Handbook of Neural Network Signal Processing

Handbook of NEURAL NETWORK SIGNAL PROCESSING

Edited by
YU HEN HU
JENQ-NENG HWANG

CRC PRESS

Boca Raton London New York Washington, D.C.

Library of Congress Cataloging-in-Publication Data

Handbook of neural network signal processing / editors, Yu Hen Hu, Jenq-Neng Hwang.
p. cm.— (Electrical engineering and applied signal processing (Series))
Includes bibliographical references and index.
ISBN 0-8493-2359-2
1. Neural networks (Computer science)—Handbooks, manuals, etc. 2. Signal processing—Handbooks, manuals, etc. I. Hu, Yu Hen. II. Hwang, Jenq-Neng. III. Electrical engineering and signal processing series.

QA76.87 H345 2001
006.3'2—dc21 2001035674

Direct all inquiries to CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, Florida 33431.

Visit the CRC Press Web site at www.crcpress.com

International Standard Book Number 0-8493-2359-2
Library of Congress Card Number 2001035674
Printed in the United States of America 1 2 3 4 5 6 7 8 9 0
Printed on acid-free paper

Handbook of NEURAL NETWORK SIGNAL PROCESSING

THE ELECTRICAL ENGINEERING
AND APPLIED SIGNAL PROCESSING SERIES
Edited by Alexander Poularikas

The Advanced Signal Processing Handbook:
Theory and Implementation for Radar, Sonar,
and Medical Imaging Real-Time Systems
Stergios Stergiopoulos

The Transform and Data Compression Handbook
K.R. Rao and P.C. Yip

Handbook of Multisensor Data Fusion
David Hall and James Llinas

Handbook of Neural Network Signal Processing
Yu Hen Hu and Jenq-Neng Hwang

Handbook of Antennas in Wireless Communications
Lal Chand Godara

Forthcoming Titles

Propagation Data Handbook for Wireless Communications
Robert Crane

The Digital Color Imaging Handbook
Guarav Sharma

Applications in Time Frequency Signal Processing
Antonia Papandreou-Suppappola

Noise Reduction in Speech Applications
Gillian Davis

Signal Processing in Noise
Vyacheslav Tuzlukov

Electromagnetic Radiation and the Human Body:
Effects, Diagnosis, and Therapeutic Technologies
Nikolaos Uzunoglu and Konstantina S. Nikita

Digital Signal Processing with Examples in MATLAB®
Samuel Stearns

Smart Antennas
Lal Chand Godara

Pattern Recognition in Speech and Language Processing
Wu Chou and Bing Huang Juang

Preface

The field of artificial neural networks has made tremendous progress in the past 20 years in terms of theory, algorithms, and applications. Notably, the majority of real world neural network applications have involved the solution of difficult statistical signal processing problems. Compared to conventional signal processing algorithms that are mainly based on linear models, artificial neural networks offer an attractive alternative by providing nonlinear parametric models with universal approximation power, as well as adaptive training algorithms. The availability of such powerful modeling tools motivated numerous research efforts to explore new signal processing applications of artificial neural networks. During the course of the research, many neural network paradigms were proposed. Some of them are merely reincarnations of existing algorithms formulated in a neural network-like setting, while the others provide new perspectives toward solving nonlinear adaptive signal processing. More importantly, there are a number of emergent neural network paradigms that have found successful real world applications.

The purpose of this handbook is to survey recent progress in artificial neural network theory, algorithms (paradigms) with a special emphasis on signal processing applications. We invited a panel of internationally well known researchers who have worked on both theory and applications of neural networks for signal processing to write each chapter. There are a total of 12 chapters plus one introductory chapter in this handbook. The chapters are categorized into three groups. The first group contains in-depth surveys of recent progress in neural network computing paradigms. It contains five chapters, including the introduction, that deal with multilayer perceptrons, radial basis functions, kernel-based learning, and committee machines. The second part of this handbook surveys the neural network implementations of important signal processing problems. This part contains four chapters, dealing with a dynamic neural network for optimal signal processing, blind signal separation and blind deconvolution, a neural network for principal component analysis, and applications of neural networks to time series predictions. The third part of this handbook examines signal processing applications and systems that use neural network methods. This part contains chapters dealing with applications of artificial neural networks (ANNs) to speech processing, learning and adaptive characterization of visual content in image retrieval systems, applications of neural networks to biomedical image processing, and a hierarchical fuzzy neural network for pattern classification.

