{net} = \sum_{i=1}^{k} w_{im} v_i(x_i) + w_0$$

where k is the number of hidden layer nodes (RBF function); y_{net} the output value of mth node in output layer for the nth incoming pattern; w_{im} the weight between ith RBF unit and mth output node; w_0 the biasing term at nth output node.

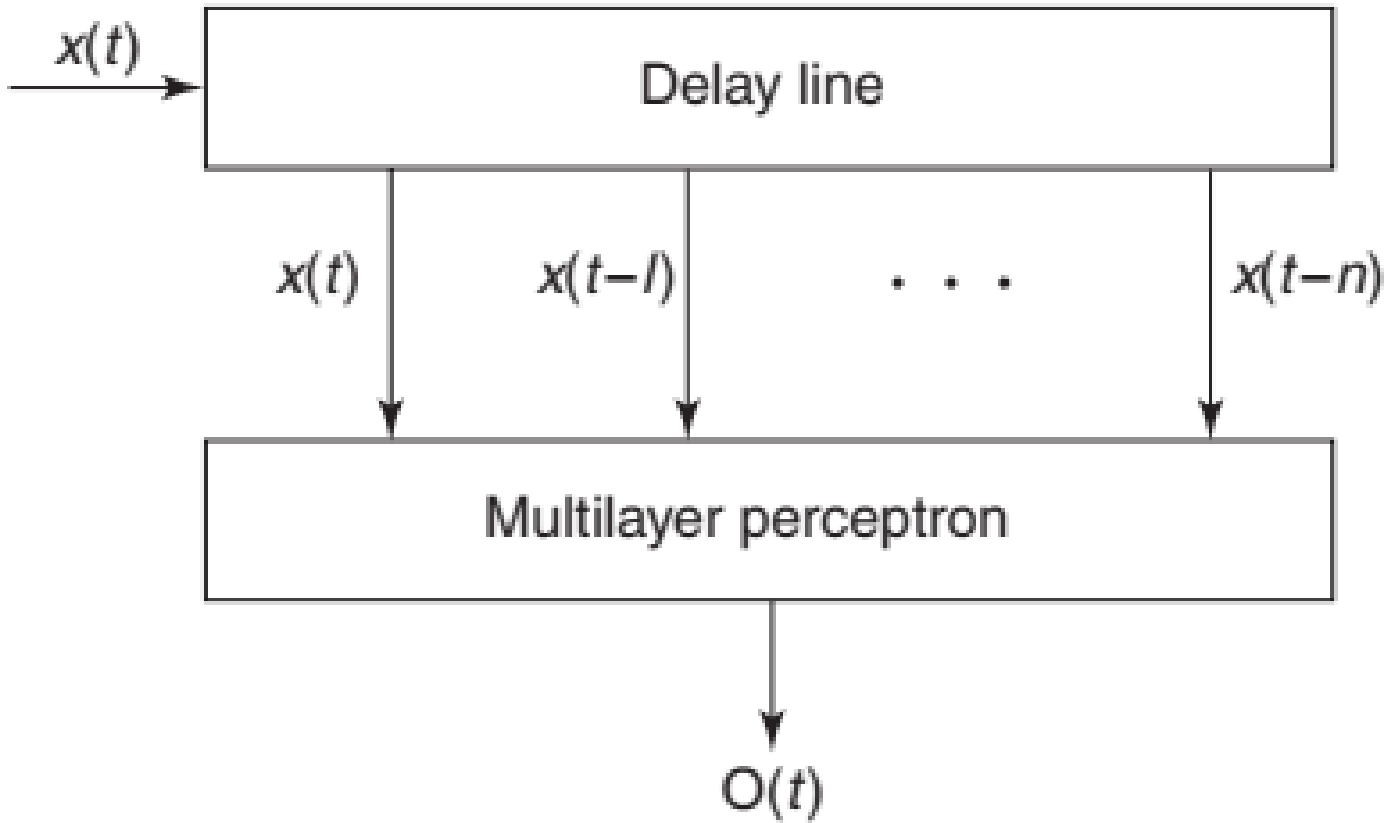
Step 8: Calculate the error and test for the stopping condition. The stopping condition may be number of epochs or to a certain extent weight change.

TIME DELAY NEURAL NETWORK

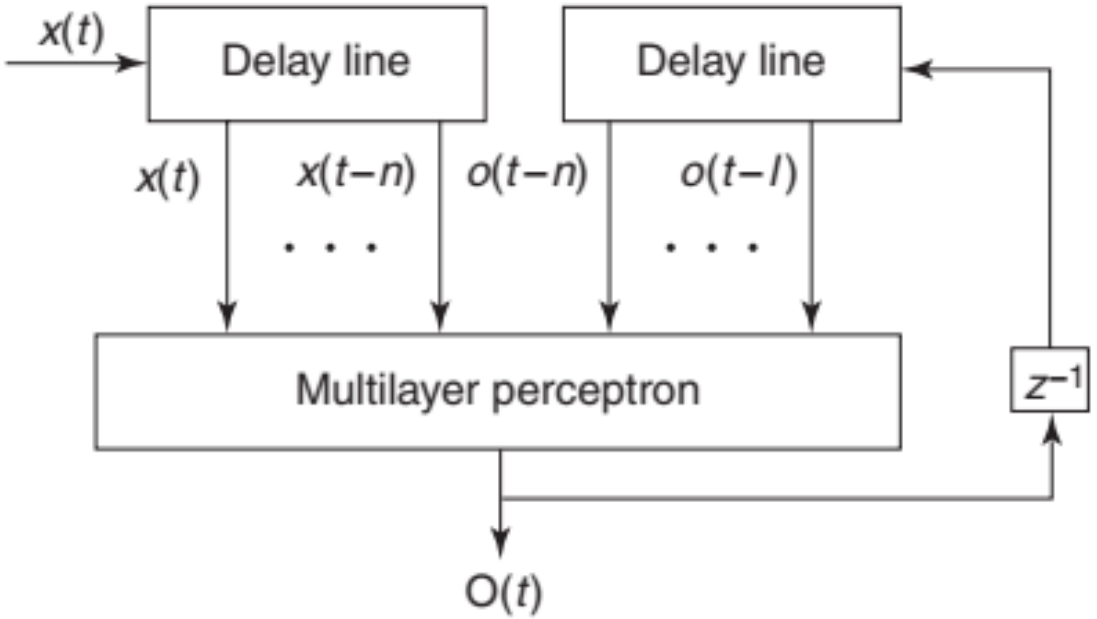
- The neural network has to respond to a sequence of patterns. Here the network is required to produce a particular output sequence in response to a particular sequence of inputs. A shift register can be considered as a tapped delay line. Consider a case of a multilayer perceptron where the tapped outputs of the delay line are applied to its inputs. This type of network constitutes a time delay neural network (TDNN). The output consists of a finite temporal dependence on its inputs, given as

$$U(t) = F\left[x(t),\ x(t-1),\ldots,x(t-n)\right]$$

where F is any nonlinearity function. The multilayer perceptron with delay line


x(t)
Delay line
x(t)
x(t−l)
. . .
x(t−n)
Multilayer perceptron
O(t)

- When the function U t() is a weighted sum, then the TDNN is equivalent to a finite impulse response filter (FIR). In TDNN, when the output is being fed back through a unit delay into the input layer, then the net computed here is equivalent to an infinite impulse response (IIR) filter.



- Thus, a neuron with a tapped delay line is called a TDNN unit, and a network which consists of TDNN units is called a TDNN. A specific application of TDNNs is speech recognition. The TDNN can be trained using the back-propagation learning rule.