FUNCTION APPROXIMATION USING WAVELET AND RADIAL BASIS FUNCTION NETWORKS

by

RABIHAH BINTI RAMLI

Dissertation submitted in partial fulfillment

of the requirements for the degree

of Master of Science in Mathematics

Jun 2004

ACKNOWLEDGEMENTS

In the name of Allah swt, Most Gracious and Most Merciful, I would like to take this opportunity to thank those who have helped me and encouraged me in various ways during the course of my candidature. First off all, I would like to acknowledge my deepest gratitude to my supervisor Prof. Madya Zarita Zainuddin who has constantly and consistently given me her expert guidance, undivided attention, valuable suggestions and critical advice in the whole process of my research.

Special thanks also go to all my friends for their help, support, and advice. Last but most importantly, I would like to thank my beloved parents Encik Ramli Che Mat and Puan Zainab Mohd Noor, my family for their unfailing love and affection, advice, support and motivation. Without them around me at those crucial moments, I would have felt an immense sense of loneliness.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

PENGHAMPIRAN FUNGSI MENGGUNAKAN RANGKAIAN WAVELET DAN FUNGSI ASAS RADIAL

ABSTRAK

Rangkaian Wavelet telah diperkenalkan sebagai proses suap depan bagi rangkaian neural yang disokong oleh teori wavelet. Rangkaian neural ini dapat digunakan secara langsung dalam penghampiran fungsi. Dalam disertasi ini, Rangkaian Wavelet dibuktikan sebagai salah satu sub-bahagian dalam kumpulan keturunan di mana rangkaian neural ini mempunyai sifat yang sama dengan kumpulan yang di namakan Fungsi Asas Radial Berpemberat. Hal ini juga berlaku bagi rangkaian neural yang mempunyai paradigma yang berlainan. Disertasi ini juga merangkumi pengkajian dalam Fungsi Asas Radial berperingkat 2.Fungsi ini juga dikenali sebagai Fungsi Asas Radial Piawai kerana mempunyai persamaan dimana fungsi ini akan bertindak sebagai Fungsi Asas Radial Piawai apabila fungsi exponent mempunyai sifat yang sama dengan fungsi pengaktifan Gaussian apabila peringkat bagi eksponen n =2. Selain daripada itu, kita dapat melihat perbandingan diantara Wavelet dan Fungsi Asas Neural Berpemberat peringkat 2 bagi melakukan penghampiran fungsi.Hal ini dikaji bagi membolehkan kita membuat demostrasi bagi menyatakan bahawa pendekatan bagi penghampiran fungsi yang baik bergantung pada pemilihan yang dibuat dalam memilih fungsi pengaktifan bagi Rangkaian Fungsi Asas Radial dan ibu fungsi bagi Rangkaian Wavelet. Mexican Hat, Gaussian Wavelet dan Morlet digunalan sebagai fungsi ibu dalam Rangkaian Wavelet manakala Gaussian sebagai fungsi pengaktifan dalam Rangkaian Fungsi Asas

Radial. Fungsi Kos Kuadratik digunakan bagi meminimakan ralat yang dinilai. Perhitungan yang telah dibuat daripada semua rangkaian neural dinilai dengan mengambil *Normalised Squre Root Mean Squared Error* (*NSRMSE*). Empat fungsi digunakan bagi membuat simulasi dan dua daripada fungsi tersebut melibatkan satu pembolehubah. Selebihnya adalah fungsi dalam dua pembolehubah bagi membolehkan penghampiran fungsi dinilaikan melalui rangkaian neural yang telah dinyatakan di atas. Simulasi juga dilakukan dalam menaksir data sebenar iaitu dalam meramalkan harga rumah di pendalaman Boston. Simulasi dilakukan dengan menggunakan MATLAB V6.5.

ABSTRACT

The Wavelet Neural Network has been introduced as a special feedforward neural network supported by the wavelet theory. Such network can be directly used in function approximation problems. In this dissertation, wavelet networks are proven to be as well as many other neural paradigms, a specific case of generic paradigm named Weighted Radial Basis Functions Network. In this dissertation we will also investigate the WRBF-2.

WRBF-2 is standard RBF since the exponential function behaves as a Gaussian, due to the exponent n = 2.In addition a comparison between Wavelet and WRBF-2 networks for function approximation is attempted, in order to demonstrate that the performance depends only on how good the chosen mother function for the WNN and activation function for the RBFN "fits" the function itself. Mexican Hat, Gaussian Wavelet and Morlet are used as the mother wavelet function in WNN and Gaussian activation function in RBFN. Quadratic cost function is used for error minimization. The performances of the networks are estimated by Normalised Square Root Mean Square Error (NSRMSE). Four functions have been used for the simulations. Two of the functions involved one variable function and the rest are two variable functions to be approximated by the networks. We also used the real word data for the simulations. Simulations are done by using Matlab V6.5.

CHAPTER 1

INTRODUCTION OF THE WAVELET NEURAL NETWORKS

1.1 INTRODUCTION

Wavelet Neural Networks (WNNs) are an implementation of Wavelet Decomposition, a technique which has recently emerged as a powerful tool for many applications in the field of signal processing, such as data compression and function approximation. The wavelet network is an approach for system identification in which nonlinear functions are approximated as the superposition of dilated and translated versions of a single function (Zhang,1992).Wavelet network uses a wavelet like activation function. Families of wavelet functions especially wavelet frames are universal approximators for identification of nonlinear system. The parameters of wavelet networks are dilation (t), translation (e), bias (Θ) and weight (w). The parameters are optimized during the learning phase.

The basic idea of Wavelet decomposition is to expand a generic signal $f(x) \in L^2(\Re^N)$ into a series of functions obtained by dilating and translating a single function $\Phi(x)$, the so-called mother wavelet.

The term mother wavelet gets its name from two important properties of the wavelet analysis. The term wavelet means a small wave. The term mother implies that the functions with different regions of support are used in the transformation process. They are derived from one main function, the mother wavelet. In other words, the mother wavelet is a prototype for generating the other window function (Polikar, 2001). Mother wavelet function gives an efficient and useful description of the signal of interest. This function has universal property.