The theory and design of artificial neural networks have advanced significantly during the past 20 years. Much of that progress has a direct bearing on signal processing. In particular, the nonlinear nature of neural networks, the ability of neural networks to learn from their environments in supervised and/or unsupervised ways, as well as the universal approximation property of neural networks make them highly suited for solving difficult signal processing problems.

From a signal processing perspective, it is imperative to develop a proper understanding of basic neural network structures and how they impact signal processing algorithms and applications. A challenge in surveying the field of neural network paradigms is to distinguish those neural network structures that have been successfully applied to solve real world problems from those that are still under development or have difficulty scaling up to solve realistic problems. When dealing with signal processing applications, it is critical to understand the nature of the problem formulation so that the most appropriate neural network paradigm can be applied. In addition, it is also important to assess the impact of neural networks on the performance, robustness, and cost-effectiveness of signal processing systems and develop methodologies for integrating neural networks with other signal processing algorithms.

We would like to express our sincere thanks to all the authors who contributed to this handbook: Michael T. Manry, Hema Chandrasekaran, and Cheng-Hsiung Hsieh (Chapter 2); Andrew D. Back (Chapter 3); Klaus-Robert Müller, Sebastian Mika, Gunnar Rätsch, Koji Tsuda, and Bernhard Scholköpf (Chapter 4); Volker Tresp (Chapter 5); Jose C. Principe (Chapter 6); Scott C. Douglas (Chapter 7); Konstantinos I. Diamantaras (Chapter 8); Yuansong Liao, John Moody, and Lizhong Wu (Chapter 9); Shigeru Katagirig (Chapter 10); Paisarn Muneesawang, Hau-San Wong, Jose Lay, and Ling Guan (Chapter 11); Tülay Adali, Yue Wang, and Huai Li (Chapter 12); and Jinshiuh Taur, Sun-Yuan Kung, and Shang-Hung Lin (Chapter 13). Many reviewers have carefully read the manuscript and provided many constructive suggestions. We are most grateful for their efforts. They are Andrew D. Back, David G. Brown, Laiwan Chan, Konstantinos I. Diamantaras, Adriana Dumitras, Mark Girolami, Ling Guan, Kuldip Paliwal, Amanda Sharkey, and Jinshiuh Taur.

We would like to thank the editor-in-chief of this series of handbooks, Dr. Alexander D. Poularikas, for his encouragement. Our most sincere appreciation to Nora Konopka at CRC Press for her infinite patience and understanding throughout this project.

Editors

Yu Hen Hu received a B.S.E.E. degree from National Taiwan University, Taipei, Taiwan, in 1976. He received M.S.E.E. and Ph.D. degrees in electrical engineering from the University of Southern California in Los Angeles, in 1980 and 1982, respectively. From 1983 to 1987, he was an assistant professor in the electrical engineering department of Southern Methodist University in Dallas, Texas. He joined the department of electrical and computer engineering at the University of Wisconsin in Madison, as an assistant professor in 1987, and he is currently an associate professor. His research interests include multimedia signal processing, artificial neural networks, fast algorithms and design methodology for application specific micro-architectures, as well as computer aided design tools for VLSI using artificial intelligence. He has published more than 170 technical papers in these areas. His recent research interests have focused on image and video processing and human computer interface.

Dr. Hu is a former associate editor for *IEEE Transactions of Acoustic, Speech, and Signal Processing* in the areas of system identification and fast algorithms. He is currently associate editor of the *Journal of VLSI Signal Processing*. He is a founding member of the Neural Network Signal Processing Technical Committee of the IEEE Signal Processing Society and served as committee chair from 1993 to 1996. He is a former member of the VLSI Signal Processing Technical Committee of the Signal Processing Society. Recently, he served as the secretary of the IEEE Signal Processing Society (1996–1998).

Dr. Hu is a fellow of the IEEE.

Jenq-Neng Hwang holds B.S. and M.S. degrees in electrical engineering from the National Taiwan University, Taipei, Taiwan. After completing two years of obligatory military services after college, he enrolled as a research assistant at the Signal and Image Processing Institute of the department of electrical engineering at the University of Southern California, where he received his Ph.D. degree in December 1988. He was also a visiting student at Princeton University from 1987 to 1989.

