7

WEEK7: RADIAL BASIS FUNCTION NETWORKS

Vasant Honavar

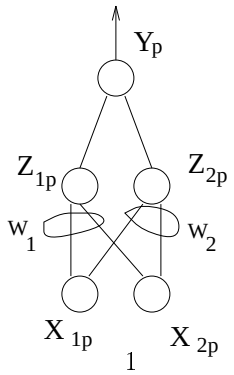
Artificial Intelligence Research Group
Department of Computer Science
Iowa State University
Ames, Iowa 50011

©Vasant Honavar, 1997.

1 RADIAL BASIS FUNCTIONS

Note that although the universal approximation theorem for multi-layer networks was stated using sigmoid neurons, similar theorems can be derived for a wide range of non-linear differentiable activation functions. One such family of functions is the radial basis function (RBF). A radial basis function serves as fuzzy or soft template matching function which compares the input pattern with a stored pattern (e.g., the weight vector) and produces an output that is inversely proportional to the distance between the them. A commonly used measure of distance is the square of the Euclidian distance between the two patterns.

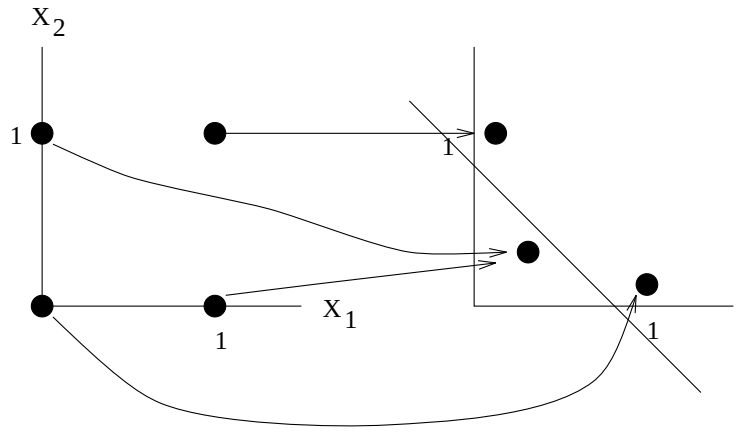
The following example illustrates the operation of simple RBF.



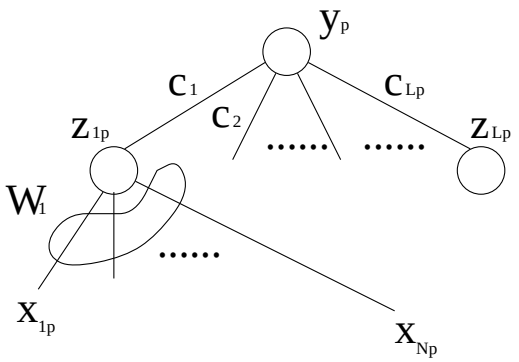
$$Z_{1p} = e^{-\frac{\|W_1 - X_p\|^2}{\sigma_1^2}}, \qquad Z_{2p} = e^{-\frac{\|W_2 - X_p\|^2}{\sigma_2^2}}$$

Suppose $W_1 = [0,0]^T, W_2 = [1,1]^T$ and $\sigma_1 = \sigma_2 = 1$. Then,

x_1	x_2	z_1	z_2
0	0	1	.135
0	1	.37	.37
1	0	.37	.37
1	1	.135	1



More generally, we might have a network of RBFs shown below.



where

$$Z_{jp} = e^{\frac{-\|X_p - W_j\|^2}{2\sigma_j^2}}$$

$$y_p = \sum_{j=1}^{L} C_j Z_{jp}$$

$$E_p = \frac{1}{2}(d_p - y_p)^2, \quad d_p = desired \;\; output$$

$$X_p = [x_{1p} \ldots x_{Np}]^T$$

$$W_j = [w_{ji} \ldots w_{jN}]^T$$

The weight update equations for W_j, C_j, σ_j can be computed as follows:

