Chapter 1. General Introduction to Wavelets and Artificial Neural Networks

1.1. Introduction

The work that is to be presented in this thesis is related to the exploration of the methods of wavelet transforms and various neural network architectures as applied to signal processing and pattern recognition in developing an Optical Character Recognition (OCR) system for the Gujarati script. Optical Character Recognition is used to convert the digital images of printed documents in to files of editable text by using computers.

Development of Optical Character Recognition (OCR) systems for various scripts used by different societies is among the most important tasks that are grouped under the title of Natural Language Processing Systems. Much work has gone in to the study and development of these systems in the past 50 years and, as a result, quite reliable OCR systems are now available for the European and some other scripts.

But the scenario regarding the OCR systems for Indian (or Indic) scripts is not a very happy one since there are almost no commercially successful OCR products for these scripts available in the market. This is partly because of the fact that these scripts with the numerous conjuncts and the vowel modifiers occurring in all directions of the basic symbol are much more complex in comparison to the linear European scripts. There have been many documented efforts, mainly at research level, regarding the development of OCR technology for Indic scripts during the past 35 years. But, regarding the OCR technology for the Gujarati script, there had been only one documented effort [1] before 2005.

In this thesis, we have explored the usage of Wavelets and Artificial Neural Networks (ANN) for the development of an Optical Character Recognition (OCR) system for the Gujarati script. Wavelets with the important characteristics of space and frequency localization are found to be good for extracting features of images. Artificial Neural

Networks with their generalization capabilities are good for the construction of a robust classifier. This introductory chapter is organized as follows:

This chapter contains five sections. After giving an introduction in the first section, we present the general introduction to bandlimited functions and Shannon's sampling theorem in the second section. Section 3 highlights the advantages of Wavelets over the Fourier transform followed by the introduction of continuous and discrete wavelet. In section 4, we describe the applicability of various Artificial Neural Network architectures and ultimately present detailed discussion of two of the most widely used ANN architectures viz. Multilayer Perceptron and Radial Basis Funciton networks. At the end a brief summary and organization of the thesis is presented in section-5.

1.2. Shannon's theorem

The use of Shannon's theorem which is based on bandlimited functions plays a vital role in Wavelets. A function f in $L^2(R)$[†] is called bandlimited if its Fourier transform $\hat{f}\left(\hat{f}(\xi)=\frac{1}{\sqrt{2\pi}}\int dt\, e^{-i\omega t} f(t)\right)$ has compact support, i.e. $\hat{f}(\xi)\equiv 0$ *for* $|\xi| \succ \Omega$, where Ω is a finite real number.

Let us suppose, for simplicity, that $\Omega = \pi$. Then $\hat{f}$ can be represented by its Fourier series,

$$\hat{f}(\xi)=\sum_{n\in Z} c_n e^{-in\xi}$$

where $c_n=\frac{1}{2\pi}\int_{-\pi}^{\pi} d\xi\, e^{in\xi}\,\hat{f}(\xi)$

$$=\frac{1}{2\pi}\int_{-\infty}^{\infty} d\xi\, e^{in\xi}\hat{f}(\xi)$$

$$=\frac{1}{\sqrt{2\pi}} f(n) \quad \left(\text{Inverse Fourier transform: } f(n)=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} d\xi\, e^{in\xi}\hat{f}(\xi)\right)$$

[†] $L^2(R)$: Set of square integrable function. Let f and g are in $L^2(R)$ then L^2-inner product is defined as

$$\langle f,g\rangle=\int dx\, f(x)\,\bar{g}(x)$$

From the inversion of Fourier transform, it follows that

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} d\xi \, e^{ix\xi} \hat{f}(\xi)$$

$$= \frac{1}{\sqrt{2\pi}} \int_{-\pi}^{\pi} d\xi \, e^{ix\xi} \sum_n c_n \, e^{-in\xi}$$

$$= \frac{1}{\sqrt{2\pi}} \sum_n c_n \int_{-\pi}^{\pi} d\xi \, e^{i(x-n)\xi}$$

$$= \sum_n f(n) \frac{\sin \pi(x-n)}{\pi(x-n)} \qquad (1.1)$$

Formula (1.1) tells us that f is completely determined by its "sampled" values $f(n)$.

If we lift the restriction $\Omega = \pi$ and assume support $\hat{f} \subset [-\Omega, \Omega]$, with Ω arbitrary real number, then equation (1.1) becomes

$$f(x) = \sum_n f\left(n\frac{\pi}{\Omega}\right) \frac{\sin(\Omega x - n\pi)}{\Omega x - n\pi} \qquad (1.2)$$

The function in equation (1.2) is now determined by its samples $f\left(n\frac{\pi}{\Omega}\right)$, $n \in Z$.

Shannon's theorem states that an Ω-bandlimited function can be reconstructed completely from its values

$$(f(kT) \mid k \in Z), \quad T = \frac{\pi}{\Omega}$$

sampled at the discrete points kT. "Completely" means at all points $t \in R$ we get back the exact original value $f(t)$.

1.3. Wavelets

Due to the localization properties of wavelets in time and frequency domain, they are widely used in the field of image analysis, feature extraction etc.

The wavelet transform is a tool that cuts up data or functions or operators into different frequency components, and then studies each component with a resolution matched to its scale. In this chapter, we emphasize only on signal processing. The

wavelet transform of a signal evolving in time (e.g. the amplitude of the pressure on an eardrum, for acoustical applications) depends on two variables: scale (or frequency) and time; wavelets provide a tool for time-frequency localization [11].

A brief description of the progression of concepts from Fourier transforms to wavelet transforms via the Windowed Fourier transform is provided below:

- In many applications, given a continuous signal *f(t)*, one is interested in its frequency content *locally* in time. This is similar to music notation, for example, which tells the player which notes (= frequency information) to play at any given moment. Standard Fourier transform,

$$(Ff)(\omega) = \frac{1}{\sqrt{2\pi}} \int dt\, e^{-i\omega t} f(t)$$

gives a representation of the frequency content of *f*, but it can not provide the information concerning time-localization of *f*.

- Time-localization can be achieved by first windowing the signal *f*, so as to cut off only a well-localized slice of *f*, and then taking its Fourier transform:

$$(T^{win} f)(\omega, t) = \int ds\, f(s) g(s-t) e^{-i\omega s} = g_{\omega,t}$$

This is the windowed Fourier transform, which is a standard technique for time-frequency localization.

- The wavelet transform provides a similar time-frequency description, with a few important differences. The wavelet transform (to be explained in the following subsection) of a function *f* results in an expression of the following type involving two parameters *a* and *b* called the dilation and translation:

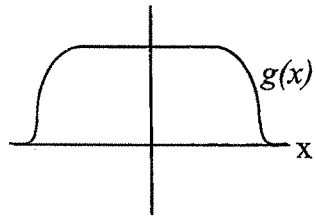
$$(T^{wav} f)(a,b) = |a|^{-1/2} \int dt\, f(t) \psi\left(\frac{t-b}{a}\right)$$

$$(T^{wav} f)(a,b) = \int dt\, f(t) \psi^{a,b}$$

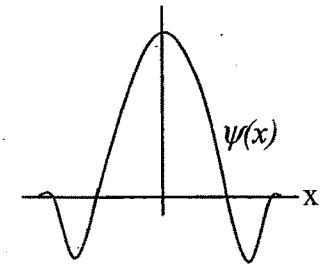
where $\psi^{a,b} = |a|^{\frac{1}{2}} \psi\left(\frac{t-b}{a}\right)$ is called analyzing wavelet.

In this case ψ satisfies

$$\int dt\,\psi(t) = 0$$



[Fig. 1.1 Windowed Fourier transform $g_{w,t}$]



[Fig.1.2 Wavelets $\psi^{j,k}$]

The difference between the wavelet and windowed Fourier transforms lies in the shapes of the analyzing functions $g_{w,t}$ and $\psi^{i,j}$ as shown in the figures 1.1 and 1.2. The functions $g_{w,t}$ all consist of the same envelope function g, translated to the proper time location, and "filled in" with higher frequency oscillations. All the $g_{\omega,t}$, regardless of the value of ω, have the same width. In contrast, the $\psi^{i,j}$ have time-widths adapted to their frequency: high frequency $\psi^{i,j}$ are very narrow, while low frequency $\psi^{i,j}$ are much broader (in the case of continuous wavelet transform). As a result, the wavelet transform is better than the windowed Fourier transform to "zoom in" on very short lived high frequency phenomena, such as transients in signals.

In the following subsections, we discuss Continuous Wavelet Transform and Discrete Wavelet Transform in some detail.

1.3.1 Continuous Wavelet Transform

In many applications, given a signal *f(t)* ($f \in L^2(R)$), one is interested in its frequency content locally in time. The wavelet transform provides time-frequency description.

The Continuous Wavelet Transform can be defined as below:

$$(T^{wav} f)(a,b) = \left\langle f, \psi^{a,b} \right\rangle = |a|^{-1/2} \int dt\, f(t) \psi\left(\frac{t-b}{a}\right) \qquad (1.3)$$

where a and b are the dilation and translation parameters respectively which vary continuously over R (with the constraint $a \neq 0$) and $\psi \in L^2(R)$.

The wavelet transform is given in equation (1.3) and a function can be reconstructed from its wavelet transform by means of the resolution of identity formula [11]:

$$f = C_\psi^{-1} \iint \frac{da\,db}{a^2} \langle f, \psi^{a,b} \rangle \psi^{a,b}$$

where $\psi^{a,b}(t) = |a|^{-1/2} \psi\left(\frac{t-b}{a}\right)$, and $\langle , \rangle$ denotes the L^2 – inner product. The constant C_ψ depends only on Ψ and is given by

$$C_\psi = 2\pi \int_{-\infty}^{\infty} d\xi \, |\hat{\psi}(\xi)|^2 |\xi|^{-1} \quad < \quad \infty \qquad (1.4)$$

where $\hat{\psi}$ is the Fourier transform of the function Ψ. The condition $C_\psi < \infty$ is known as the admissibility condition.

A function $\Psi : R \to C$ satisfying the conditions $\Psi \in L^2(R)$, $\|\psi\| = 1$ and $C_\psi < \infty$ is called a mother wavelet or simply a wavelet.

The following example is an illustration of compactly supported Continuous Wavelet Transform (CWT).