In the summer of 1989, Dr. Hwang joined the Department of Electrical Engineering of the University of Washington in Seattle, where he is currently a professor. He has published more than 150 journal and conference papers and book chapters in the areas of image/video signal processing, computational neural networks, and multimedia system integration and networking. He received the 1995 IEEE Signal Processing Society's Annual Best Paper Award (with Shyh-Rong Lay and Alan Lippman) in the area of neural networks for signal processing.

Dr. Hwang is a fellow of the IEEE. He served as the secretary of the Neural Systems and Applications Committee of the IEEE Circuits and Systems Society from 1989 to 1991, and he was a member of the Design and Implementation of Signal Processing Systems Technical Committee of the IEEE Signal Processing Society. He is also a founding member of the Multimedia Signal Processing Technical Committee of the IEEE Signal Processing Society. He served as the chairman of the Neural Networks Signal Processing Technical Committee of the IEEE Signal Processing Society from 1996 to 1998, and he is currently the Society's representative to the IEEE Neural Network Council. He served as an associate editor for *IEEE Transactions on Signal Processing* from 1992 to 1994 and currently is the associate editor for *IEEE Transactions on Neural Networks* and *IEEE Transactions on Circuits and Systems for Video Technology*. He is also on the editorial board of the *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*. Dr. Hwang was the conference program chair of the 1994 IEEE Workshop on Neural Networks for Signal Processing held in Ermioni, Greece in September 1994. He was the general co-chair of the International Symposium on

Artificial Neural Networks held in Hsinchu, Taiwan in December 1995. He also chaired the tutorial committee for the IEEE International Conference on Neural Networks held in Washington, D.C. in June 1996. He was the program co-chair of the International Conference on Acoustics, Speech, and Signal Processing in Seattle, Washington in 1998.

Contributors

Tülay Adali
University of Maryland
Baltimore, Maryland

Andrew D. Back
Windale Technologies
Brisbane, Australia

Hema Chandrasekaran
U.S. Wireless Corporation
San Ramon, California

Konstantinos I. Diamantaras
Technological Education Institute of Thessaloniki
Sindos, Greece

Scott C. Douglas
Southern Methodist University
Dallas, Texas

Ling Guan
University of Sydney
Sydney, Australia

Cheng-Hsiung Hsieh
Chien Kou Institute of Technology
Changwa
Taiwan, China

Yu Hen Hu
University of Wisconsin
Madison, Wisconsin

Jenq-Neug Hwang
University of Washington
Seattle, Washington

Shigeru Katagiri
Intelligent Communication Science Laboratories
Kyoto, Japan

Sun-Yuan Kung
Princeton University
Princeton, New Jersey

Jose Lay
University of Sydney
Sydney, Australia

Huai Li
University of Maryland
Baltimore, Maryland

Yuansong Liao
Oregon Graduate Institute of Science and Technology
Beaverton, Oregon

Shang-Hung Lin
EPSON Palo Alto Laboratories
ERD
Palo Alto, California

Michael T. Manry
University of Texas
Arlington, Texas

Sebastian Mika
GMD FIRST
Berlin, Germany

John Moody
Oregon Graduate Institute of Science and Technology
Beaverton, Oregon

Klaus-Robert Müler
GMD FIRST and University of Potsdam
Berlin, Germany

Paisarn Muneesawang
University of Sydney
Sydney, Australia

Jose C. Principe
University of Florida
Gainesville, Florida

Gunnar Rätsch
GMD FIRST and University of Potsdam
Berlin, Germany

Bernhard Schölkopf
Max-Planck-Institut für Biologische Kybernetik
Tübingen, Germany

Junshiuh Taur
National Chung-Hsing University
Taichung
Taiwan, China

Volker Tresp
Siemens AG
Corporate Technology
Munich, Germany

Koji Tsuda
AIST Computational Biology Research Center
Tokyo, Japan

Yue Wang
The Catholic Universtiy of America
Washington, DC

Hau-San Wong
University of Sydney
Sydney, Australia

Lizhong Wu
HNC Software, Inc.
San Diego, California

Contents

1

Introduction to Neural Networks for Signal Processing

Yu Hen Hu
University of Wisconsin

Jenq-Neng Hwang
University of Washington

1.1 Introduction

The theory and design of artificial neural networks have advanced significantly during the past 20 years. Much of that progress has a direct bearing on signal processing. In particular, the nonlinear nature of neural networks, the ability of neural networks to learn from their environments in supervised as well as unsupervised ways, as well as the universal approximation property of neural networks make them highly suited for solving difficult signal processing problems.