In the following we shall consider only radial wavelets in $L^2(\Re^N)$, for which $\Phi(x) = g(\|x\|)$ where $g: \Re \rightarrow \Re$. Radial functions are characterized by a radial Fourier transform; a function is admissible as a wavelet if $C_\Phi = (2\pi)^N \int_0^\infty \frac{|\hat{\Phi}(hw)|^2}{h} dh < \infty$ and C_Φ is independent of ω.For the Discrete Wavelet Transform, the parameters which determine the dilation and translation of the mother wavelet are discretised, namely a countable set is extracted, such that the corresponding wavelet family

$$\{\Phi_k = \det(D_k^{1/2})\Phi[D_k(x-t_k)] : t_k \in \Re^N, D_k = diag(d_k), d_k \in \Re_+^N, k \in M\} \quad (1.1)$$

is a basis for the functions in $L^2(\Re^N)$. To this aim, additional conditions are required both on Φ an on the parameters discretisation. The obtained basis is not necessarily orthonormal and can be somehow redundant. In this latter case family (1.1) is more correctly referred to as frame. Frames of wavelet have been used extensively to approximate functions of one or two variables (Meyer, 1992), but as the number of variables increases, the required number of basis functions grows exponentially. In

practice a signal g is approximated by the weighted sum of finite number of functions in (1.1) plus a bias which help the approximation of functions with nonzero mean value:

$$g(x) = \sum_{k=1}^{K} a_k \Phi[D_k(x - t_k)] + \Theta \qquad (1.2)$$

which is analogous to the output of a 2-layer network, provided that the activation function of the hidden neurons are wavelets (Zhang,1992). Such network has been named Wavelet Network (WN). WN with radial wavelets presents the main advantage of an efficient initialization procedure derived from the wavelet decomposition (Zhang, 1997).Furthermore a fast procedure based on the Orthogonal Least Squares (OLS) algorithm, a method already applied to RBF networks (Chen, *et al.*, 1991), is provided for choosing among all the basis functions those which give the greatest contribution to the approximation.

Depending on the form of the function to be approximated, the expansion of a signal into a wavelet series can be more efficient than other solutions, in the sense that fewer basis functions can be needed for achieving a fixed approximation error. This is due to the time-frequency local properties of most wavelets, which make them particularly suitable to represent short-time high-frequency signal features. Fewer basis functions and more efficient initialization lead to smaller networks and fast training. On the other hand some signal features are better represented by the linear combination of different function, thus WN are not suitable to fit any curve.

The success of radial basis function (RBF) neural networks for function approximation was a good indicator of this yet another field of application on wavelets. A wavelet

frame replaces the radial basis functions in a RBF network, the center and covariance matrix (spread) are replaced by the shifts and scales of the wavelets (Zhang, *et al.*, 1995).

The architecture of wavelet network consists of three different layers: an input layer which is made of source nodes, a hidden layer in which each neuron computes its output using a wavelet basis function and an output layer which builds a linear weighted sum of the hidden layer. The input layer to the hidden layer transformation is nonlinear while from the hidden layer to the output layer, the transformation is linear. The structure of WNN for n-dimensional input and one output is shown as in Figure 1 (Sheng & Shu, 1999).

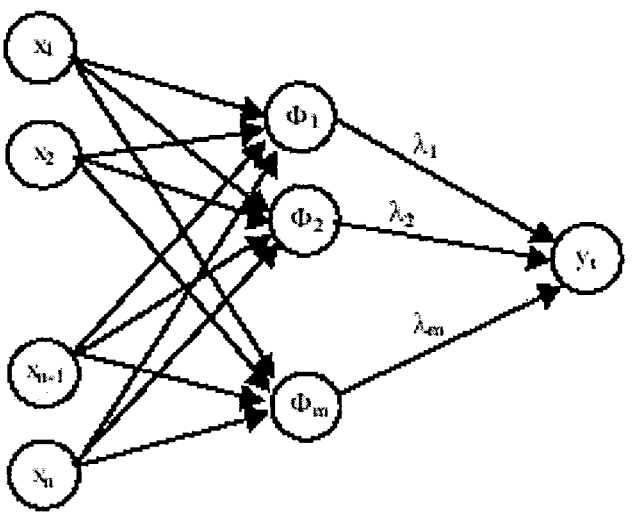


Figure 1: An Wavelet Neural Network structure with one output

1.2 HISTORICAL BACKGROUND

The origin of wavelet networks can be traced back to the work of Daugmann (1988) in which Gabor wavelets were used for image classification. Wavelet networks have become popular after the work by Pati (1991, 1992), Zhang (1992), and Szu, *et al.* (1992). Wavelet networks were introduced as a special feedforward neural networks. Zhang did apply wavelet networks for the problem of controlling a robot arm. As a mother wavelet, they used the function $\Phi(x) = (x^T x - \dim(x)).e^{-1/2 x^T x}$.

Szu took a different function, $\Phi(x) = \cos(1.75\ t)\exp(-t^{2/2})$, as mother wavelet for classification of phonemes and speaker recognition. Simple combinations of sigmoids were chosen by Pati (1991). Fernando Marar, *et al.* (1996) have shown an effective procedure for generating polynomial forms of wavelet functions from successive powers of sigmoid functions.

A motivation for using wavelet networks is that there are universal function estimators that may represent a function to some precision very compactly. This follows from the work by Hornik (1989) and Kreinovich, *et al.* (1994). Hornik has shown that an arbitrary continuous function on compact set can be approximated by a 3-layer neural network within a precision ε.

Kreinovich has proven that wavelet neural networks are asymptotically optimal approximates for functions of one variable. Wavelet neural networks are optimal in the sense that they require the smallest number of bits to store, for reconstructing a function within a precision ε.

From the practical point of view, the determination of the number of wavelets and their initialization represents a major problem with wavelet networks. A good initialization of wavelet neural networks is extremely important to obtain a fast convergence of the algorithm. A number of methods have been implemented (Thuillard, 2000). Zhang (1992) initializes the coefficient with the orthogonal least-squares procedure. As an alternative, the dyadic wavelet decomposition may be used to initialize the network. Echauz (1998) applies a clustering method to position the wavelets. The distribution of points about a cluster permits to approximate the necessary dilation of the wavelet. Echauz (1996) also proposes an elegant method using trigonometric wavelets. He uses functions of the form: Cos trap(x) = $\cos(3\pi/2x).\min\{\max\{3/2(1-|x|),0\},1\}$ as a transfer function. Trigonometric wavelets can be approximated by polynomials. Fitting of the polynomial is a linear problem that is solved more easily than fitting trigonometric wavelets. The fitting parameter of the polynomials can be used to approximate the initialization parameters of the corresponding wavelets. In Boubez and Peskin (1993), the network is initialized by positioning and approximating first low resolution wavelets.

