

神经元网络系统设计方法

NEURAL NETWORKS SYSTEM DESIGN METHODOLOGY

David D. Zhang

*Department of Computer Science
City University of Hong Kong
Kowloon, Hong Kong*



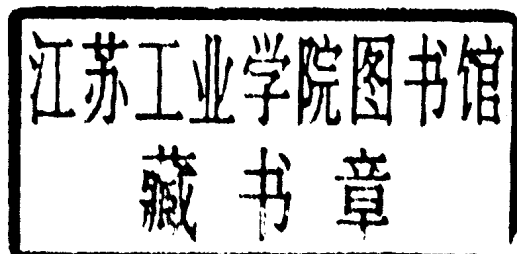
TSINGHUA UNIVERSITY PRESS

NEURAL NETWORKS SYSTEM DESIGN METHODOLOGY

神经元网络系统设计方法

David D. Zhang

*Department of Computer Science
City University of Hong Kong
Kowloon, Hong Kong*



清华大学出版社

(京)新登字 158 号

内 容 简 介

本书从讨论神经网络的模型入手,系统阐述了神经网络系统设计原理和方法及应用,并着重介绍了如何用 VLSI(大规模集成电路)实现神经网络。本专著包含了作者在该领域的研究成果,提供了参考资料目录。既是人工智能、计算机和微电子学方面的技术参考书,又可作高年级大学生和研究生的专业教材。

© Tsinghua University Press, Beijing, China, 1996, All rights reserved.

© 清华大学出版社,中国北京,1996。版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

神经网络系统设计方法:英文/张大鹏著. —北京:

清华大学出版社,1996

ISBN 7-302-02163-5

I. 神… II. 张… III. 神经网络-网络系统-系统设计-方法 IV. TP393

中国版本图书馆 CIP 数据核字(96)第07792号

出版者:清华大学出版社(北京清华大学校内,邮编100084)

印刷者:清华大学印刷厂

发行者:新华书店总店北京科技发行所

开 本:787×1092 1/16 印张:12 字数:280千字

版 次:1996年5月第1版 1996年5月第1次印刷

书 号:ISBN 7-302-02193-5/TP·1092

印 数:0001—1200

定 价:26.00元

NEURAL NETWORKS SYSTEM DESIGN METHODOLOGY

Neural Networks System Design Methodology presents an integrated design approach to this new engineering discipline. Three topics of the system design methodology, including several model definitions, architectural descriptions, and hardware implementations, are investigated and their coordination are discussed in detail after an overview of the field of ANNs. Engineering applications of ANNs to fuzzy clustering, speech recognition, classification and pattern recognition are covered. The book can be used as a text book or reference for graduate or senior undergraduate courses on the subject. It can also be used by researchers in the field.

David Zhang graduated in computer science from Peking University in 1974 and received his M.Sc and Ph.D degrees in computer science and engineering from Harbin Institute of Technology (HIT) in 1983 and 1985, respectively. From 1986 to 1988 he was a postdoctoral fellow at Tsinghua University and became an associate professor at Academia Sinica, Beijing, China. In 1988, he joined the University of Windsor, Ontario, Canada, as a visiting professor in electrical engineering. He received his second Ph.D in electrical and computer engineering at University of Waterloo, Ontario, Canada, in 1994. Currently, he is an associate professor in City University of Hong Kong. He has authored and co-authored near 100 papers including two books, as well as received several recognisable project awards. Dr. Zhang is a senior member of the IEEE.

*To the memory of my beloved supervisor
—Professor Tong Chang*

Preface

Researchers and engineers have long been fascinated by how efficient and how fast biological neural networks are capable of performing such complex tasks as recognition. Such networks are capable of recognizing input data from *any of the five senses* with the necessary accuracy and speed to allow living creatures to survive. Machines which perform such complex tasks as recognition, with similar accuracy and speed, were difficult to be implemented until the technological advances of VLSI circuits and systems in the late 1980's. Since then, the field of Artificial Neural Networks (ANNs) have witnessed an exponential growth and a new engineering discipline was born. Today, many engineering curriculums have included a course or more on the subject at the graduate or senior undergraduate levels.