From a signal processing perspective, it is imperative to develop a proper understanding of basic neural network structures and how they impact signal processing algorithms and applications. A challenge in surveying the field of neural network paradigms is to identify those neural network structures that have been successfully applied to solve real world problems from those that are still under development or have difficulty scaling up to solve realistic problems. When dealing with signal processing applications, it is critical to understand the nature of the problem formulation so that the most appropriate neural network paradigm can be applied. In addition, it is also important to assess the impact of neural networks on the performance, robustness, and cost-effectiveness of signal processing systems and develop methodologies for integrating neural networks with other signal processing algorithms. Another important issue is how to evaluate neural network paradigms, learning algorithms, and neural network structures and identify those that do and do not work reliably for solving signal processing problems.

This chapter provides an overview of the topic of this handbook — neural networks for signal processing. The chapter first discusses the definition of a neural network for signal processing and why it is important. It then surveys several modern neural network models that have found successful signal processing applications. Examples are cited relating to how to apply these nonlinear

0-8493-2359-2/01/$0.00+$1.50

computation paradigms to solve signal processing problems. Finally, this chapter highlights the remaining contents of this book.

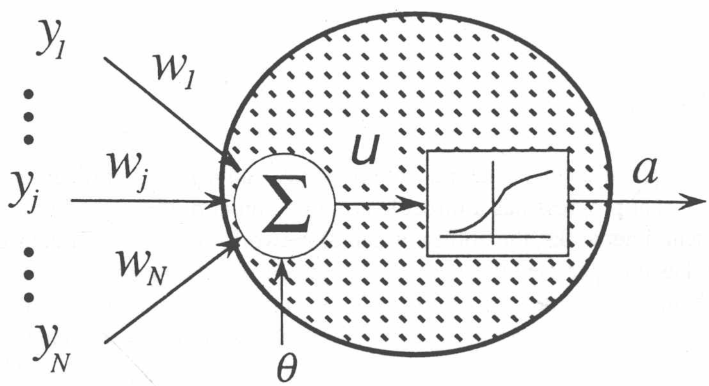
1.2 Artificial Neural Network (ANN) Models — An Overview

1.2.1 Basic Neural Network Components

A neural network is a general mathematical computing paradigm that models the operations of biological neural systems. In 1943, McCulloch, a neurobiologist, and Pitts, a statistician, published a seminal paper titled "A logical calculus of ideas imminent in nervous activity" in *Bulletin of Mathematical Biophysics* [1]. This paper inspired the development of the modern digital computer, or the electronic brain, as John von Neumann called it. At approximately the same time, Frank Rosenblatt was also motivated by this paper to investigate the computation of the eye, which eventually led to the first generation of neural networks, known as the perceptron [2]. This section provides a brief overview of ANN models. Many of these topics will be treated in greater detail in later chapters. The purpose of this chapter, therefore, is to highlight the basic concept of these neural network models to prepare the readers for later chapters.

1.2.1.1 McCulloch and Pitts' Neuron Model

Among numerous neural network models that have been proposed over the years, all share a common building block known as a neuron and a networked interconnection structure. The most widely used neuron model is based on McCulloch and Pitts' work and is illustrated in Figure 1.1.



1.1 McCulloch and Pitts' neuron model.

In Figure 1.1, each neuron consists of two parts: the net function and the activation function. The net function determines how the network inputs $\{y_j; 1 \le j \le N\}$ are combined inside the neuron. In this figure, a weighted linear combination is adopted:

$$u = \sum_{j=1}^{N} w_j y_j + \theta \tag{1.1}$$

$\{w_j; 1 \le j \le N\}$ are parameters known as synaptic weights. The quantity θ is called the bias (or threshold) and is used to model the threshold. In the literature, other types of network input combination methods have been proposed as well. They are summarized in Table 1.1.