At present, supervised learning is probably the most frequently used technique in the field of neural networks. A teacher provides training examples of an arbitrary mapping which the network start to learn. Learning in this context means an incremental adaptation of connecting weights that transport information between simple processing units. In fact, this sort of learning can be expressed as a minimization problem over a many dimensional parameter space, namely the vector space spanned by the weights. A typical technique to perform this kind of optimization is gradient descent. The learning rule of the most popular supervised learning procedure is the backpropagation

algorithm. Backpropagation algorithms, conjugate gradient method (Szu, et al., 1992) stochastic gradient algorithm (Zhang & Beneviste, 1992) or genetic algorithms (Prochazka & Sys, 1994) are used for training the network.

Wavelets networks were first mentioned by (Zhang & Benveniste, 1992) in the context of non-parametric regression of functions in $L^2(\Re^2)$. In wavelet networks, the radial basis functions of RBF-networks are replaced by wavelets. During the training phase, the network weights as well as the degrees of freedom (position, scale, orientation) of the wavelet functions are optimized. Zhang and Benveniste realized that wavelet networks inherit the properties of wavelet decomposition and mention especially their universal approximation property, the availability of convergence rates and the explicit link between the network coefficients and the wavelet transform.

However, since their introduction in 1992, wavelet networks (WN) have received little attention. (Szu, *et al.*, 1992, Szu, *et al.*, 1996) have used WNs for signal representation and classification. They have explained how a WN template, a *superwavelet*, can be generated and presented original ideas for how they can be used for pattern matching. In addition, they mention the large data compression achieved by such a WN representation. (Zhang, 1997) showed that WNs are able to handle nonlinear regression of moderately large input dimension with sparse training data. (Holmes & Mallick, 2000), analyzed WNs in the context of a Bayesian framework. (Reyneri, 1999) lately analyzed the relations between artificial neural networks (ANNs), fuzzy systems and WNs have been discussed. In their pioneering 1996 paper, Bakshi and Stephanopoulus (1992) showed that neural networks using wavelets as basis functions are particularly efficient in learning from sparse data and is dense, and a lower resolution when data is

sparse also fit naturally into multiresolution wavelet analysis scheme. More recently, Bernard, Mallat and Slotine proposed wavelet interpolation networks capable of real time learning of unknown functions.

1.3 APPLICATIONS OF WAVELET NEURAL NETWORKS

A number of interesting applications have taken advantage of the multiresolution properties of wavelet networks. Many manufacturing process monitoring systems have the function of detecting abnormal vibrations (Pittner, *et al.*, 1998). For vibration detection and classification, wavelet-based methods represent good alternatives to Fourier analysis. Engine knock detection systems have been developed by PSA-Peugeot-Citroen (Thomas, *et al.*, 1996) on the basis of wavelet networks. Another related application is detection of vibrations of detective circuit breakers in electric power (Lee, 1999).

Wavelet networks have been implemented with success to identify and classify rapidly varying signals for instance to identify high risks patients in cardiology (Dickhaus & Heinvich, 1996) or for echo cancellation (Lixia, *et al.*, 1996).

Studies on radar applications have dealt with aircraft velocity estimation (Sanchez-Redondo & Zufina, 1998) or rain forecasting (Yeung & Kwok, 1996).

Wavelet networks have been tested on a number of classical control problems, from the detection of small variations in a plant to the control of robotics arms (Katic & Vukobratovic, 1997).

An interesting alternative to wavelet networks consists of using dictionary of dyadic wavelets and to optimize only the weights w_i. This approach is generally referred to as wave-net or wavenets. It was first proposed by Bakshi, *et al.*, (1994).

1.4 GUIDES TO DISSERTATION

The main purpose of this dissertation is to use Wavelet Network and Radial Basis Function Networks in function approximation and demonstrate that the performance depends on how good the chosen mother/activation/ transfer function "fits" the function itself. The investigation includes a fair comparison between WNs and Radial Basis Functions (RBFs) which are a specific case of Weighted Radial Basis Functions Networks (WRBF).So far WNs and RBFs have been seen as two rather different approaches to the task of function approximation, and most paper published on the subject are willing to prove that one method is far better than the other due to some hot point specific on the method. It has been proven (Reyneri, 1996) that many neural and fuzzy paradigms are nothing but specific cases of a generic paradigm called Weighted Radial Basis Functions (WRBFs), which therefore behaves as a neuro-fuzzy unification paradigm. In this dissertation we will show that also WNs are a specific case of WRBFs, therefore it can easily be shown that WNs and RBFs can behave exactly alike when they are properly designed. In order to evaluate the performance of chosen mother/activation function, we used four different functions. Two of the functions involve are one variable functions and the rest are two variables functions to be approximated by the proposed networks. We also used the real world data for the simulations.

Literature survey on wavelet networks are presented in this chapter. In this chapter we introduce the wavelet networks that we use which are from continuous wavelet frames, their architectures, the history and the applications of wavelet network. Basic concepts and theory of wavelet networks in function approximation will be explained in Chapter

and the wavelet neural network (WNN) along with an introduction to function approximation. The relationship between these neural networks and function approximation theory is also shown. In Chapter 3, we discussed wavelet and neural paradigms so that that we can implement wavelet network and radial basis function to look exactly alike and then we look at the similarities of the structure of their network by using this generic paradigm so that we can find the best of the chosen mother/activation function and how good it "fits" the function. We implement this concept in Chapter 5 using MATLAB version 6. In Chapter 4, we discussed various mother/activation functions that we used in this thesis to approximate the four functions and the benchmark problem that we used as an experiment. We also look at the learning and training algorithms for wavelet network and radial basis function network. We use the same initialization and learning rules for both types of networks because the aim of this dissertation is to compare WNs and RBFs fairly. The implementation of wavelet network algorithms using radial basis function network algorithms are analyzed in Chapter 5.We can do this because they have the same design structure and the only difference is the activation function. The simulations of functions approximation are given and the results of simulations are discussed in Chapter 6.We also implement our experiments on real word data from StatLib library which is obtained from Carnegie Mellon University and is discussed in the same chapter.

Finally, a discussion and suggestions for future work related to this research will be presented in Chapter 7.