Example: Mexican Hat

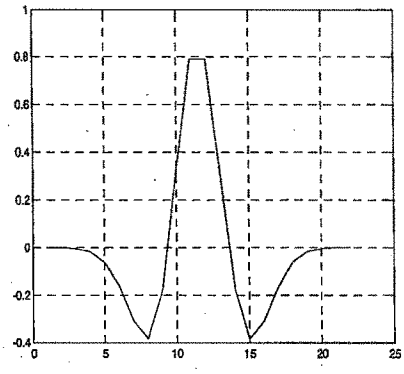
The Mexican Hat function is the second derivative of the Gaussian function $e^{-\frac{t^2}{2}}$. It was first used in computer vision to detect multiscale edges [29]. The use of Mexican hat function is extended as a nonlinear activation function in the field of Artificial Neural Networks (to be discussed later in this chapter). The branch of ANN in which all the neurons of hidden layers possess continuous wavelets as an activation function is known as Wavelet Neural Networks (WNN). Mexican hat function is used quite frequently in WNN. If we normalize the second derivative of the Gaussian function so that its L^2 – norm becomes 1, we get

$$\psi(t) = \frac{2}{\sqrt{3}} \pi^{-1/4} (1 - t^2) \exp^{-t^2/2}$$

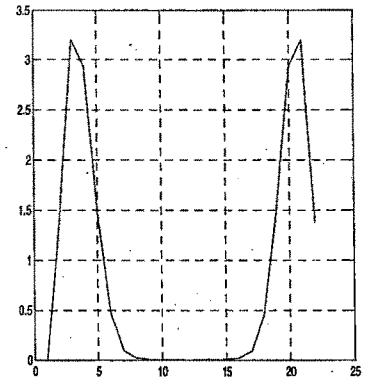
and its Fourier transform is

$$\hat{\psi}(\omega) = \frac{\sqrt{8}}{\sqrt{3}} \pi^{-1/4} \omega^2 \exp\left(\frac{-\omega^2}{2}\right)$$

Figures 1.3 and 1.4 demonstrate Ψ and the magnitude of its Fourier transform $\hat{\Psi}$.



[Fig. 1.3 Graph of Maxican Hat Ψ]



[Fig. 1.4 Graph of $|\hat{\Psi}|$]

1.3.2 Discrete Wavelet Transform

Shannon's sampling theorem (equation 1.2) accomplishes the full reconstruction of a bandlimited time signal f from a discrete collection $(f(kT) \mid k \in Z)$, $T = \frac{\pi}{\Omega}$ of sample values [32].

Suppose $B=\{v_0, v_1, ..., v_{N-1}\}$ is a basis for $l^2(Z_N)$ such that all the basis elements of B are localized in space. For a vector $\mathbf{z} \in l^2(Z_N)$, we can write

$$\mathbf{z} = \sum_{n=0}^{N-1} a_n v_n \qquad (1.5)$$

for some scalars $a_0, a_1, ..., a_{N-1}$. Suppose that we wish to focus on the portion of z near some particular point n_0. Terms involving basis vectors that are 0 or negligibly small near n_0 can be deleted from relation (1.5) without changing the behavior near n_0 significantly. Thus we may be able to replace a full sum over N terms by a much smaller sum when considering only the portion of $\mathbf{z}$ near n_0 [12].

More generally, a spatially localized basis for expressing a signal is useful in signal processing because it provides a local analysis of the signal: if a certain coefficient in

the expansion of **z** is large, we can identify the location with which this large coefficient is associated. We could then, for example, focus on this location and analyze it in more detail. One example is to look closely at a potential tumor. Another is radar or sonar imaging, for example in oil prospecting to identify the boundary of an oil pocket, or in archeology to locate artifacts.

Our ultimate goal is to obtain a basis whose elements are both spatially and frequency localized. Then a vector expansion coefficients in this basis will provide both spatial and frequency information. The scaling function for wavelet series expansion can be described as below:

Scaling functions:

Consider the set of expansion functions composed of integer translations and binary scalings of the real, square-integrable function $\phi(x)$; that is, the set $\{\phi_{j,k}(x)\}$where

$$\phi_{j,k}(x) = 2^{\frac{j}{2}}\phi(2^j x - k) \qquad \text{for all } j,k \in \mathrm{Z}.$$

Here, k determines the position of $\phi_{j,k}(x)$ along the x-axis, j determines $\phi_{j,k}(x)$'s width - how broad or narrow it is along the x-axis and the term $2^{j/2}$ controls its height or amplitude [13]. Because the shape of $\phi_{j,k}(x)$ changes with j, $\phi(x)$ is called a scaling function. By choosing $\phi(x)$ wisely, $\{\phi_{j,k}(x)\}$can be made to span $L^2(R)$, the set of all measurable, square-integrable functions.

If we restrict j to a specific value, say $j = j_0$, the resulting expansion set $\{\phi_{j_0,k}(x)\}$, is a subset of $\{\phi_{j,k}(x)\}$. It will not span $L^2(R)$, but a subspace within it. The subspace can be defined as

$$V_{j_0} = span\{\phi_{j_0,k}(x)\}$$

That is , V_{j_0} is the span of $\phi_{j_0,k}(x)$ over k. If $f(x) \in V_{j_0}$, it can be written as

$$f(x) = \sum_k u_k \phi_{j_0,k}(x)$$

More generally, we will denote the subspace spanned over k for any j as

$$V_j = span\{\phi_{j,k}(x)\}$$

The following two subsections demonstrate the formation of the coefficients u and v for Haar and Daubechies wavelets respectively.

1.3.2(a) Haar wavelets

Haar scaling function:

Let $\phi: R \to R$ be defined by

$$\phi(x) = \begin{cases} 1 & x \in [0,1) \\ 0 & otherwise \end{cases}$$

Define $\phi_{j,k} : R \to R$ as (as shown in section 1.3.2)

$$\phi_{j,k}(x) = 2^{\frac{j}{2}} \phi(2^j x - k)$$

Here, $\phi_{j,k}$ is known as the father wavelet.

Define the vector space V^j as

$$V^j = span\{\phi_{j,k}\}_{j,k \in Z}$$

Therefore V^j can be expressed as a linear combination of $\phi_{j,k}$ as follows:

$$V^j = \left\{ \sum_k u_j(k) \phi_{j,k} ; u = (u(k))_{k \in Z} \in l^2(Z) \right\}$$

where $u_j(k)$ is called approximation coefficients at level j.

Since $\phi \in V^0 \subseteq V^1$, the above expression implies that

$\phi(x) = \sum_{k \in Z} u_1(k) \phi_{1,k}(x) = \sum_{k \in Z} u_1(k) \sqrt{2}\, \phi(2x - k)$ and hence ϕ is called scaling function.

Haar wavelet function:

Let $\psi: R \to R$ be defined by

$$\psi(x) = \begin{cases} 1 & x \in [0, 1/2) \\ -1 & x \in [1/2, 1) \\ 0 & otherwise \end{cases}$$

The wavelet function ψ can be expressed in terms of scaling function ϕ as $\psi = \sum_{k \in Z} v_1(k)\phi_{1,k}$.

Define $\psi_{j,k} : R \to R$ as $\psi_{j,k}(x) = 2^{\frac{j}{2}} \psi(2^j x - k)$

Here $\psi_{j,k}$ is known as the mother wavelet.

Define the vector space W^j as

$$W^j = span\{\psi_{j,k}\}_{i,j \in Z}$$

Therefore W^j can be expressed as

$$W^j = \left\{ \sum_k v_j(k)\psi_{j,k} ; v = (v(k))_{k \in Z} \in l^2(Z) \right\}$$

where $v_j(k)$ is called detail coefficients at level j

Since $\phi = \phi_{0,0} \in V^0 \subseteq V^1$, $\phi(x)$ will be expressed as below :

$$\phi(x) = \sum_{k \in Z} u(k)\phi_{1,k}(x)$$

$$= \sum_{k \in Z} u(k)\sqrt{2}\,\phi(2x - k)$$

where $u(k) = \langle \phi, \phi_{1,k} \rangle$ and $u(k) = u_1(k)$

Scaling coefficients $u(k) = \begin{cases} 1/\sqrt{2} & if \quad n = 0 \quad or \quad n = 1 \\ 0 & otherwise \end{cases}$

and wavelet coefficients $v(k) = \begin{cases} 1/\sqrt{2} & if \quad n = 0 \\ -1/\sqrt{2} & if \quad n = 1 \\ 0 & otherwise \end{cases}$

are low pass and high pass filters respectively. It can be seen that $v(k) = (-1)^{k-1}\overline{u(1-k)}$. Where $\overline{u(1-k)}$ is a complex conjugate of $u(1-k)$.

1.3.2(b) Daubechies wavelets

Even though the Haar wavelets are providing local analysis of a signal in frequency as well as in spatial domain, they are not smooth in nature due to their step function behavior. Daubechies has discovered new wavelet bases [11] which overcome the limitation of the Haar wavelets. The formulation of Daubechies wavelets is detailed below:

Since $\phi = \phi_{0,0} \in V_0 \subset V_1$, and the $\phi_{1,n}$ are an orthonormal basis in V_1, we have

$$\phi = \sum_n u_n \phi_{1,n} \tag{1.6}$$

with

$$u_n = \langle \phi, \phi_{1,n} \rangle \quad \text{and} \quad \sum_{n \in Z} |u_n|^2 = 1 \tag{1.7}$$

As defined in the definition of the scaling function ϕ discussed in 1.3.2(b), we can rewrite equation (1.6) as

$$\phi(x) = \sqrt{2} \sum_n u_n \phi(2x - n) \tag{1.8}$$

The fourier transform of (1.8) results in

$$\hat{\phi}(\xi) = \frac{1}{\sqrt{2}} \sum_n u_n e^{-in\xi/2} \hat{\phi}(\xi/2) \tag{1.9}$$

The equation (1.9) can be rewritten as

$$\hat{\phi}(\xi) = m_0(\xi/2)\hat{\phi}(\xi/2) \tag{1.10}$$

where,
$$m_0(\xi) = \frac{1}{\sqrt{2}} \sum_n u_n e^{-in\xi} \tag{1.11}$$

Equality in (1.10) holds pointwise almost everywhere (*a. e.*). As (1.7) shows, m_0 is a 2π-periodic function in $L^2([0, 2\pi])$.