1. $\frac{\partial E_p}{\partial C_j} = -(d_p - y_p)Z_{jp}$
   Thus, $C_j \leftarrow C_j + \eta(d_p - y_p)Z_{jp}$
2. $\frac{\partial E_p}{\partial W_{ji}} = \frac{\partial E_p}{\partial y_p}\frac{\partial y_p}{\partial Z_{jp}}\frac{\partial Z_{jp}}{\partial W_{ji}}$
   $\frac{\partial Z_{jp}}{\partial W_{ji}} = \frac{\partial}{\partial W_{ji}}[\frac{-||X_p - W_j||^2}{2\sigma_j^2}]e^{\frac{-||X_p - W_j||^2}{2\sigma_j^2}}$
   $= \frac{-Z_{jp}}{2\sigma_j^2}\frac{\partial}{\partial W_{ji}}[(X_{1p} - W_{j1})^2 + \cdots + (X_{ip} - W_{ji})^2 + \cdots + (X_{Np} - W_{jN})^2]$
   $= \frac{-Z_{jp}}{2\sigma_j^2}[0 + 0 + \cdots + 2(X_{ip} - W_{ji})(-1) + \cdots + 0]$
   $= \frac{Z_{jp}}{\sigma_j^2}(X_{ip} - W_{ji})$
   Thus, $W_{ji} \leftarrow W_{ji} + \eta(d_p - y_p)C_j\frac{Z_{jp}}{\sigma_j^2}(X_{ip} - W_{ji})$
3. $\frac{\partial E_p}{\partial \sigma_j} = \frac{\partial E_p}{\partial y_p}\frac{\partial y_p}{\partial Z_{jp}}\frac{\partial Z_{jp}}{\partial \sigma_j}$
   $\frac{\partial Z_{jp}}{\partial \sigma_j} = Z_{jp}\frac{\partial}{\partial \sigma_j}[\frac{-||X_p - W_{ji}||^2}{2\sigma_j^2}] = -Z_{jp}\frac{2}{\sigma_j}\ln Z_{jp}$
   Thus, $\sigma_j \leftarrow \sigma_j - \eta(d_p - y_p)C_j Z_{jp}\frac{2}{\sigma_j}\ln Z_{jp}$

The hidden neurons compute essentially a gaussian function which has maximum value when the input exactly coincides with the weight vector. As the difference between the weight vector and the input increases, the neuron's output approaches zero. The rate of this decrease in output is governed by σ_j.

We can generalize such networks further by using a more general form of radial basis functions. A weighted norm of a vector V is defined as follows:

$$||V||_C^2 = (CV)^T(CV) = V^T C^T C V$$

where C is the norm weighting matrix. If C^TC is an identity matrix (I) then $||V||_C^2 = ||V||^2 = (v_1^2 + \cdots + v_n^2)$. Suppose we pick a matrix $\frac{1}{2}\Sigma^{-1} = C_j^T C_j$, then we have

$$Z_{jp} = e^{-1/2(X_p - W_j)^T {\Sigma_j}^{-1}(X_p - W_j)}$$

.

Note following rules of derivatives for vectors:

- $\frac{d}{dX}[AX] = A$
- $\frac{d}{dX}[X^TAX] = 2AX$ (when A is symmetric)
- $\frac{d}{dA}[X^TAX] = XX^T$

Now, we can get the weight update equations for W_j and ${\Sigma_j}^{-1}$ (for networks with multiple output neurons) as follows:

1. $\frac{\partial E_p}{\partial W_j} = \frac{\partial E_p}{\partial Z_{jp}} \cdot \frac{\partial Z_{jp}}{\partial W_j}$
   $= \sum_k \frac{\partial E_p}{\partial y_{kp}} \cdot \frac{\partial y_{kp}}{\partial Z_{jp}} \cdot \frac{\partial Z_{jp}}{\partial W_j}$
   $= -\sum_k (d_{kp} - y_{kp}) C_{kj} Z_{jp} \frac{\partial}{\partial W_j}[-1/2(X_p - W_j)^T {\Sigma_j}^{-1}(X_p - W_j)]$
   $= -\sum_k (d_{kp} - y_{kp}) C_{kj} Z_{jp} {\Sigma_j}^{-1}(X_p - W_j)$
   Thus,
   $$W_j \leftarrow W_j + \eta \sum_k (d_{kp} - y_{kp}) C_{kj} Z_{jp} \Sigma^{-1}(X_p - W_j)$$
2. $\frac{\partial E_p}{\partial {\Sigma_j}^{-1}} = \frac{\partial E_p}{\partial Z_{jp}} \cdot \frac{\partial Z_{jp}}{\partial {\Sigma_j}^{-1}}$
   $= -\sum_k (d_{kp} - y_{kp}) C_{kj} \frac{\partial Z_{jp}}{\partial {\Sigma_j}^{-1}}$
   $= -\sum_k (d_{kp} - y_{kp}) C_{kj} Z_{jp} (-1/2)(X_p - W_j)(X_p - W_j)^T$
   Thus,
   $${\Sigma_j}^{-1} \leftarrow {\Sigma_j}^{-1} - \frac{\eta}{2} \sum_k (d_{kp} - y_{kp}) C_{kj} Z_{jp} (X_p - W_j)(X_p - W_j)^T$$

In practice, several variants of RBF learning algorithms can be used.