CHAPTER 2

BASIC CONCEPTS AND THEORY OF WAVELET NETWORKS IN FUNCTION APPROXIMATION

2.1 FUNCTION APPROXIMATION

Consider a function $y = F(x)$, which maps an input vector x onto an output vector y. To be specific, let the set of input-output data available for approximation be described by

$$\begin{aligned} &\text{Input signal:} && x_i \in \Re^m, && i = 1,2,\ldots,N \\ &\text{Desired response:} && d_i \in \Re^1, && i = 1,2,\ldots,N \end{aligned} \qquad (2.1)$$

Note that the output is assumed to be one dimensional. Let the approximating function be denoted by $F(x)$.The goodness of fit of d_i on a set of samples is given by an error function. A commonly used measure is the standard error (distance) between the desired (target) response d_i and the actual response y_i for training example $i = 1,2,\ldots,N$. Specially, we define:

$$E(F) = \frac{1}{2}\sum_{i=1}^{N}(d_i - y_i)^2$$

$$= \frac{1}{2}\sum_{i=1}^{N}\left[(d_i - F(x_i))\right]^2 \qquad (2.2)$$

How to find the good approximation d_i. Usually d_i is chosen to be a parametric function, where the parameters determine the exact shape of the function. Then these parameters can be optimized, to minimize the error on the samples Mean Squared Error (MSE). In approximation theory this is called parameter estimation; in neural network terminology this is called learning. A large number of methods exist to find the optimal parameters for some given parametric function.

2.1.1 SMOOTHNESS

A large class of functions found in practice is the so called smooth functions. A smooth function has the following characteristics:

- The function is continuous.
- Input vectors close to each other in the input space are mapped onto output vectors close to each other in output space. Closeness can be measured for example by the Euclidean distance.

The mapping d_i can be viewed as an (n + m) dimensional landscape, where n and m are the dimensions of input vector x and output vector y respectively. For example, a 2-dimensional input and 1-dimensional output mapping can be seen as a 3-dimensional landscape (surface). Using this analogy, a smooth mapping does not have sharp peaks and valley, and the slopes do not change suddenly. Of course, there is no sharp

distinction between smooth and non-smooth functions: smoothness is a matter of degree.

2.1.2 APPROXIMATING SMOOTH FUNCTIONS

Approximating a smooth function from a given set of samples means creating a mapping with the following properties:

- The error on the learning samples should be small as possible, since these samples are used to optimize the unknown parameters d_i.
- The approximation should be as smooth as possible, since d_i is assumed to be smooth.

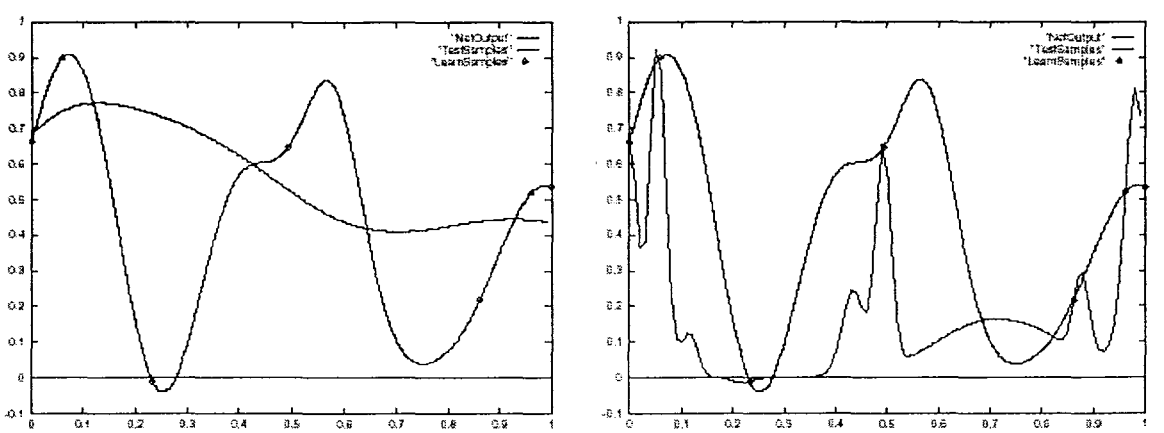
These two properties are contradictory. A very smooth approximation cannot approximate the learning samples properly: it has a high bias. On the other hand, approximating the learning samples perfectly compromises smoothness, and is not needed because the samples are noisy anyway: the approximation has high variance. Having a small error on the learning set while the smoothness of the graph is bad (and therefore bad generalization) is called overfitting. We can observe this in Figure 2.1 (Morozov, 1993).

There is a tradeoff between having high bias and having high variance, which is commonly referred to as the bias versus variance dilemma. What the balance should be between these two aspects for one particular problem is not known beforehand (Bosman, 1996).

(Demuth & Beale, 2000) states that it is difficult to know beforehand how large a network should be for a specific application. There are two other methods for improving generalization that are implemented in Neural Network Toolbox: regularization and early stopping. Note that if the number of parameters in the network is much smaller than the total number of points in the training set, then there is little or no chance of overfitting. Only the error on the test set can give feedback about the effect of a particular choice (after optimization though).

2.2 ILL - POSED PROBLEMS AND WELL - POSED PROBLEMS

To develop a deep understanding of the overfitting problem and how to cure it, we first go back to the viewpoint that the design of a neural network trained to retrieved an output pattern when presented with an input pattern is equivalent to learning a hypersurface that defines the output in terms of the input. According to Keller (1976), two related problems are said to be inverse of each other if the



(a) (b)

Figure 2.1: Underfitting and overfitting on a smooth function: (a) underfitting (b) overfitting.

formulation of each of them requires partial or full knowledge of the other. However from mathematical perspective, there is another more important difference between a direct problem and inverse problem. Specifically, a problem of interest may be well-posed or ill-posed.

Assume that we have domain X and a range Y taken to be metric spaces, and that are related by a fixed but unknown mapping F. The problem of reconstructing the mapping F is said to be well-posed if three conditions are satisfied (Tikhonov & Arsenin,1977 ;Morozov,1993):

- Existence. For every input vector $x \in \Re$, there does exist an output $y = F(x)$, where $y \in Y$.
- Uniqueness. For any pair of input vectors $x, t \in \Re$ we have $F(x) = F(t)$ if, and only if, $x = t$.
- Continuity. The mapping is continuous, that is for any $\varepsilon > 0$ there exists $\delta = \delta(\varepsilon)$ such that the condition $\rho_x(x,t) < \delta$ implies that $\rho_y(F(x),F(t)) < \varepsilon$, where $\rho(.,.)$ is the symbol for distance between the two arguments in their respective spaces. This property of continuity is also referred to as stability.

If any of these conditions is not satisfied, the problem is said to be ill-posed. Basically, an ill-posed problem means that large data sets may contain a surprisingly small amount of information about the desired solution.

2.3 GENERALIZATION

An approximation is not useful if it can remember the samples, but performs poorly on the rest of the input space. We want d_i to generalize over the samples, using the samples to create the plausible approximation of d_i for the complete input domain. In the general case, this is impossible. To make approximation possible on the complete input domain, the function d_i must be redundant in the sense that a limited set of samples contains information about the rest of the mapping (Bosman, 1996). The generalization performance of d_i can not be measured by the error on the samples used to optimize d_i's parameters. Therefore, the set of available samples is commonly split into two sets: a training set, used for optimizing the parameters, and a test set, used to get an indication of the generalization. Note that the error on the test set is just an estimate of the generalization performance, since it still does not measure the error on the complete input domain.