Hence, the orthonormality of the $\phi(. - k)$ leads to special properties for m_0 [11]. We have

$$|m_0(\xi)|^2 + |m_0(\xi + \pi)|^2 = 1 \quad a.e. \tag{1.12}$$

With the help of orthonormality of ϕ and ψ, the equation (1.11) should be of the form [11],

$$m_0(\xi) = \left(\frac{1+e^{-i\xi}}{2}\right)^N L_1(\xi) \tag{1.13}$$

with $N \geq 1$, and L_1- a trigonometric polynomial.

Now considering $M_0(\xi) = |m_0(\xi)|^2$, we find that $M_0(\xi)$ is a polynomial in $cos\xi$, satisfying the property

$$M_0(\xi) + M_0(\xi + \pi) = 1 \tag{1.14}$$

By taking modulus, the equation 1.13 can be written as

$$|m_0(\xi)| = \left|\frac{1+e^{-i\xi}}{2}\right|^N |L_1(\xi)|$$

$$= \frac{1}{2^N}\left[(1+\cos\xi)^2 + \sin^2\xi\right]^{\frac{N}{2}} |L_1(\xi)|$$

$$= \frac{1}{2^N}[2+2\cos\xi]^{\frac{N}{2}} |L_1(\xi)|$$

$$= 2^{\frac{-N}{2}}[1+\cos\xi]^{\frac{N}{2}} |L_1(\xi)|$$

$$= 2^{\frac{-N}{2}}[2\cos^2\frac{\xi}{2}]^{\frac{N}{2}} |L_1(\xi)|$$

$$= [\cos^2\frac{\xi}{2}]^{\frac{N}{2}} |L_1(\xi)|$$

Therefore, $$M_0(\xi) = |m_0(\xi)|^2 = \left(\cos^2\frac{\xi}{2}\right)^N L(\xi)$$

where $L(\xi) = |L_1(\xi)|^2$ is also a polynomial in $cos\xi$. For our purpose it is convenient to rewrite $L(\xi)$ as a polynomial in $\sin^2(\xi/2) = (1-\cos\xi)/2$. Therefore,

$$M_0(\xi) = \left(\cos^2\frac{\xi}{2}\right)^N P\left(\sin^2\frac{\xi}{2}\right)$$

In terms of P, the constraint (1.14) becomes

$$(1-y)^N P(y) + y^N P(1-y) = 1 \tag{1.15}$$

where, $y = \sin^2\left(\frac{\xi}{2}\right)$.

This formula is valid for $y \in [0,1]$, hence for all $y \in R$.

To solve equation (1.15) for P, by using Bezout's theorem [11] there exist a polynomial P_N of degree $\leq (N-1)$ such that the equation takes the form

$$(1-y)^N P_N(y) + y^N P_N(1-y) = 1 \tag{1.16}$$

where,
$$P_N(y) = \sum_{k=0}^{N-1} \binom{N+k-1}{k} y^k \tag{1.17}$$

Daubechies D4 (scaling function 2^{ϕ}) Wavelet

We begin with the equation (1.17) by considering the value of $N=2$.

$$P_2(y) = \binom{1}{0} + \binom{2}{1} y = 1 + 2y$$

and consequently from equation (1.15),

$$P_2(\sin^2 \frac{\xi}{2}) = 1 + 2(\sin^2 \frac{\xi}{2})$$

$$= 2 - \cos\xi = a_0 \cos 0\xi + a_1 \cos\xi = \sum_{k=0}^{n=1} a_k \cos k\xi$$

The following lemma of Riez [32] gives an important relationship among the trigonometric polynomials, which can be stated as below:

Lemma 1.2 (Riez): Let $A(\xi)$ be a positive trigonometric polynomial invariant under the substitution ξ to $-\xi$; A is necessarily of the form

$$A(\xi) = \sum_{k=0}^{n} a_k \cos k\xi, \quad a_k \in R .$$

Then there exists a trigonometric polynomial B of order n ,

$$B(\xi) = \sum_{k=0}^{n} b_k e^{ik\xi}, \quad b_k \in R,$$

such that,

$$A(\xi) \equiv B(\xi)\, B(-\xi) \tag{1.18}$$

identically in ξ.

So, using equation (1.18) in our problem, we get

$$\left(b_0 + b_1 e^{-i\xi}\right)\left(b_0 + b_1 e^{i\xi}\right) = 2 - \frac{1}{2}\left(e^{i\xi} + e^{-i\xi}\right),$$

By simplifying , we get two equations

$$b_0^2 + b_1^2 = 1, \quad b_0 b_1 = -\frac{1}{2}$$

Solution of these two equations leads to $b_0 = \frac{1}{2}\left(1+\sqrt{3}\right), b_1 = \frac{1}{2}\left(1-\sqrt{3}\right).$

Now, using equation (1.13), we can have

$$m_0(\xi) = \left(\frac{1+e^{-i\xi}}{2}\right)^2 L_1(\xi)$$

$$= \frac{1}{4}\left(1+2e^{-i\xi}+e^{-2i\xi}\right)\left(b_0 + b_1 e^{-i\xi}\right)$$

$$= \frac{1}{8}\left(1+2e^{-i\xi}+e^{-2i\xi}\right)\left(1+\sqrt{3}+(1-\sqrt{3})e^{-i\xi}\right)$$

$$= \frac{1}{8}\left(1+\sqrt{3}+(3+\sqrt{3})e^{-i\xi}+(3-\sqrt{3})e^{-2i\xi}+(1-\sqrt{3})e^{-3i\xi}\right)$$

When $m_0(0)=1$ is also satisfied.

By comparing the value of $m_0(\xi)$ of equation (1.11) with the values obtained above, we get

$$\frac{1}{\sqrt{2}}\sum_n u_n e^{-in\xi} = \frac{1}{8}\left(1+\sqrt{3}+(3+\sqrt{3})e^{-i\xi}+(3-\sqrt{3})e^{-2i\xi}+(1-\sqrt{3})e^{-3i\xi}\right)$$

Hence, we get only four nonzero components of approximation coefficient u which are mentioned in table 1. This table demonstrates the scaling (high pass) coefficients $u(k)$ and wavelet (low pass) coefficients $v(k)$ of Daubechies D4 wavelets for scaling function 2^{ϕ} and wavelet function 2^{ψ} respectively.

Table 1. Daubechies D4 low pass and High pass filters

k	$u(k)$	$v(k)=(-1)^{k-1}\overline{u(1-k)}$
0	$\frac{1+\sqrt{3}}{4\sqrt{2}}$	$\frac{1-\sqrt{3}}{4\sqrt{2}}$
1	$\frac{3+\sqrt{3}}{4\sqrt{2}}$	$-\frac{3-\sqrt{3}}{4\sqrt{2}}$
2	$\frac{3-\sqrt{3}}{4\sqrt{2}}$	$\frac{3+\sqrt{3}}{4\sqrt{2}}$
3	$\frac{1-\sqrt{3}}{4\sqrt{2}}$	$-\frac{1+\sqrt{3}}{4\sqrt{2}}$

Daubechies D6 (scaling function 3^{ϕ}) Wavelet

We can obtain low pass and high pass coefficients for Daubechies D6 wavelets for by substituting N=3 in equation (1.17). Table 2. gives a list of the scaling (high pass) coefficients $u(k)$ and wavelet (low pass) coefficients $v(k)$ of Daubechies D6 wavelets for scaling function 3^{ϕ} and wavelet function 3^{ψ} respectively.

Table 2. Daubechies D6 low pass and high pass coefficients

k	$u(k)$	$v(k)=(-1)^{k-1}\overline{u(1-k)}$
0	*0.3326705529500825*	*0.0352262918857095*
1	*0.8068915093110924*	*0.0854412738820267*
2	*0.4598775021184914*	*-0.1350110200102546*
3	*-0.1350110200102546*	*-0.4598775021184914*
4	*-0.0854412738820267*	*0.8068915093110924*
5	*0.0352262918857095*	*-0.3326705529500825*

Thus, table 1 and 2 specify the first stage wavelet basis for Daubechies D4 and D6 respectively.

1.3.3 Series Expansion and Multiresolution Analysis (MRA)

A multiresolution analysis, formulated in the fall of 1986 by Mallat and Meyer, provides a natural framework for the understanding of wavelet bases, and for the construction of new examples. A multiresolution analysis (or MRA) with scaling function φ is a sequence $\{V_j\}_{j\in Z}$ of subspaces of $L^2(R)$ having the following properties:

1. Monotonicity: The sequence is increasing, that is, $V_j \subseteq V_{j+1}$ for all $j \in Z$.
2. Existence of the scaling function: There exists a function $\varphi \in V_0$ such that the set $\{\varphi_{0,k}(x)\}_{k\in Z}$ is orthonormal and

$$V_0 = \left\{ \sum_{k\in Z} z(k)\varphi_{0,k} : z = (z(k))_{k\in Z} \in l^2(Z) \right\}$$

3. Dilation property: For each j, $f(x) \in V_0$ if and only if $f(2^j x) \in V_j$.
4. Trivial intersection property: $\bigcap_{j\in Z} V_j = \{0\}$.
5. Density: $\bigcup_{j\in Z} V_j$ is dense in $L^2(R)$.

MRA plays a major role in the development of series expansions of a signal for the local analysis of that signal. The description of series expansion of a signal using MRA technique of DWT is given below:

A signal or function $f(x)$ can often be better analyzed as a linear combination of expansion functions [13]

$$f(x) = \sum_k \alpha_k \varphi_k \qquad (1.1a)$$

where k is an integer index of the finite or infinite sum, the α_k are real-valued expansion coefficients, and the $\varphi_k(x)$ are real–valued expansion functions. If the expansion is unique-that is, there is only one set of α_k for any given $f(x)$ - the $\varphi_k(x)$ are called basis functions, and expansion set, $\{\varphi_k(x)\}$, is called a basis for the class of functions that can be so expressed. The expressible functions form a function space that is referred to as the span of the expansion set, denoted

$$V = Span\{\varphi_k(x)\}. \qquad (1.2a)$$

Therefore, any $f(x) \in V$ can be written in the form of Equation (1.1a).

For any function space V and corresponding expansion set $\{\varphi_k(x)\}$, there is a set of dual functions, denoted $\{\tilde{\varphi}_k(x)\}$, that can be used to compute the α_k coefficients of equation (1.1a) for any $f(x) \in V$. These coefficients are computed by taking the inner product of the dual $\tilde{\varphi}_k(x)$'s and function $f(x)$. That is,

$$\alpha_k = \langle \tilde{\varphi}_k(x), f(x) \rangle \tag{1.3a}$$

Depending upon the orthogonality of the expansion set, this computation assumes one of three possible forms.