TABLE 1.1 Summary of Net Functions

Net Functions	Formula	Comments
Linear	$u = \sum_{j=1}^{N} w_j y_j + \theta$	Most commonly used
Higher order (2nd order formula exhibited)	$u = \sum_{j=1}^{N} \sum_{k=1}^{N} w_{jk} y_j y_k + \theta$	u_i is a weighted linear combination of higher order polynomial terms of input variable. The number of input terms equals N^d, where d is the order of the polynomial
Delta ($\sum - \prod$)	$u = \prod_{j=1}^{N} w_j y_j$	Seldom used

The output of the neuron, denoted by a_i in this figure, is related to the network input u_i via a linear or nonlinear transformation called the activation function:

$$a = f(u) . \tag{1.2}$$

In various neural network models, different activation functions have been proposed. The most commonly used activation functions are summarized in Table 1.2.

TABLE 1.2 Neuron Activation Functions

Activation Function	Formula $a = f(u)$	Derivatives $\frac{df(u)}{du}$	Comments
Sigmoid	$f(u) = \frac{1}{1+e^{-u/T}}$	$f(u)[1 - f(u)]/T$	Commonly used; derivative can be computed from $f(u)$ directly.
Hyperbolic tangent	$f(u) \tanh\left(\frac{u}{T}\right)$	$\left(1 - [f(u)]^2\right)/T$	T = temperature parameter
Inverse tangent	$f(u) = \frac{2}{\pi} \tan^{-1}\left(\frac{u}{T}\right)$	$\frac{2}{\pi T} \cdot \frac{1}{1+(u/T)^2}$	Less frequently used
Threshold	$f(u) = \begin{cases} 1 & u > 0; \\ -1 & u < 0. \end{cases}$	Derivatives do not exist at $u = 0$	
Gaussian radial basis	$f(u) = \exp\left[-\Vert u - m\Vert^2/\sigma^2\right]$	$-2(u - m) \cdot f(u)/\sigma^2$	Used for radial basis neural network; m and σ^2 are parameters to be specified
Linear	$f(u) = au + b$	a	

Table 1.2 lists both the activation functions as well as their derivatives (provided they exist). In both sigmoid and hyperbolic tangent activation functions, derivatives can be computed directly from the knowledge of $f(u)$.

1.2.1.2 Neural Network Topology

In a neural network, multiple neurons are interconnected to form a network to facilitate distributed computing. The configuration of the interconnections can be described efficiently with a directed graph. A directed graph consists of nodes (in the case of a neural network, neurons, as well as external inputs) and directed arcs (in the case of a neural network, synaptic links).

The topology of the graph can be categorized as either acyclic or cyclic. Refer to Figure 1.2a; a neural network with acyclic topology consists of no feedback loops. Such an acyclic neural network is often used to approximate a nonlinear mapping between its inputs and outputs. As shown in Figure 1.2b, a neural network with cyclic topology contains at least one cycle formed by directed arcs. Such a neural network is also known as a recurrent network. Due to the feedback loop, a recurrent network leads to a nonlinear dynamic system model that contains internal memory. Recurrent neural networks often exhibit complex behaviors and remain an active research topic in the field of artificial neural networks.

(a) Acyclic topology (b) Cyclic topology

1.2 Illustration of (a) an acyclic graph and (b) a cyclic graph. The cycle in (b) is emphasized with thick lines.

1.2.2 Multilayer Perceptron (MLP) Model

The multilayer perceptron [3] is by far the most well known and most popular neural network among all the existing neural network paradigms. To introduce the MLP, let us first discuss the perceptron model.

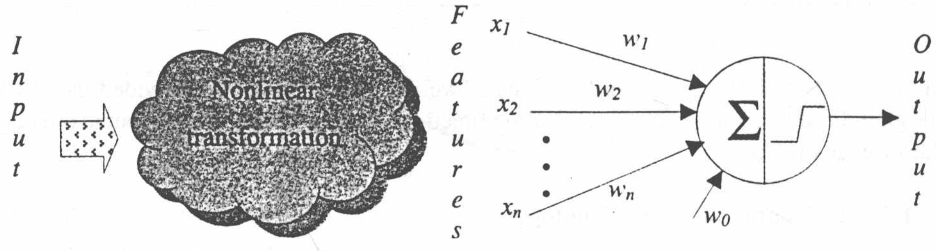
1.2.2.1 Perceptron Model

An MLP is a variant of the original perceptron model proposed by Rosenblatt in the 1950s [2]. In the perceptron model, a single neuron with a linear weighted net function and a threshold activation function is employed. The input to this neuron $\underline{x} = (x_1, x_2, \ldots, x_n)$ is a feature vector in an n-dimensional feature space. The net function $u(\underline{x})$ is the weighted sum of the inputs:

$$u(\underline{x}) = w_0 + \sum_{i=1}^{n} w_i x_i \tag{1.3}$$

and the output $y(\underline{x})$ is obtained from $u(\underline{x})$ via a threshold activation function:

$$y(\underline{x}) = \begin{cases} 1 & u(\underline{x}) \geq 0 \\ 0 & u(\underline{x}) < 0 \, . \end{cases} \tag{1.4}$$