2.4 THE REGULARIZATION APPROACH TO THE APPROXIMATION PROBLEM

The approach regularizes the ill-posed problem of function approximation from sparse data by assuming an appropriate prior on the class of approximating functions. Regularization techniques (Tikhonov, 1963; Wahba, 1990) typically impose smoothness constraints on the approximating set of functions.

Then according to Tikhonov regularization theory (Haykin, 1994) the function F can be obtained by minimizing an error functional given by

$$E(F) = E_s(F) + \lambda E_r(F) \tag{2.3}$$

where λ is the regularization parameter, E_s is the standard error between the desired output and the actual response y

$$E_S = 1/2 \sum_{i}^{N} (d_i - y_i)^2 \tag{2.4}$$

And E_r is the regularizing term that depends on the properties of F. If P is a linear pseudo differential operator embedding a smoothness constraint,

$$E_r = \frac{1}{2} |PF|^2 \tag{2.5}$$

The resulting solution is smooth and therefore, continuous. In order to find F that minimizes the total error, we differentiate E with respect to F using the Frechet differential and set it equal to zero.

$$dE(F,h) = 2\left[h, P^* PF - \frac{1}{\lambda} \sum_{i}^{N} (d_i - F)\delta_{x_i} \right]_H \tag{2.6}$$

where $h(x)$ is a fixed function of the vector x, $\delta_{x_i} = \delta(x - x_i)$, P* is the adjoint of P, and the symbol $(\ldots)_H$ denotes the inner product in H space. Since $\lambda \in (0,\infty)$, the Frechet differential is zero for any h(x) in H if an only if

$$P^*PF = \frac{1}{\lambda}\sum_{i}^{N}(d_i - F)\delta(x - x_i) \tag{2.7}$$

Equation (2.7) is referred as the Euler- Lagrange equation for the cost functional E(*F*) and its solution is given by

$$F(x) = \int_R G(x,\theta)\frac{1}{\lambda}\sum_{i}^{N}(d_i - F(\theta_i))\delta(\theta - x_i)d\theta \tag{2.8}$$

where θ is the variable of integration and $G(x,x_i)$ is the Green's function for the self-adjoint operator P*P, for example

$$P^*PG(x, x_i) = \delta(x - x_i) \tag{2.9}$$

Integrating, we get

$$F(x) = \frac{1}{\lambda}\sum_{i}^{N}(d_i - F(x_i))\,G(x,x_i) \tag{2.10}$$

which can be written in matrix-vector form as

$$F = Gw \tag{2.11}$$

with

$$w = \frac{1}{\lambda}(d - F) \tag{2.12}$$

and

$$G = \begin{pmatrix} G(x_1,x_1) & G(x_1,x_2) & \dots & G(x_1,x_p) \\ G(x_2,x_1) & G(x_2,x_2) & \dots & G(x_2,x_p) \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ G(x_p,x_1) & G(x_p,x_1) & \dots & G(x_p,x_p) \end{pmatrix} \tag{2.13}$$

Since the operator P*P is self-adjoint, the associated Green's function and consequently the matrix G will be symmetric. Further (Light, 1992) has proved that the matrix G is positive definite provided that the data points x_1, x_2,..., x_p are distinct. In practice, λ may be chosen sufficiently large so that the matrix $G + \lambda I$ is positive definite. This implies that the system of equations (2.12) has a unique solution given by

$$w = (G + \lambda I)^{-1} d \tag{2.14}$$

and the function F is given by

$$F(x) = \sum_{i}^{N} w_i G(x, x_i) \tag{2.15}$$

The number of Green's functions used in this expansion is equal to the number of data points.

From equation (2.3), the first term is enforcing closeness to the data, and the second smoothness, while the regularization parameter controls the tradeoff between these two terms, and can be chosen according to cross-validation techniques (Cravend & Wahba,1979). We first need to give a more precise definition of what we mean by smoothness and define a class of suitable smoothness functional. We refer to smoothness as a measure of the "oscillatory" behavior of a function. Therefore, within a class of differentiable functions, one function will be said to be smoother than the other one if it oscillates less.

2.5 RADIAL BASIS FUNCTION NEURAL NETWORKS

The theory described above can be implemented as a radial basis function (RBF) neural network. Radial basis function (RBF) neural networks are a class of networks that are widely used for solving multivariate function approximation problems (Haykin, 1994). An RBF neural network consists of an input and output layer of nodes and a single hidden layer can be observed from Figure 2.2 (Ramuhalli, 2002). Each node in the hidden layer implements a basis function $G(x,x_i)$ as the basis functions. The input-output relation for the RBFNN is given by

$$y_l = \sum_{i}^{N} w_i G(x, x_i) \quad l = 1,2,\ldots,M \tag{2.16}$$

where N is the number of basis functions used, $y = (y_1, y_2,\ldots, y_M)^T$ is the output of the RBFNN, x is the test input, x_j is the center of the basis function and w_{lj} are the expansion coefficients or weights associated with each basis function. Each training data samples is selected as the center of a basis function. Basis functions $G(x,x_i)$ that

are radially symmetric are called radial basis functions. Commonly used radial basis functions include the Gaussian and inverse multiquadrics. In this dissertation, we only concentrate on Gaussian.

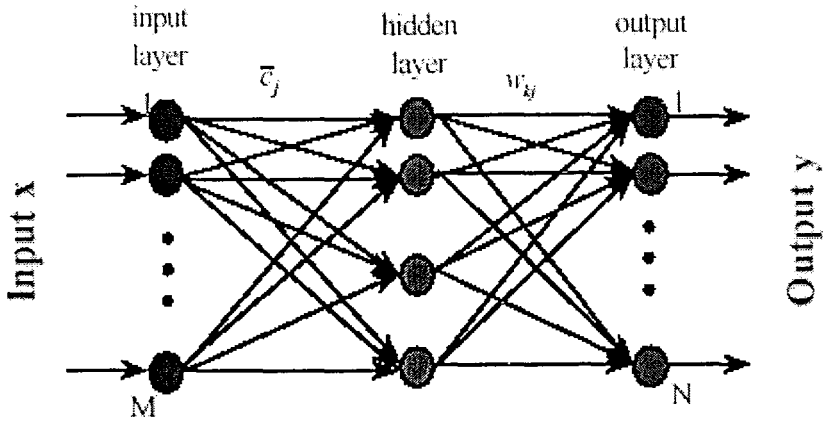


Figure 2.2: The radial basis function neural network

The network described above is called an exact RBFNN, since each training data point is used as a basis center. The storage costs of an exact RBFNN can be enormous, especially when the training database is large. An alternative to an exact RBFNN is a generalized RBFNN where the number of basis functions is less than the number of training data points. The problem then changes from strict interpolation (in exact RBFNN) to an approximation, where certain error constraints are to be satisfied. We use the concept of generalized radial basis function neural network because from approximation Eq.(1.2) with a radial function, Φ can be seen as a generalization of the "radial basis function" (Poggio & Girosi, 1990a). We will show the generalization in wavelet network in Chapter 4 where it involves mother wavelet function.