Case 1: If the expansion functions form an orthonormal basis for V, meaning that

$$\langle \varphi_j(x), \varphi_k(x) \rangle = \delta_{jk} = \begin{cases} 0 & j \neq k \\ 1 & j = k \end{cases} \tag{1.4a}$$

the basis and its dual are equivalent. That is, $\varphi_k(x) = \tilde{\varphi}_k(x)$ and equation (1.3a) becomes

$$\alpha_k = \langle \varphi_k(x), f(x) \rangle . \tag{1.5a}$$

Case 2: If the expansions are not orthonormal, but are an orthogonal basis for V, then

$$\langle \varphi_k(x), \varphi_j(x) \rangle = 0 \qquad j \neq k$$

and the basis functions and their duals are called biorthogonal. The α_k are computed using equation (1.3a), and the biorthogonal basis and its dual are such that

$$\langle \varphi_j(x), \tilde{\varphi}_k(x) \rangle = \delta_{jk} = \begin{cases} 0 & j \neq k \\ 1 & j = k \end{cases}$$

Case 3: If the expansion set is not a basis for V, but supports the expansion defined in equation (1.1a), it is spanning set in which there is more than one set of α_k for any set $f(x) \in V$. The expansion functions and their duals are said to overcomplete or redundant. They form a frame [11] in which

$$A\|f(x)\|^2 \leq \sum |\langle \varphi_k(x), f(x) \rangle|^2 \leq B\|f(x)\|^2 \tag{1.6a}$$

for some $A > 0$, $0 < B < \infty$, and all $f(x) \in V$. The norm, $\|\cdot\|$, of $f(x)$, is defined as the square root of the inner product of $f(x)$ with itself.

Dividing this equation by the norm squared of $f(x)$, we see that A and B "frame" the normalized inner products of the expansion coefficients and the function. If $A = B$, the expansion set is called a tight frame and it can be shown that

$$f(x) = \frac{1}{A}\sum_k \langle \varphi_k(x), f(x) \rangle \varphi_k(x) \tag{1.7a}$$

Except for the A^{-1} term, which is a measure of the frame's redundancy, this is identical to the expression obtained by substituting equation (1.5a) (for orthonormal basis) into equation (1.1a).

Let V^1 be an inner product space and *let* V^0 be subspace of the space V^1. Let W^0 be an orthogonal compliment of V^0 in V^1 so that W^0 is also a subspace of V^1. Then $V^0 \oplus W^0 = \{v_0 + w_0, v_0 \in V^0 \text{ and } w_0 \in W^0\}$ is called the orthogonal direct sum of V^0 and W^0.

In particular, if we say $V^0 \oplus W^0 = V^1$, we mean that V^0 and W^0 are subspaces of V^1, $V^0 \oplus W^0$ and every element of $x \in V^1$ can be written as $x = u + v$ for some $u \in V^0$ and $v \in W^0$.

$$\therefore V^1 = V^0 \oplus W^0 \tag{1.19}$$

We define the wavelet series expansion of function $f(x) \in L^2(R)$ relative to wavelet $\psi(x)$ and scaling function $\phi(x)$ using first stage wavelet basis as below:

$$f(x) = \phi_{1,k}(x) = \sum_k u_0(k)\phi_{0,k}(x) \quad + \quad \sum_k v_0(k)\psi_{0,k}(x)$$

Equation (1.19) can be generalized using the first and fourth properties of MRA (stated above) as $V^{j+1} = V^j \oplus W^j$ and form a Multiresolution Analysis [12, 13]. Therefore, the above equation will take the following form

$$f(x) = \sum_k u_{j_0}(k)\phi_{j_0,k}(x) \quad + \quad \sum_j \sum_k v_j(k)\psi_{j,k}(x) \quad \text{(see fig. 1)} \tag{1.20}$$

where j_0 is an arbitrary integer called the starting scale.

The coefficients u_j and v_j in this expansion are called the approximation and detail coefficients respectively.

The computational aspect of equation (1.20) can be stated by the following lemma [12]:

Lemma 1.3: Suppose $\{V^j\}_{j\in Z}$ is a Multi resolution Analysis (MRA) with scaling function φ and scaling sequence $u=(u_j(k))_{k\in z}$. Suppose $v=(v_j(k))_{k\in z}$ is defined by $v(k)=(-1)^{k-1}\overline{u(1-k)}$, and $\psi=\sum_{k\in Z} v(k)\varphi_k^j$, where $\phi_k^j=2^{j/2}\phi(2^j-k)$. Suppose $f\in L^2(R)$ and, for each $j\in z$, define sequences $x_j=(x_j(k))_{k\in z}$ and $y_j=(y_j(k))_{k\in z}$ by

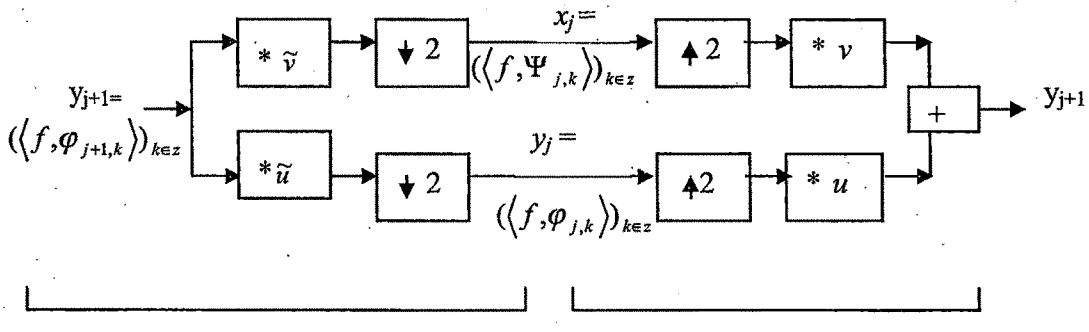
$$x_j(k)=\langle f,\Psi_k^j\rangle \text{ where } \psi_k^j=2^{j/2}\psi(2^j-k) \text{ and } y_j(k)=\langle f,\psi_k^j\rangle.$$

Then $x_j=D(y_{j+1} * \tilde{v})$ and $y_j=D(y_{j+1} * \tilde{u})$, where downsampling operator *(D)* and convolution (*) are on $l^2(z)$ and $\tilde{u}(k)=u(-k)$ *and* $\tilde{v}(k)=v(-k)$ are duals of u(approximation coefficients) and v(detail coefficients) respectively by considering approximation coefficients $u=u_j$ and detail coefficients $v=v_j$.

Also
$$y_{j+1}=U(y_j)*u+U(x_j)*v \tag{1.21}$$

where U is the upsampling operator on $l^2(z)$.

The computation described by the above lemma is pictorially represented by the following diagram:



Analysis phase Synthesis phase

[Fig. 1.5 analysis and synthesis phase of a signal]

1.4. Artificial Neural Networks

Work on artificial neural networks, commonly referred to as "neural networks", has been motivated right from its inception by the recognition that the human brain computes in an entirely different way from the conventional digital computer. The brain is a highly complex, nonlinear and parallel computer (information-processing system). It has the capability to organize its structural constituents, known as neurons, so as to perform certain computations (e.g. pattern recognition, perception, and motor control) many times faster than the fastest digital computer in existence today [18].

To visualize the complexity of biological neural processing, consider the *sonar* of a bat. Sonar is an active echo-location system. In addition to providing information about how far away a target (e.g. a flying insect) is, a bat sonar conveys information about the relative velocity of the target, the size of various features of the target, and the azimuth and elevation of the target. The complex neural computations needed to extract all this information from the target echo occur within the bat's brain having the size of a plum.

Artificial Neural networks are computational algorithms that can broadly be defined as follows:

A neural networks is a massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

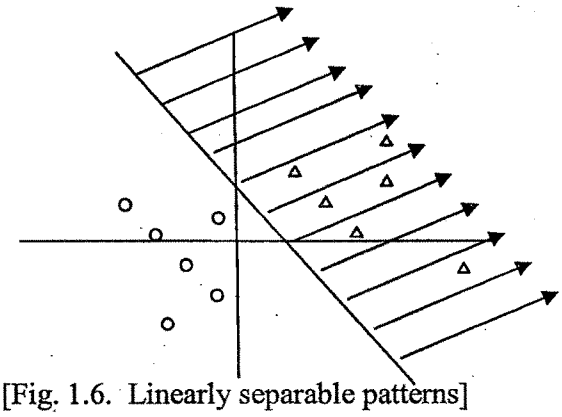
1. Knowledge is acquired by the network from its environment through a learning process
2. Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

Processing units are known as neurons.

ANN can be applicable in several areas like Pharmaceutical problems [2], medical diagnosis, weather forecasting etc. The work to be presented in the thesis explores the use of various ANN architectures for the purpose of classifying the symbols that occur in Gujarati language.

The history of ANN starts from 1946, McCulloh Pits have developed the first neural network model which simply takes binary inputs and computes the output using only one neuron in the hidden layer. Hard limit function was used as an activation function in the hidden layer.

Some 15 years after the publication of McCulloch and Pitt's classic paper, a new approach to the pattern recognition problem was introduced by Rosenblatt (1958). He has proposed the perceptron as the first model for learning with teacher (i.e. supervised learning). The perceptron is the simplest form of a neural network used for the classification of patterns said to be linearly separable (figure 1.6) (i.e., patterns that lie on opposite sides of a hyperplane). Rosenblatt proved that if the patterns used to train the perceptron are drawn from two linearly separable classes, then the perceptron algorithm converges and positions the decision surface in the form of hyperplane between the two classes. The algorithm is trained with the help of Least Mean Square (LMS) algorithm. LMS algorithm behaves like a *low-pass filter*, passing the low frequencey components of the error signal and attenuating its high frequency components (Haykin, 1996).



[Fig. 1.6. Linearly separable patterns]

The proof of convergence of the algorithm is known as Perceptron Convergence Theorem. The Perceptron Convergence Theorem is stated as below:

Let the subsets of training vectors X_1 and X_2 be lineraly sepearble. Let the inputs presented to the perceptron originate from these two subsets. The perceptron converges after some n_0 iterations, in the sense that

$$w(n_0)=w(n_0+1)=w(n_0+2)=...$$

is a solution vector for $n_0 \leq n_{max}$

Perceptron Convergence Theorem converges after some n_o iterations provided the training vectors are linearly separable. The limitation of Rosenblatt's model of Perceptron is it does not converge for the non separable training vectors.