1.3 A perceptron neural network model.

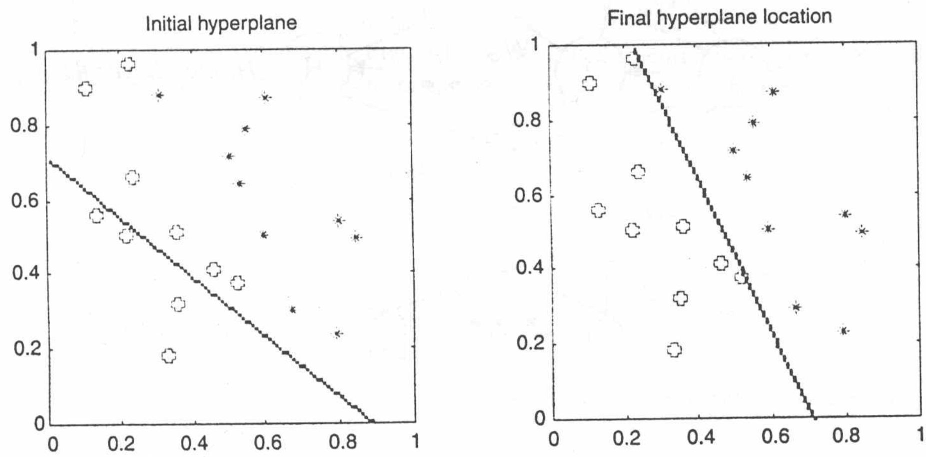
The perceptron neuron model can be used for detection and classification. For example, the weight vector $\underline{w} = (w_1, w_2, \ldots, w_n)$ may represent the template of a certain target. If the input feature vector $\underline{x}$ closely matches $\underline{w}$ such that their inner product exceeds a threshold $-w_0$, then the output will become $+1$, indicating the detection of a target.

The weight vector $\underline{w}$ needs to be determined in order to apply the perceptron model. Often, a set of training samples $\{(\underline{x}(i), d(i)); i \in I_r\}$ and testing samples $\{(\underline{x}(i), d(i)); i \in I_t\}$ are given. Here, $d(i) (\in \{0, 1\})$ is the desired output value of $y(\underline{x}(i))$ if the weight vector $\underline{w}$ is chosen correctly, and I_r and I_t are disjoined index sets. A sequential online perceptron learning algorithm can be applied to iteratively estimate the correct value of $\underline{w}$ by presenting the training samples to the perceptron neuron in a random, sequential order. The learning algorithm has the following formulation:

$$\underline{w}(k+1) = \underline{w}(k) + \eta(d(k) - y(k))\underline{x}(k) \tag{1.5}$$

where $y(k)$ is computed using Equations (1.3) and (1.4). In Equation (1.5), the learning rate $\eta(0 < \eta < 1/|\underline{x}(k)|_{max})$ is a parameter chosen by the user, where $|\underline{x}(k)|_{max}$ is the maximum magnitude of the training samples $\{\underline{x}(k)\}$. The index k is used to indicate that the training samples are applied sequentially to the perceptron in a random order. Each time a training sample is applied, the corresponding output of the perceptron $y(k)$ is to be compared with the desired output $d(k)$. If they are the same, meaning the weight vector $\underline{w}$ is correct for this training sample, the weights will remain unchanged. On the other hand, if $y(k) \neq d(k)$, then $\underline{w}$ will be updated with a small step along the direction of the input vector $\underline{x}(k)$. It has been proven that if the training samples are linearly separable, the perceptron learning algorithm will converge to a feasible solution of the weight vector within a finite number of iterations. On the other hand, if the training samples are not linearly separable, the algorithm will not converge with a fixed, nonzero value of η.