CHAPTER 3

WAVELET AND NEURAL PARADIGMS

This section describes and compares the WNs and NNs used during the work and briefly describes the unification paradigm which proved to be very useful for training and initialization. So far WNs and RBFs have been considered as two rather different approaches to the task of function approximation. In practice, it has been proven (Reyneri,1996) that many neural and fuzzy paradigms are nothing but specific cases of generic paradigm called Weighted Radial Basis Functions (WRBFs), which therefore behaves as a neuro-fuzzy unification paradigm. In this chapter, we will also show that WNs are a specific case of WRBFs, therefore it can easily be shown that WNs and RBFs can behave when properly designed exactly alike.

3.1 DISTANCED BASED (OR, RADIAL) NEURAL NETWORKS

Distanced–Based (or Radial) Neural Networks: include most neural paradigms which are not correlation-based such as radial basis functions (RBFs), Kohonen networks, self organizing maps (SOM's) and restricted Coulomb energy networks (RCEs).In this dissertation, we only discuss radial basis functions.

Such networks are based on radial basis neurons (or, R-neurons), which have a model based on the nth order distance between the input vector $\vec{X}$ and a center vector $\vec{C}$ of identical dimensions

$$y_j = G\left(\sqrt[n]{\sum_i |x_i - c_{ji}|^n}\right) = G\left(\| \vec{X} - \vec{C_j} \|_n\right) \tag{3.1}$$

where $\| \cdot \|$ is the n-norm of the argument typically $n \in \{1,2\}$,that is, Hamming or Euclidean distance when n = 1 and n = 2 respectively, while G(z) is usually a nonlinear, monotonic decreasing for $z \geq 0$, and limited activation function [note that G(z) is always used only for $z \geq 0$]. In many cases, G(z) is a generalized exponential function see Figure 3.1(Reyneri, 1999).

$$G(z) = G_{min} + (G_{max} - G_{min}) \cdot e^{-|z/\sigma|^n} \tag{3.2}$$

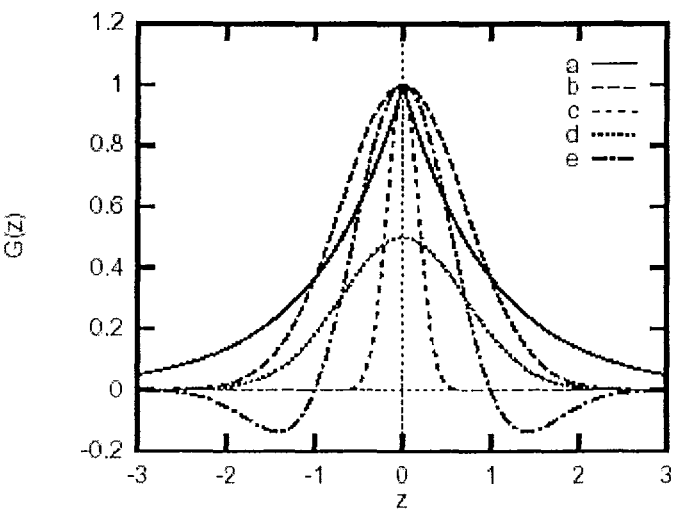


Figure 3.1: Generalized exponential function with G_{min}= 0: a) n=1,G_{max}= +1, σ =1;

b) n=2, G_{max}=+1, σ =1; c) n=2, G_{max}=+1, σ = 0.2; d) n=2, G_{max}=+1, σ =1.

which coincides with the exponential and the Gaussian functions, for n = 1 and n = 2 respectively, and $G_{min} = 0$, $G_{max} = 1$.The parameter n is the order of the function and determines its steepness around $z = \sigma$, while σ is the width factor. In most cases an R-neuron has the input – output characteristic as shown in Figure 3.2 (plot b)(Reyneri, 1996),

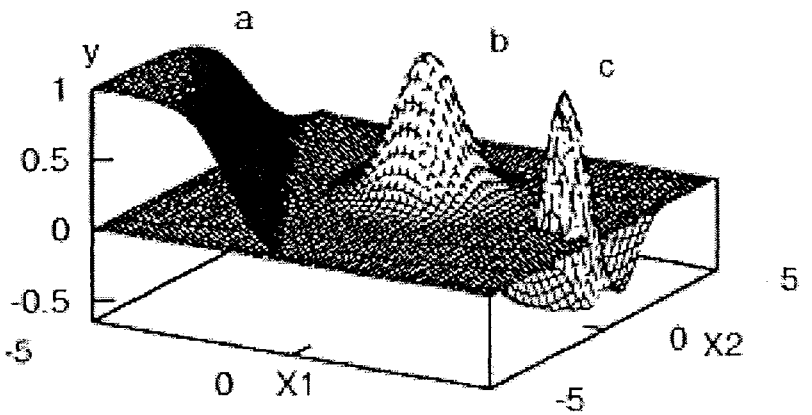


Figure 3.2: Input-output characteristic: (a) P-neuron; (b) R-neuron;(c) Mexican Hat wavelon.

with a closed decision boundary $\| \vec{X} - \vec{C} \|_n$ = constant (spherical, for n = 2, from which the name of radial basis functions is given to a specific type of R-neurons). Some examples of the such decision boundaries for two inputs are given in Figure 3.3 (plot b, c, d e, f) (Reyneri, 1999), but we only consider (plot c)(Reyneri, 1999). R-neurons often suffer from dimensional problem, which arises when trying to sum up together, in Eq.(3.1), inputs with different physical dimensions.