In 1986 the development of the back-propagation algorithm was reported by Rumelhart, Hinton, and Williams which is playing a major in the most popular learning algorithm for the training of Multilayer Perceptron Significant (MLP). Also in 1988, Broomhead and Lowe described a procedure for the design of layered feedforward networks using Radial Basis Function (RBF), which provide an alternative to MLP. Specht D. F. in 1991, has brought the idea of General Regression Neural Networks (GRNN). The theory is based on non-parametric estimator of Statistics and estimates the output without any kind of training. In the early 1990s, Vapnik and Coworkers invented a computationally powerful class of supervised learning networks, called Support Vector Machine (SVM).

The description of MLP network is to be discussed in the next section of this chapter while in the fourth chapter MLP is used as a classifier for the development of Gujarati OCR. The Mathematical and computational aspects (algorithmic aspects) of General Regression Neural Network will be presented in chapter 5. While kernel based techniques Radial Basis Function and Support Vector Machine, in which the classification of the patterns is based on function-separable space, will be described in the 6^{th} chapter of the thesis.

Now, in the following two subsections we describe two widely used architectures of ANN viz. Multilayer Perceptron networks (MLP) and Radial Basis Function networks (RBFN).

1.4.1. Multilayer Perceptron (MLP):

The MLP is the most widely used neural network architecture. Typically, the network consists of a set of sensory units (source nodes) that constitute the input layer, one or more hidden layers of computation nodes, and an output layer of computation nodes. The input signal propagates through the network in a forward direction, on a layer-by-layer basis. These neural networks are commonly referred to as multilayer perceptron (MLPs).

Multiplayer perceptrons have been applied successfully to solve some difficult and diverse problems by training them in a manner with a highly popular algorithm known as the error backpropagation algorithm is based on the error-correction learning rule.

Basically, error back–propagation learning consists of two passes through the different layers of the networks: a forward pass and backward pass .In the forward pass , an activity patterns (input vector) is applied to the sensory nodes of the network, and its effect propagates through the layer by layer . Finally, a set of output is produced as the actual response of the networks. During the forward pass the synaptic weights of the networks are all fixed. During the backward pass, on the other hand , the synaptic weights are all adjusted in accordance with an error-correction rule. Specifically, the actual response of the networks is subtracted from a desired (target)response to produce an error signal. This error signal is then propagated backward through the networks against the direction of synaptic connections-hence the name "error back –propagation". The synaptic weights are adjusted to make the actual response of the network move closer to the desired response in a statistical sense. The error back-propagation algorithm is also referred to in the literature as the back-propagation algorithm, or simply back –prop .Henceforth we will refer to it as

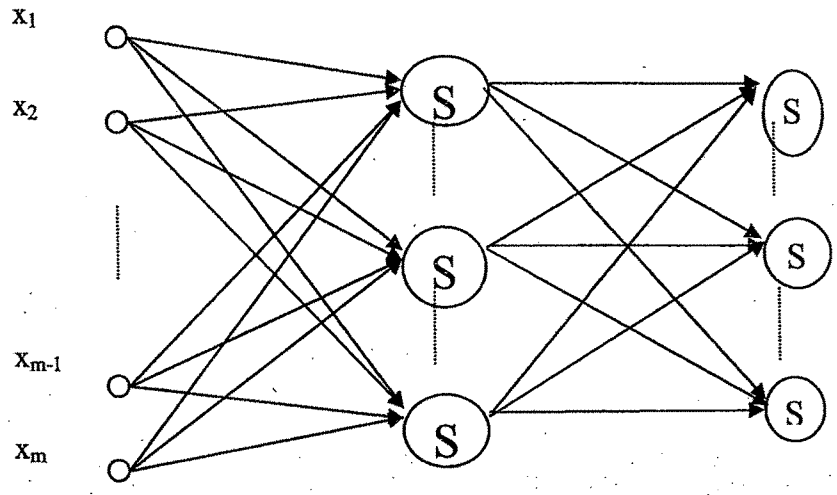
the back-propagation algorithm. The learning process performed with the algorithm is called back-propagation learning.

Figure 1.7 shows the architectural graph of a multilayer perceptron with one hidden layer and an output layer. To set the stage of a description of the multilayer perceptron in its general form, the network shown here is fully connected. This means that a neuron in any layer of the network is connected to all the nodes/neurons in the previous to right and on a layer-by-layer basis.

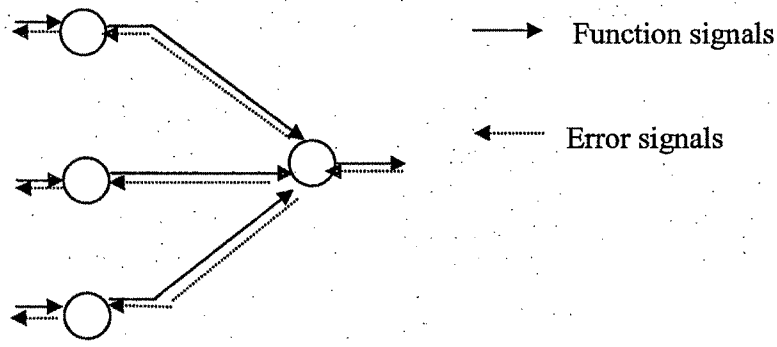
Figure 1.8 depicts a portion of the multilayer perceptron. Two kinds of signals are identified in this network:

1. *Function Signals:* A function signal is an input signal (stimulus) that comes in at the input end of the network, propagates forward (neuron by neuron) through the network, and emerges at the output end of the network as an output signal. We refer to such a signal as a "function signal" for two reasons. First, it is presumed to perform a useful function at the output of the network. Second, at each neuron of the network through which a function signal passes, the signal is calculated as a function of the inputs and associated weights applied to that neuron. The function signal is also referred to as the input signal.

2. *Error Signals*: An error originates at an output neuron of the network, and propagates backward (layer by layer) through the network. We refer to it as an "error signal" because its computation by every neuron of the network involves an error-dependent function in one form or another.



[Fig.1.7 Architecture of a multilayer perceptron]



[Fig. 1.8. forward propagation of function signals and back-propagation of error-signal]

Determining the number of hidden layers and the number of neurons in these hidden layers for a given problem in the case of MLP networks are very critical decisions for applications involving large networks. In such conditions following theorem leads to important information regarding hidden layers.

The Universal Approximation theorem for Multilayer perceptron is as under[18]:

"Let $\varphi(.)$ be a nonconstants, bounded and monotone-increasing continuous function. Let I_{m_0} denote the m_0-dimensional unit hypercube $[0,1]^{m_0}$. The space of continuous

functions on I_{m_0} is denoted by $C(I_{m_0})$. Then, given any function $f \in C(I_{m_0})$ and $\varepsilon > 0$, there exist an integer M and sets of real constants α_i, b_i, and w_{ij}, where i = 1,...,m_0 such that we may define

$$F(x_1, x_2, ...x_{m_0}) = \sum_{i=1}^{m_1} \alpha_i \varphi(\sum_{j=1}^{m_0} w_{ij} x_j + b_i) \qquad (1.22)$$

as an approximate realization of the function $f(\cdot)$; that is,

$$\left| F(x_1, x_2, ...x_{m_0}) - f(x_1, x_2, ...x_{m_0}) \right| < \varepsilon$$

for all $x_1, x_2, ...x_{m_0}$ that lie in the input space"

The universal approximation theorem is directly applicable to multilayer perceptrons. We first note that the logistic function (sigmoid) used as the nonlinearity in a neuronal model for the construction of a MLP is indeed a nonconstant, bounded and monotone-increasing function; it therefore satisfies the conditions imposed on the function $\varphi(.)$. Next, we note that equation (1.22) represents the output of a MLP described as follows:

(i) The network has m_0 input nodes and a single hidden layer consisting of m_1 neurons; the inputs are denoted by $x_1, x_2, ...x_{m_0}$

(ii) Hidden neuron i has synaptic weights $w_{i1}, w_{i2}, ...w_{m_0}$ and bias b_i

(iii) The network output is a linear combination of the outputs of the hidden neurons with $\alpha_1, \alpha_2, ...\alpha_{m_1}$, defining the synaptic weights of the output layer.

The universal approximation theorem is an existence theorem in the sense that it provides the mathematical justification for the approximation of an arbitrary continuous function as opposed to exact representation. Theorem states that a single hidden layer is sufficient for a multilayer perceptron to compute a uniform ε approximation to a given training set represented by the set of inputs x_1, x_2, ..., x_{m0} and a desired (target) output $f(x_1, x_2, ..., x_{m0})$.

The usage of the term "back-propagation" appears to have evolved after 1985, when its use was popularized through the publication of the seminal book entitled Parallel Distributed Processing (Rumelhart and McClelland, 1986). Back-propagation

algorithm is playing an important role in the development of Multilayer perceptron classification technique of ANN. The algorithm can be summarized as below:

Error-Backpropagation algorithm:

Consider the N training samples $\{(\mathbf{X}(n), \mathbf{D}(n))\}_{n=1}^{N}$ where $\mathbf{X}(n)$ and $\mathbf{D}(n)$ are input and desired output vectors . The algorithm can broadly be described as follows:

1. *Initialization:* Assuming that no prior information is available, pick the synaptic weights and thresholds from a uniform distribution whose mean is zero and whose variation is chosen to make the standard deviation of the induced local fields of the neurons lie at the transition between the linear and saturated parts of the sigmoid activation function.

2. *Presentations of Training Examples*: Present the network with an epoch of training examples. For each example in the set, ordered in some fashion, perform the sequence of forward and backward computations described above.