MATLAB Demonstration Using MATLAB m-files `perceptron.m`, `datasepf.m`, and `sline.m`, we conducted a simulation of a perceptron neuron model to distinguish two separable data samples in a two-dimensional unit square. Sample results are shown in Figure 1.4.



1.4 Perceptron simulation results. The figure on the left-hand side depicts the data samples and the initial position of the separating hyperplane, whose normal vector contains the weights to the perceptron. The right-hand side illustrates that the learning is successful as the final hyperplane separates the two classes of data samples.

1.2.2.1.1 Applications of the Perceptron Neuron Model

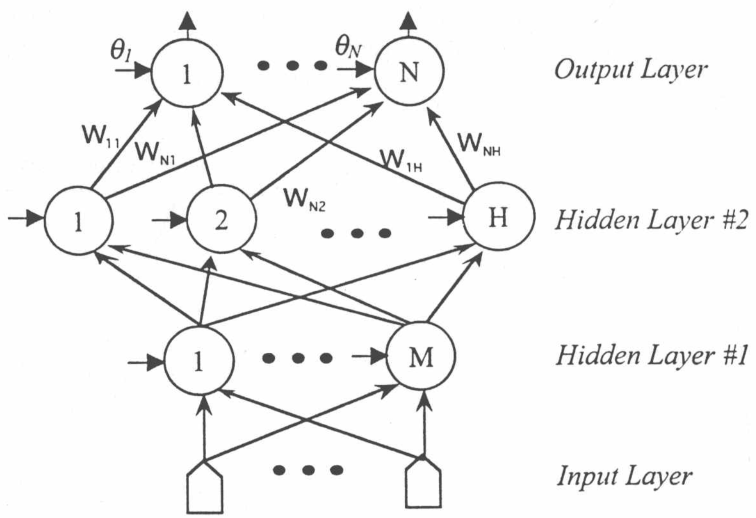
There are several major difficulties in applying the perceptron neuron model to solve real world pattern classification and signal detection problems:

1. The nonlinear transformation that extracts the appropriate feature vector x is not specified.
2. The perceptron learning algorithm will not converge for a fixed value of learning rate η if the training feature patterns are not linearly separable.
3. Even though the feature patterns are linearly separable, it is not known how long it takes for the algorithm to converge to a weight vector that corresponds to a hyperplane that separates the feature patterns.

1.2.2.2 Multilayer Perceptron

A multilayer perceptron (MLP) neural network model consists of a feed-forward, layered network of McCulloch and Pitts' neurons. Each neuron in an MLP has a nonlinear activation function that is often continuously differentiable. Some of the most frequently used activation functions for MLP include the sigmoid function and the hyperbolic tangent function.

A typical MLP configuration is depicted in Figure 1.5. Each circle represents an individual neuron. These neurons are organized in layers, labeled as the hidden layer #1, hidden layer #2, and the output layer in this figure. While the inputs at the bottom are also labeled as the input layer, there is usually no neuron model implemented in that layer. The name *hidden layer* refers to the fact that the output of these neurons will be fed into upper layer neurons and, therefore, is hidden from the user who only observes the output of neurons at the output layer. Figure 1.5 illustrates a popular configuration of MLP where interconnections are provided only between neurons of successive layers in the network. In practice, any acyclic interconnections between neurons are allowed.



1.5 A three-layer multilayer perceptron configuration.

An MLP provides a nonlinear mapping between its input and output. For example, consider the following MLP structure (Figure 1.6) where the input samples are two-dimensional grid points, and the output is the z-axis value. Three hidden nodes are used, and the sigmoid function has a parameter $T = 0.5$. The mapping is plotted on the right side of Figure 1.6. The nonlinear nature of this mapping is quite clear from the figure. The MATLAB m-files used in this demonstration are `mlpdemo1.m` and `mlp2.m`.

It has been proven that with a sufficient number of hidden neurons, an MLP with as few as two hidden layer neurons is capable of approximating an arbitrarily complex mapping within a finite support [4].

1.2.2.3 Error Back-Propagation Training of MLP

A key step in applying an MLP model is to choose the weight matrices. Assuming a layered MLP structure, the weights feeding into each layer of neurons form a weight matrix of that layer (the input layer does not have a weight matrix as it contains no neurons). The values of these weights are found using the error back-propagation training method.