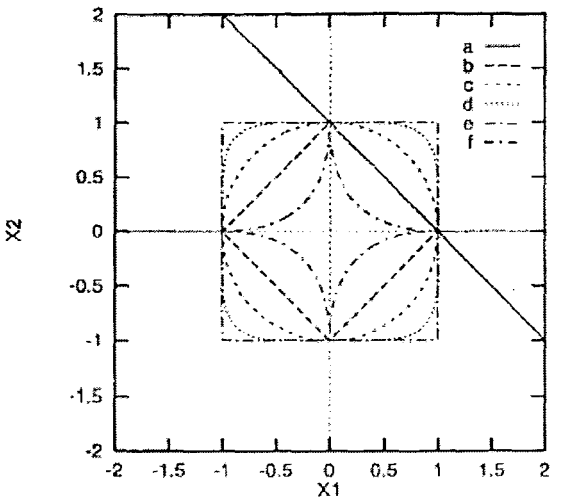


Figure 3.3: Decision boundaries: (a) linear boundary of P-neurons; (b) R-neurons WRBF-1; (c) R-neurons WRBF-2, wavelons.

3.2 RADIAL WAVELET NETWORKS

Radial Wavelet Networks are based on wavelet decomposition and use radial mother wavelet $\Phi(\ \|X\|\ ) \in L^2(\Re^N)$ appropriately dilated and translated. Such networks are based on radial wavelons which have a model based on Euclidean distance between the input vector $\vec{X}$ and a translation vector $\vec{E_j}$ where each distance component is weighted by a component of a dilation vector $\vec{T_j}$

$$y_j = \Phi\left(\sqrt{\sum_i \left(\frac{x_i - e_{ji}}{t_{ji}}\right)^2}\right) = \Phi\left(\left\|\frac{\vec{X} - \vec{E}}{\vec{T}}\right\|\right) \tag{3.3}$$

where the "vector division" in the right term is to be calculated "element-by-element". A commonly used function is the Mexican Hat:

$$\Phi(z) = (1-2z^2).e^{-z^2} \qquad (3.4)$$

A wavelon has the input-output characteristic shown in Figure 3.2 (plot c)(Reyneri,1996),with the hyper-elliptical decision boundary $D(\lambda)$given by $\sum_i \left(\left(x_i - e_{ji} / t_{ji} \right) \right)^2 = (\Phi^{-1}(\lambda)^2)$ as shown in Figure 3.3 (plot c)(Reyneri,1999). In this dissertation, we have chosen to simulate our experiment by using three different mother functions which are Mexican Hat, Morlet and Gaussian Wavelet .

In this dissertation we do not include the correlation based networks which are based on perceptron-like neurons (or, P-neurons) .We only use the unification paradigm. We only concentrate on distanced based neural networks (R-neurons) and radial wavelet networks (R-wavelons).

3.3 NEURO-WAVELET UNIFICATION

In Section 3.1 and Section 3.2, we have presented the mathematical models of the various neural paradigms. Formulae (3.1) and (3.3) have intentionally been written in such a form that their common "structures" become evident. Namely the output of each Wavelon (or Neuron) is a non-linear function of either the dot product or the distance between $\vec{X}$ and one or two parameters vectors $\left(\vec{W}, \vec{C}, \vec{E}, \vec{T}\right)$ associated with each Wavelon (respectively Neuron). The WRBF unification paradigm described in Section 3.3.1 has been derived from this consideration (Reyneri, 1996).

3.3.1 WEIGHTED RADIAL BASIS FUNCTIONS

Each WRBF layer is a collection of M (possibly 1) neurons and is associated with the following parameters:

- an order $n \in \Re$,defining the neuron's metric (mostly $n \in \{0,1,2\}$)
- a weight matrix $\underline{\underline{W}}$, deriving from the dilation vectors of Wavelons;
- a center matrix $\underline{\underline{C}}$, deriving from either the center vectors of R-neurons or the translation vector of Wavelons,
- an optional bias vector Θ, deriving from P-neurons;
- an activation function G(z) also called transfer function or basis function. These three terminologies are used interchangeably throughout this study. A WRBF neuron is associated with a set of parameters: an order $n \in \Re$, defining the neuron's metric; a weight vector w; a center c, a bias Θ and an activation function G(z).The mathematical model of a WRBF neuron of order n (or, WRBF-n) is then:

$$\vec{Y} = H_n^{G(Z)}\left(\vec{X};\underline{\underline{C}};\underline{\underline{W}};\Theta\right) \tag{3.5}$$

where

$$y_j = G\left(w^T \Delta_n\left(x_i - c_{ji}\right) + \Theta\right) \tag{3.6}$$

where $w = [w_1, w_2, \ldots, w_n]^T$ and $\Delta_n(x_i - c_{ji})$ is a factor of $\Re^N$ whose entries are:

$$\Delta_{n,i}(x_i - c_{ji}) \underset{\equiv}{\Delta} (x_i - c_{ji}) \qquad \text{for } n = 0; \tag{3.7}$$

$$\Delta_{n,i}(x_i - c_{ji}) \underset{\equiv}{\Delta} |x_i - c_{ji}|^2 \qquad \text{for } n \neq 0; \tag{3.8}$$

where $\underset{\equiv}{\Delta}$ denotes $n \rightarrow$ infinity, G(z) can be any function, although in most cases monotonic functions or Wavelets (such as (3.2) and (3.4)) or linear or polynomial functions are used. Note that the former functions are unbound, therefore they can approximate unbound functions, although they are more critical to train.

The RBF neural paradigms can be reconduced to WRBF networks (Reyneri, 1996). Here we underline that Wavelet Network is a specific case of WRBF, provided that we define:

$d = [w_1^2, w_2^2, \ldots, w_n^2]^T$,

$D = \text{diag}(d)$,

$\Phi(x) = F(\| x \|)$,

$n = 2$ and $\Theta = 0$,

Under these hypotheses, we have:

$$G(w^T \cdot \Delta_2(x - c)) = G(\Delta_2(d^T(x - c))) = \Phi(D(x - c)) \tag{3.9}$$

Table 3.1:Summaries of all the unifications results. The layers of several network structures as particular instances of WRBF layer

Kind of layer	Parameters of the equivalent WRBF layer
RBF hidden	n = 2, $\Theta = 0$, G(z) exponential
WNN hidden	n = 2, $\Theta = 0$, G(z) wavelet
RBF and WN output	n = 2, c = 0, G(z) linear

3.3.2 COMMENTS ON UNIFIED NEURO-WAVELET NETWORKS

All the Neuro-Wavelet Networks (NWNs) paradigms used in this work have been re-conduced to WRBF in order to have a common paradigm, methodology, initialization strategy and learning rule. In particular (from Eq.(3.1) and Eq.(3.3)):

- Radial wavelons are WRBF-2 with $w_{ij} = \left(\frac{1}{t_{ij}}\right)^2$ and $\underline{\underline{C}} = \underline{\underline{E}}$ (namely, the matrix made of one translation vector $\vec{E}$ per neuron), while the activation function comes from the radial Mother Wavelet G(z) = Φ ($\sqrt{z}$).
- R-neurons are WRBF-n (typically, n $\in$ {1, 2}) with $\underline{\underline{W}}$=1 (namely, a matrix of all w_{ij} = 1) and G(z) a generalized exponential with n = 1. With weight vector components equal to one, the input/output characteristic of the neuron is equivalent to that of a standard RBF (Wasserman, 1993), since the exponential function behaves as the gaussian, due to the exponent n = 2 from equation(3.2) (Reyneri, 1995).