3. *Forward Computation*: Let a training example in the epoch be denoted by $(\mathbf{X}(n), \mathbf{D}(n))$, with the input vector **X**(n) applied to the input layer of sensory nodes and the desired response vector **D**(n) presented to the output layer of computation nodes. Compute the induced local fields and function signals of the network by proceeding forward through the network, layer by layer. The induced field $v_j^{(l)}(n)$ for neuron j in layer l is

$$v_j^{(l)}(n) = \sum_{i=0}^{m_0} w_{ji}^{(l)} y_i^{(l-1)}(n)$$

where $y_i^{(l-1)}(n)$ is the output (function) signal of neuron i in the previous $l-1$ at iteration n and $w_{ji}^{(l)}(n)$ is the synaptic weight of neuron j in layer l that is fed from neuron i in layer $l-1$. The computed output of $(l\text{-}1)^{th}$ layer can be given by $y_0^{(l-1)}(n) = +1$ for some particular i=0, and the corresponding weight vector of the link from j^{th} neuron of the l^{th} layer to 0^{th} neuron of the $(l-1)^{th}$ layer can be given

by $w_{j_0}^{(l)}(n) = b_j^{(l)}(n)$. Assuming the use of a sigmoid function, the output signal of neuron j in layer l is

$$y_j^{(l)} = \varphi\left(v_j(n)\right)$$

If neuron j is in the first hidden layer (i.e. $l = 1$), set

$$y_j^{(1)} = x_j(n)$$

where $x_j(n)$ is the j^{th} element of the input vector $\mathbf{X}(n)$. If neuron j is in the output layer (i.e. $l = L$, where L is referred to as the depth of the network), set

$$y_j^{(l)} = o_j(n)$$

Compute the error signal

$$e_j(n) = d_j(n) - o_j(n)$$

where $d_j(n)$ is the j^{th} element of the desired response vector $\mathbf{D}(n)$.

4. *Backward Computation*: Compute the δ_s (i.e. local gradient) of the network defined by

$$\delta_j^{(l)}(n) = \begin{cases} e_j^{(L)}(n)\varphi_j'\left(v_j^{(L)}(n)\right) & \textit{for neuron } j \textit{ in output layer } L \\ \varphi_j'\left(v_j^{(l)}(n)\right)\sum_k \delta_k^{(l+1)}(n) w_{kj}^{(l+1)}(n) & \textit{for neuron } j \textit{ in hidden layer } l \end{cases}$$

where the prime $\varphi_j'(.)$ denotes differentiation with respect to the argument. For the necessary correction in the synaptic weights, the generalized delta rule can be used as shown below:

The total error energy $\xi(n)$ is obtained by $\xi(n) = \frac{1}{2}\sum_{j \in C} e_j^2(n)$, where set C includes all the neurons in the output layer of the network. The correction $\Delta w_{ji}(n)$ applied to $w_{ji}(n)$ in the l^{th} layer is denoted as $\Delta w_{ji}^l(n)$ and defined by the delta rule

$$\Delta w_{ji}^l(n) = -\eta \frac{\partial \xi(n)}{\partial w_{ji}^l(n)}, \qquad \text{where } \eta \text{ is a learning rate parameter}$$

Therefore the correction in the links of the synaptic weights connecting neuron j in layer l and neuron i in layer $l-1$ is given by

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \Delta w_{ji}^{(l)}(n)$$

$$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) - \eta \frac{\partial \xi(n)}{\partial w_{ji}^{l}(n)}$$

Therefore, $$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \eta \delta_{j}^{(l)}(n) y_{i}^{(l-1)}(n)$$

4. *Iteration*: Iterate the forward and backward computations under points 3 and 4 by presenting new epochs of training examples to the network the absolute rate of change in the average squared error per epoch is sufficiently small.

1.4.2. Radial Basis Function Networks (RBFN) :

The design of RBFN can be viewed as a curve fitting (approximation) problem in a high-dimensional space. In the context of neural networks, the hidden units provide a set of "functions" that constitute an arbitrary "basis" for the input patterns (vectors) when they are expanded into the hidden space; these functions are called radial-basis functions.

RBFN have following two major properties which make it suitable for our classification problems:

- A pattern-classification problem cast in a high dimensional space is more likely to be linearly separable than in a low-dimensional space (Cover-1965)
- RBF networks using exponentially decaying localized nonlinearities (e.g. Gaussian functions) construct *local* approximations to nonlinear input-output mappings.

Cover's Theorem on the Separability of Patterns:

When a radial basis function network is used to perform a complex pattern-classification task, the problem is basically solved by transforming it into a high dimensional space in a nonlinear manner. The Cover's theorem on the separability of patterns can be stated as below:

"A complex pattern-classification problem cast in a high-dimensional space nonlinearly is more likely to be linearly separable than in a low-dimensional space"

Consider a family of surfaces where each naturally divides an input space into two regions. Let χ denote a set of N patterns (vectors) $\mathbf{X}_1,\mathbf{X}_2,\ldots,\mathbf{X}_N$, each of which is assigned to one of two classes χ_1 and χ_2. This dichotomy (binary partition) of the points is said to be separable with respect to the family of surfaces if a surface exists in the family that separates the points in the class χ_1 from those in the class χ_2. For each pattern $\mathbf{X}\in\chi$, define a vector made up of a set of real-valued functions $\{\varphi_i(\mathbf{X})|i=1,2,\ldots,m_1\}$, as shown by

$$\varphi(\mathbf{X})=[\varphi_1(\mathbf{X}), \varphi_2(\mathbf{X}), \ldots, \varphi_{m1}(\mathbf{X})]^T$$

Suppose that the pattern $\mathbf{X}$ is a vector in an m_0-dimensional input space into corresponding points in a new space of dimension m_1. We refer to $\varphi_i(\mathbf{X})$ as a hidden function, because it plays a role similar to that of a hidden unit in a feedforward neural network. Correspondingly, the space spanned by the set of hidden functions $\{\varphi_i(\mathbf{X})\}_{i=1}^{m_1}$ is referred to as the hidden space or feature space.

A dichotomy $[\chi_1, \chi_2]$ of χ is said to be φ-separable if there exists an m_1-dimensional vector w such that we may write

$$w^T\varphi > 0,\ \mathbf{X}\in\chi_1$$
$$w^T\varphi < 0,\ \mathbf{X}\in\chi_2$$

The hyperplane defined by the equation

$$w^T\varphi=0$$

describes the separating surface in the φ-space (i.e., hidden space).

To illustrate the significance of the idea of φ-separablity of patterns, consider XOR problem. In the XOR problem there are four points (patterns): (1, 1), (0, 1), (0, 0) and

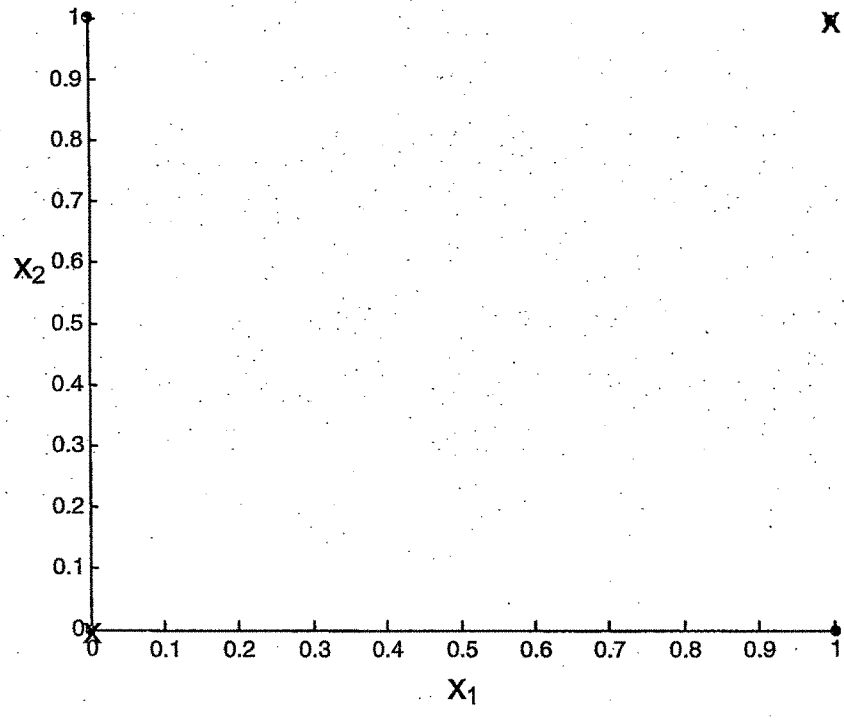
(1, 0), in a two-dimensional input space, as depicted in Fig. 1.9. The requirement is to construct a pattern classifier that produces the binary output 0 in response to the input patterns (1, 1) or (0, 0) and the binary output 1 in response to the input pattern (0, 1) or (1, 0).

Define a pair of Gaussian hidden functions as follows:

$$\phi_1(\mathbf{X}) = e^{-\|\mathbf{X}-\mathbf{t}_1\|^2}, \quad \mathbf{t}_1 = [1,1]^T$$

$$\phi_2(\mathbf{X}) = e^{-\|\mathbf{X}-\mathbf{t}_2\|^2}, \quad \mathbf{t}_2 = [0,0]^T$$

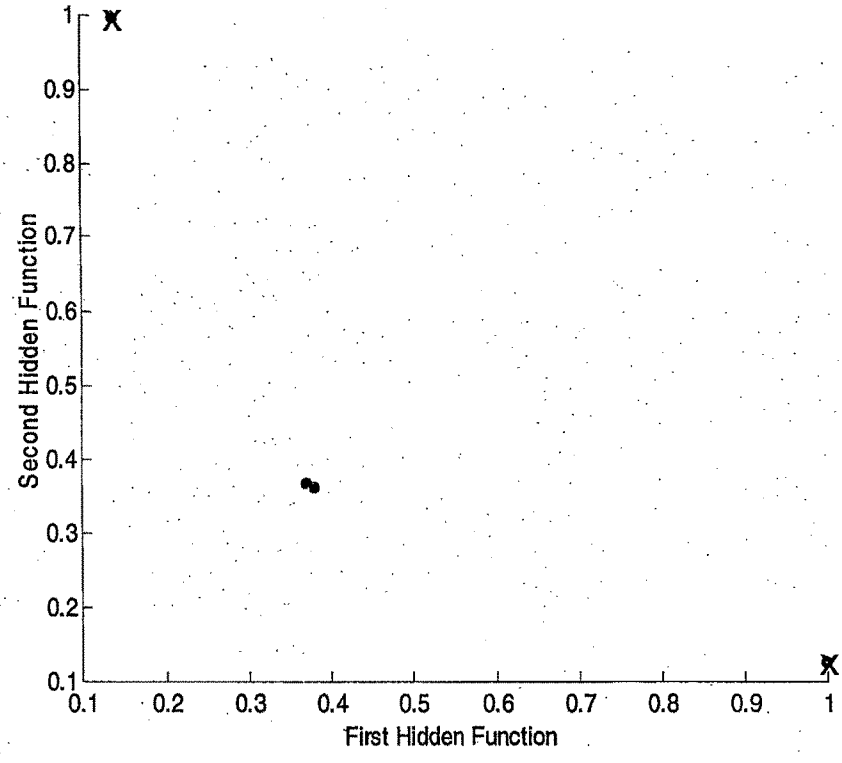
where the norm used here is the Euclidean norm.



[Fig. 1.9. Linearly nonseparable in X_1-X_2 plane]

Table 3. Specification of the Hidden Functions for XOR

Input Pattern, X	**First Hidden Function, $\phi_1(X)$**	**Second Hidden Funciton, $\phi_2(X)$**
(1,1)	1	0.1353
(0,1)	0.3678	0.3678
(0,0)	0.1353	1
(1,0)	0.3678	0.3678



[Fig. 1.10. Linearly separable in φ_1-φ_2 plane]

The input patterns are mapped onto the φ_1-φ_2 plane as shown in figure 1.10. Here we observe that the input patterns (0, 1) and (1, 0) are linearly separable from the remaining patterns (1, 1) and (0, 0). The functional relationship between the input and output pairs is sometimes referred to an interpolation problem, which may be described as below:

Interpolation Problem

In a practical situation, the surface Γ is unknown and the training data are usually contaminated with noise. The training and generalization phase of the learning process may be respectively viewed as follows (Broomhead and Lowe, 1988):

- The training phase constitutes the optimization of a fitting procedure for the surface Γ, based on known data points presented to the network in the form of input-output patterns.
- The generalization phase is synonymous with interpolation between the data points, with the interpolation being performed along the constrained surface generated by the fitting procedure as the optimum approximation to the true surface Γ.

Thus we are led to the theory of multivariable interpolation in high-dimensional space. The interpolation problem may be stated as follows:

Given a set off N different points $\{\mathbf{x}_i \in R^{m_0} \mid i = 1,2,...,N\}$ and a corresponding set of N real numbers $\{d_i \in R^1 \mid i = 1,2,...,N\}$, find a function $F : R^N \rightarrow R^1$ that satisfies the interpolation condition:

$$F(\mathbf{x}_i) = d_i, \qquad i = 1,2,\ldots,N$$

For the strict interpolation, the interpolating surface (i.e., function F) is constrained to pass through all the training data points.

The radial-basis functions (RBF) technique consists of choosing a function $y = F(\mathbf{X})$ that has the following form (Powell, 1988):

$$y(\mathbf{X}) = \sum_{i=1}^{N} \mathbf{w}_i \varphi\left(\left\|\mathbf{X} - \mathbf{t}_i\right\|\right) \tag{1.22}$$

where $\{\varphi(\|\mathbf{X} - \mathbf{X}_i\|) \mid i = 1,2,...,N\}$ is a set of N arbitrary nonlinear functions, known as radial basis functions, and $\|\ .\ \|$ denotes a norm that is usually Euclidean. The known data points $\mathbf{X}_i \in R^{m_0}$, i = *1,2,...,N* are taken to be the centers of the radial basis functions and $\{\mathbf{w}_i\}$ are the unknown coefficients (weights).

Inserting the interpolation conditions of the interpolation problem in the equation (1.22), we obtain the following set of simultaneous linear equations:

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1N} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2N} \\ \vdots & \vdots & & \vdots \\ \varphi_{N1} & \varphi_{N2} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix}$$

where $\varphi_{ji} = \varphi(x_j - x_i) \qquad j,i = 1,2,...,N$

Let $\mathbf{\Phi}$ denote an N x N matrix with elements φ_{ji}:

$$\mathbf{\Phi} = \{\varphi_{ji} \mid (j,i) = 1,2,...,N\}$$

In vector form the above matrix equation can be expressed as :

$$\mathbf{\Phi w} = \mathbf{d}$$

where N x 1 vectors **d** and **w** represent the desired response vector and linear weight vector, respectively, where N is the size of the training sample.

Assuming that $\mathbf{\Phi}$ is nonsingular and therefore that the inverse matrix $\mathbf{\Phi}^{-1}$ exists, we may go on to solve for the weight vector **w** as shown below:

$$\mathbf{w} = \mathbf{\Phi}^{-1}\mathbf{d}$$

In order to ensure the nonsingularity of the interpolation matrix $\mathbf{\Phi}$, we may lead to the following theorem, known as Micchelli's theorem.

Micchelli's Theorem

In the article of Micchelli's (1986), the following theorem regarding the interpolation matrix $\mathbf{\Phi}$ is proved:

Let $\{\mathbf{X}_i\}_{i=1}^{N}$ be a set of distinct points in R^{m_0}. Then the N x N interpolation matrix φ, whose ji-th element is $\varphi_{ji} = \varphi(\| \mathbf{X_j} - \mathbf{X_i} \|)$, is nonsingular

There is a large class of radial basis functions that is covered by Micchelli's theorem: it includes the following functions are of particular interest in the study of RBF networks:

a. Multiquadrics:

$\varphi(r) = (r^2 + c^2)^{1/2}$ for some $c > 0$ and $r \in R$

b. Inverse multiquadrics:

$\varphi(r) = (r^2 + c^2)^{-1/2}$ for some $c > 0$ and $r \in R$

c Gaussian functions:

$$\varphi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \text{ for some } \sigma > 0 \text{ and } r \in R$$

For the radial basis functions listed above to be nonsingular, the points $\{x_i\}_{i=1}^{N}$ must all be different (i.e., distinct). That is all that is required for nonsingularity of the interpolation matrix Φ, whatever the values of size N of the data points.

The Radial basis function networks can be categorized in to the two networks viz. Regularization networks and Generalized network. Both the networks are introduced in the following subsections.

(a) Regularization network

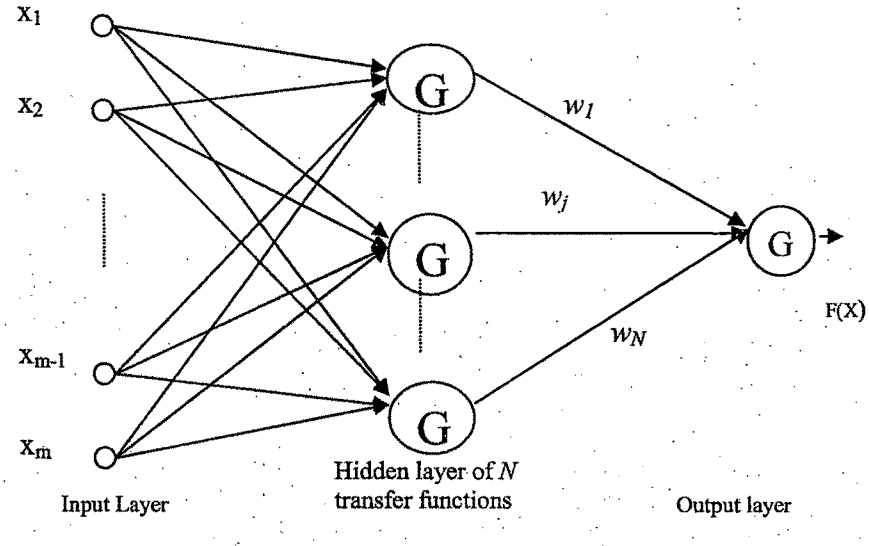
The regularization network is a universal approximator in that it can approximate arbitrarily well any multivariate continuous function on a compact subset of R^{m_0}, given a sufficiently large number of hidden units. The regularization network architecture can be described as below:

Equation 1.22 can be written by using Gaussian function as a transfer function as below:

$$y(\mathbf{X}) = \sum_{i=1}^{N} \mathbf{w}_i \exp\left(-\frac{\|\mathbf{X} - \mathbf{t}_i\|^2}{2\sigma_i^2}\right)$$

The above equation states the following (Poggio and Girosi, 1990):

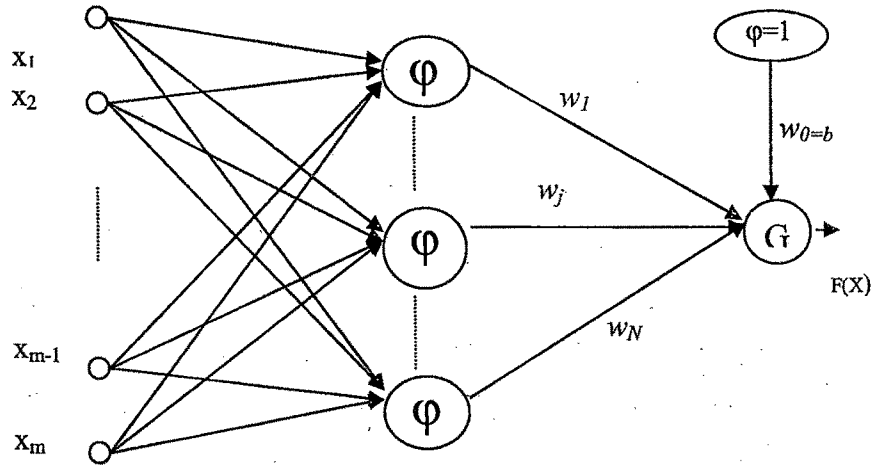
- The regularization approach is equivalent to the expansion of the solution in terms of a set of nonlinear functions mentioned above.
- The number of such functions used in the expansion is equal to the number of examples used in the training process.



[Fig.1.11 Regularization RBF network]

(b) Generalized Radial Basis Function networks

The one-to-one correspondence between the training input data $\mathbf{X}_i$ and the Green's function[18] as a transfer function $G(\mathbf{X},\mathbf{X}_i)$ *for* $i = 1,2,...,N$ produces a regularization network (figure 1.11) that may sometimes be considered prohibitively expensive to implement in computational terms for large N, specifically, the computation of the linear weights of the network. Furthermore, the likelihood of ill conditioning is higher for larger matrices; the condition number of a matrix is defined as the ratio of the largest eigen value to the smallest eigen value of the matrix. To overcome these computational difficulties, the complexities of the network would have to be reduced, which requires an approximation to the regularized solution. This kind of the form of RBF network is known as Generalized Radial-Basis Function networks (figure 1.12).