It is worth noting that introducing an additional vector W to R-neurons is equivalent to multiplying each individual distance component (x_i- c_i) by a factor w_i. This solves two major drawbacks of most R-neurons: one is the dimensional problem, which does not have the same physical dimension; using weights w_i, each one with the proper physical dimension solves completely this problem. The other problem arises in classification and function approximation applications, when the optimal decision boundary may not be spherical, the weight vector W stretches the spherical boundary into a hyper-elliptical surface.

- Linear neurons (namely, linear combinations, weighted sums) are WRBF-0 neurons with $\underline{\underline{C}} = 0$ and a linear activation function.

- Function approximation: a generic function $y = F(\vec{X})$ can be decomposed as two cascaded WRBF layer:

$$y = H_0^{lin}\left(H_n^{G(z)}(\vec{X}; \underline{\underline{C}}_1; \underline{\underline{W}}_1; \vec{\Theta}_1); 0, \vec{W}_2, \Theta_2\right) \tag{3.10}$$

where the hidden layer is NWN, while the output layer is a linear layer. In other words, a linear combination (with $\vec{W}_2$ as weights) of a finite number of basis functions G(z), each one centered around a different point in the input space ($\underline{\underline{C}}_1$, usually on a multi-dimensional lattice of points) and appropriately dilated ($\underline{\underline{W}}_1$). The bias Θ_2 helps the approximation of functions with non-zero mean.

It is now clear that the various NWN paradigms mainly differ from each other for the order n and activation function G(z).It is known (Daubechies,1992) that, in several cases, the expansion of a signal into a wavelet series can be more efficient than other solutions, in the sense that fewer basis functions can be needed to achieve a predefined approximation error. This is due to the time-frequency local properties of most Wavelets, which make them particularly suitable to present short-time high-frequency signal features. Yet it has been shown that other types of function are better approximated by non-Wavelet neural networks (Colla,1998), therefore the activation function should be chosen according to the "shape" of the function to be approximated.

We will discuss this in Chapter 6, in our simulation problems using Matlab version 6. We use this concept of function approximation paradigms from this unification to approximate our functions. From the unifications we note that standard RBF and wavelet network can be seen as two cases of WRBFs, which are WRBF-2 and WRBF-0 (Reyneri, 1999).

Table 3.2: Examples of two-layer networks expressed in terms of cascaded WRBF layers. $G^j(z), \underline{\underline{C}}^j, \underline{\underline{W}}^j, \overrightarrow{\Theta}^j$ are respectively, activation function, center, weight, bias vector in jth layer.

Paradigm	Layers	Unified version
Radial Basis Function	2	$G_0^z\left(G_n^{e^{-\frac{z^2}{\sigma}}}\left(\overrightarrow{X};\underline{\underline{C}}^1;\underline{\underline{W}}^1;\vec{0}\right);\underline{0},\underline{\underline{W}}^2;\overrightarrow{\Theta}^2\right)$
Wavelet Neural Network	2	$G_0^z\left(G_2^{\Phi(z)}\left(\overrightarrow{X};\underline{\underline{C}}^1;\underline{\underline{W}}^1;\vec{0}\right);\underline{0},\underline{\underline{W}}^2;\overrightarrow{\Theta}^2\right)$

We can observe from the table above the structure of the networks between Radial Basis Function and Wavelet Neural Network design is similar except in the activation function. This has been proven (Zhang, *et al.,* 1995) when a wavelet frame replaces the radial basis function in an RBF network, the center and covariance matrix (spread) are replaced by the translations and dilations of the wavelets.

CHAPTER 4

TRANSFER FUNCTIONS, LEARNING AND TRAINING ALGORITHMS

The choice of transfer functions in neural networks is of crucial importance to their performance. Transfer functions may be used in the input pre-processing stage or as an integral part of the network. In the last case, transfer functions contain adaptive parameters that are optimized. The simplest approach is to test several networks with different transfer functions and select the best one. Constructive methods may also be used in training several candidate nodes and selecting the one that is the best performer.

4.1 TRANSFER FUNCTIONS AND THEIR PARAMETERIZATION

Transfer functions should provide flexibility of their contours with a small number of adaptive parameters. Large networks with simple neurons may have the same power as small networks with more complex neurons. Two functions determine the way signals are processed by neurons. The activation function acting on the input vector $I(\mathrm{x})$ determines the total signal a neuron receives, and the output function $o(\mathrm{I})$, operating on scalar activation, determines the scalar output. The composition of the activation and the output function is called the transfer function $o(I(\mathrm{x}))$. For some transfer function there is no natural division between activation and output functions.

For neuron i connected to neurons j (for j = 1,...,N) sending signals x_j with the strength of the connections w_j the total activation $I(x;w)$ is

$$I(x;w) = \sum_{j=0}^{N} w_j x_j \qquad i = 1,\ldots, N; j = 1,\ldots,N \qquad (4.1)$$

where $w_0 = \Theta$ (threshold or bias) and $x_0 = 1$. Three main choices for activation functions are:

- The inner product $I(x;w) \propto w^T.x$ (as in the MLP networks).
- The similarity- based activation $D(x;t) \propto \| x - t \|$, used to calculate similarity of x to a prototype vector t.
- A combination of the two activations, $A(x;w,t) \propto \alpha\, w^T.\, x + \beta \| x - t \|$.

In each case, we may use either the final scalar value of activation , or use the vector components of this activation, for example using the distance from which we usually take the scalar $D(x;t)$, but for some output functions we may also use the vector components $D_i(x_i ; t_i) \propto (x_i - t_i)^2$, for example

$$D_i(x_i ; t_i, b_i) = (x_i - t_i)^2/(b_i)^2 \qquad i= 1,\ldots, N \qquad (4.2)$$

The square of the activation function is a quadratic form. Treating all coefficients of this form as independent and transforming it into canonical form:

$$I^2(x;w) \sim D^2(x ; t; w) = \sum_{i=0}^{N} w_i \left(x_i' - t_i\right)^2 \qquad i= 1,\ldots, N \qquad (4.3)$$