[Fig.1.12 Generalized RBF network]

The approach taken involves searching for a suboptimal solution of equation (1.22). This is done by using a standard technique known in variational problems as Galerkin's method. According to this technique, the approximated solution $F^*(\mathbf{X})$ is expanded on a finite basis, as shown by

$$F^*(\mathbf{X}) = \sum_{i=1}^{m_1} \mathbf{w}_i \varphi_i(\mathbf{X}) \tag{1.23}$$

where $\{\varphi_i(\mathbf{X}) | i=1,2,...m_1\}$ is a new set of basis functions that we assume to be linearly independent without loss of generality. Typically, the number of basis functions is less than the number of data points (i.e. $m_1 \quad N$, and the w_i constitute a new set of weights). With radial-basis functions in mind, we set

$$\varphi_i(\mathbf{X}) = G(\|\mathbf{X}-\mathbf{t}_i\|), \quad i=1,2,...m_1 \tag{1.24}$$

where, the set of centers $\{\mathbf{t}_i | i=1,2,3,...,m_1\}$ is to be determined.

Using equation 1.24 in 1.23, we may redefine $F^*(X)$ as

$$F^*(\mathbf{X}) = \sum_{i=1}^{m_1} \mathbf{w}_i G(\mathbf{X}, \mathbf{t}_i)$$

$$= \sum_{i=1}^{m_1} \mathbf{w}_i G(\|\mathbf{X}-\mathbf{t}_i\|) \tag{1.25}$$

To fit the training data, we require that

$$F^*(\mathbf{X}_j) = d_j \quad , j = 1,2,...,N$$

where X_j is an input vector and d_j is the corresponding value of the desired response. In a matrix form equation (1.25) can be expressed as

$$\mathbf{Gw} = \mathbf{d} \tag{1.26}$$

where **G** is a matrix of dimension $N \times m_1$, **w** is $m_1 \times 1$ dimensional and the desired output vector **d** is N-dimensional.

From equation (1.26), the weight vector can be computed by multiplying the psudoinverse of the matrix **G** with the vector of desired response **d.**

For instance, consider the XOR problem discussed above by taking number of centers $2 < N$ ($N = 4$, total number of training patterns).

To fit the training data of table 3, we require that

$$y(\mathbf{X}_j) = d_j, \qquad j = 1,2,3,4$$

Where $\mathbf{X}_j$ is an input vector and d_j is the corresponding value of the desired output

Then, using the values of table 3 in equation in 1.26, we get the following set of equations written in matrix form

$$\mathbf{Gw} = \mathbf{d}$$

where

$$\mathbf{G} = \begin{bmatrix} 1 & 0.1353 & 1 \\ 0.3678 & 0.3678 & 1 \\ 0.1353 & 1 & 1 \\ 0.3678 & 0.3678 & 1 \end{bmatrix}$$

$$\mathbf{d} = [0 \quad 1 \quad 0 \quad 1]^T$$

$$\mathbf{w} = [w \quad w \quad b]^T$$

The problem described here is overdetermined in the sense that we have more data points that free parameters. This explains why the matrix **G** is not square.

Consequently, no unique inverse exists for the matrix **G**. To overcome this difficulty, we use minimum norm solution i.e.

$\mathbf{w} = \mathbf{G}^+\mathbf{d}$, where $\mathbf{G}^+$ is the psudoinverse of the matrix **G**

$$= (G^T G)^{-1} G^T d$$

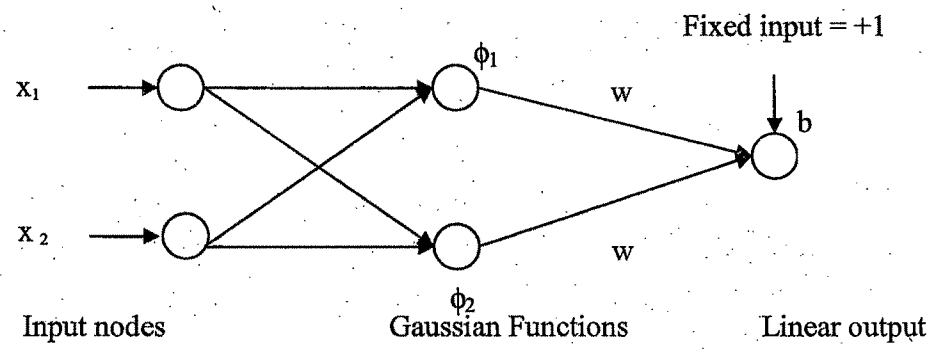
Note that $G^T G$ is a square matrix with a unique inverse of its own. So

$$G^+ = \begin{bmatrix} 1.8392 & -1.2509 & 0.6727 & -1.2509 \\ 0.6727 & -1.2509 & 1.8292 & -1.2509 \\ -0.9202 & 1.4202 & -0.9202 & 1.4202 \end{bmatrix}$$

and hence

$$w = \begin{bmatrix} -2.5018 \\ -2.5018 \\ +2.8404 \end{bmatrix}$$

These are the desired weight which provides the desired output by using equation (1.25) for $m_1 = 2$.



[Fig. 1.11. RBF network for solving the XOR problem]

The norm used in the equations 1.24 and 1.25 is usually a Euclidean norm. This norm can not useful for the large databases because the width (deviation) from the input pattern to the center, considered in this norm is 1. In all the cases, for instance pattern recognition problem, the width may not be equal to 1 hence we can thought of a weighted norm. Using weighted norm we can give proper justification to the distance calculated among the patterns by assigning suitable value of width. Chapters 5 and 6 of the thesis contain a typical application of weighted norm in the field of printed character recognition problem for Gujarati script. The weighted norm can be described as below:

Weighted Norm:

Ordinarily, Euclidean norm is being used. When however, the individual elements of the input vector **X** belong to different classes, it is more appropriate to consider a general weighted norm, the squared form of which is defined by

$$\|\mathbf{X}\|_C^2 = (C\mathbf{X})^T(C\mathbf{X})$$

$$= \mathbf{X}^T C^T C\mathbf{X}$$

Where C is an m_0 x m_0 norm weighting matrix and m_0 is the dimension of the input vector **X**. The equation 1.25 will take the form

$$F^*(\mathbf{X}) = \sum_{i=1}^{m_1} \mathbf{w}_i \, G\left(\|\mathbf{X}-\mathbf{t}_i\|_C\right)$$

where, $$\|\mathbf{X}-\mathbf{t}_i\|_C^2 = (\mathbf{X}-\mathbf{t}_i)^T C^T C(\mathbf{X}-\mathbf{t}_i) \qquad (1.27)$$

The distance defined in the equation 1.27 is known as Mahalanobis distance.

By considering input vector $\mathbf{X} = [X_1, X_2]^T$ and center vector $\mathbf{t_1} = [t_{11}, t_{12}]^T$, the equation 1.27 can be characterized as below:

$$= \sqrt{\left(\frac{X_1 - t_{11}}{\sigma}\right)^2 + \left(\frac{X_2 - t_{12}}{\sigma}\right)^2}$$ where σ is a covariance.

and hence $$\|\mathbf{X}-\mathbf{t}_i\|_C^2 = \frac{1}{\sigma^2}\left[(X_1 - t_{11})^2 + (X_2 - t_{12})^2\right]$$

$$= \frac{1}{\sigma^2}\left[\|\mathbf{X}-\mathbf{t}_1\|\right]$$

$$= \frac{1}{\sigma^2}\left[(\mathbf{X}-\mathbf{t}_1)^T(\mathbf{X}-\mathbf{t}_1)\right]$$

where the covariance (σ) can be characterized as below:

1. $\sigma(\mathbf{X}) = cov(\mathbf{X}) = \sqrt{\frac{1}{n}\sum_i (X_i - \bar{X})^2}$ for 1-variable

2. $\sigma(\mathbf{X},\mathbf{Y}) = cov(\mathbf{X},\mathbf{Y}) = \sqrt{\frac{1}{n}\sum_i (X_i - \bar{X})(Y_i - \bar{Y})}$ for 2-variable

3. $\sigma(x_1) = \frac{1}{n}(x_1 - \bar{x})^2$ for a single element x_1

$= \sigma_1$ (Singleton element)

1.5. Summary

This chapter discusses the basic introduction of different types of discrete wavelets transforms like Haar, Daubechies etc and various commonly used Artificial Neural Networks architectures like Multilayer Perceptron and Radial Basis Function networks. The advantages of wavelet transform over the fourier transform is discussed in section 2. Due to the localization property of wavelets in spatial and frequency domain they are sharper than the fourier transforms. The use of Weighted norm presented at the end of this chapter plays a vital role in chapters 4,5 and 6.

Organization of the thesis :

The work presented in this thesis involves a study of optical character recognition techniques for Gujarati script using various Artificial Neural Network architectures and wavelets. The thesis is divided in to six chapters. The following is a brief summary of the contents of each chapter.

(a) The current chapter, i.e, chapter 1 provides the details of Continuous and Discrete wavelet transforms, introduces the concepts of Artificial Neural Networks and describes the two most widely used Artificial Neural Network architectures viz. Multilayer Perceptron and Radial Basis Function networks.

(b) Pattern recognition is a typical application of Statistical learning theory. Chapter 2 discusses the use of Statistical learning theory in pattern recognition problems. Moreover, the chapter explains the complexities of Gujarati script and specifies how it differs from the Western scripts and from the other Indian language scripts.

(c) Chapter 3 describes a novel approach for approximation (interpolation) of a discrete finite length signal using the Multiresolution Analysis techniques of Discrete wavelet transforms [5].

(d) Feature extraction and recognition are two major components of any Optical Character Recognition system. Chapter 4 is concerned with an application of

wavelets for feature extraction and Multilayer Perceptron as classifier for digital images of Gujarati characters. The wavelet features are good in reducing the number of features while retaining the characteristics of the images. Multilayer perceptron network architecture is then used for the classification of Gujarati symbols, constituting the lower and middle zone glyphs [3, 4].

(e) Chapter 5 describes on the General Regression Neural Network (GRNN) architecture of Artificial neural networks as applied to character recognition. This approach of ANN is a typical application of Statistical learning theory. Applying GRNN as a classifier for the printed Gujarati symbols, we have got the highest recognition accuracy in all the three zones of the Gujarati script [30] among all over experiments.

(f) Two kernel based Artificial Neural Network architectures viz Radial Basis Function networks and Support Vector Machines are introduced in the chapter 6. We have explored these ANN architectures for the recognition purpose in the field of optical character recognition. The chapter also contains a uniform approach for two-class and multiclass problems in Support Vector Machine architecture.

(g) At the end of the thesis, the summary of all the chapters are provided followed by the references and appendix. An appendix provides a sample java code for wavelets and Multilayer perceptron networks.