This book attempts to present a system design methodology of ANNs for pattern recognition applications. The methodology emphasizes a coordination between model definition, architectural description, and hardware implementation. Depending on the different pattern recognition applications, the methodology provides appropriate ANN models suited to parallel / pipeline processing, mapping the models onto the corresponding VLSI architectures and finally VLSI implementation. The book discusses these three phases:

1. **Parallel ANN Model:** Three types of models, an unsupervised learning model for fuzzy clustering, a supervised training model for pattern classification and a neural-like network model for finite ring computing, are developed. Compared with the conventional approaches, the new models can greatly reduce the complexity of the VLSI implementation.
2. **VLSI Architecture:** Three typical architectures, including a parallel architecture built by systolic arrays, a pipeline architecture based on window operation and a simplified architecture using a priori knowledge, are designed. They are all easily implemented in VLSI medium.
3. **Hardware Implementation:** Two design approaches are investigated. One is a digital array compressor design based on a complex complementary pass-transistor logic (C²PL) and the other is a hybrid programmable ANN design using BiCMOS circuit building blocks. As an example, a VLSI implementation for finite ring neural network is developed. Our simulation results show their advantages in power, time and area.

The effectiveness of neural network system design methodology is illustrated by applying the designs to various pattern recognition applications, and analyzing the performances of the given systems.

It is my hope that this book will contribute to our understanding of this new and exciting discipline; ANNs System Engineering.

David D. Zhang
City University of Hong Kong

Acknowledgements

My sincere thanks goes to Professors. M.I. Elmasry and M. Kamel at University of Waterloo, Ontario, Canada, for their support and advice throughout this research. I would like to thank Dr. Harold H. Szu for his useful comments on neural networks. I would also like to express my gratitude to Professors. G.A. Jullien and W.C. Miller at University of Windsor, Ontario, Canada, for their previous supports and their encouragement to me to study VLSI design.

CONTENTS

1	INTRODUCTION	1
1.1	ANNs for Pattern Recognition	1
1.2	Neural Network Model	3
1.3	ANN Architecture	6
1.4	Hardware Implementation of ANN	7
1.5	A VLSI System Design Methodology	10

PART I PARALLEL ANN MODELS

2	AN UNSUPERVISED LEARNING MODEL	13
2.1	Introduction	13
2.2	Fuzzy Clustering Neural Networks	17
2.3	Parallel FCNN Architecture	20
2.4	Experimental Results	21
2.5	Summary	24
3	A SUPERVISED TRAINING MODEL	29
3.1	Introduction	29
3.2	Linear Separability Analysis	31
3.3	Layer Adaptation Approach	35
3.4	Experiment: Pattern Recognition	39
3.5	Comparisons	41
3.6	Summary	42
4	A NEURAL-LIKE NETWORK MODEL	45
4.1	Introduction	45
4.2	FRNN Computing Model	49
4.3	FRNN Architecture	51

4.4	Case Studies	52
4.5	Summary	54

PART II VLSI ARCHITECTURES

5	A PARALLEL ARCHITECTURE IMPLEMENTED BY SYSTOLIC ARRAYS	57
5.1	Introduction	57
5.2	FCNN Architecture	58
5.3	Mapping Policies	63
5.4	Typical SA Structures	65
5.5	Mapping FCNN onto SA	68
5.6	Summary	70
6	A PIPELINED ARCHITECTURE BASED ON WINDOW OPERATION	73
6.1	Introduction	73
6.2	Pipelined Architecture: Window Operation	73
6.3	Window Implementation	80
6.4	Case Studies	84
6.5	Performance Analysis	89
6.6	Summary	92
7	A SIMPLIFIED ARCHITECTURE USING A PRIORI KNOWLEDGE	93
7.1	Introduction	93
7.2	Basic Concepts	93
7.3	Typical Structure Models	96
7.4	ROM Layer in VLSI	97
7.5	Examples	98
7.6	Summary	107

PART III HARDWARE IMPLEMENTATIONS

8	DIGITAL ANN COMPRESSOR DESIGN	109
8.1	Introduction	109
8.2	C ² PL Model	110
8.3	3-2 Compressor Design	111
8.4	DNN Applications	121

8.5	Summary	123
9	HYBRID PROGRAMMABLE ANN DESIGN	125
9.1	Introduction	125
9.2	Analysis and Design for PRNN	127
9.3	Improved PRNN Circuit	131
9.4	Experimental Results	135
9.5	Summary	136
	Appendix A	137
10	VLSI IMPLEMENTATION FOR FINITE RING ANN	139
10.1	Introduction	139
10.2	FRNN Review	139
10.3	FRRR Architecture	140
10.4	VLSI Implementation	146
10.5	Comparison	151
10.6	Summary	156
	Appendix B	158
11	CONCLUSIONS AND PROSPECTS	159
	BIBLIOGRAPHY	165
	INDEX	175

LIST OF TABLES

1.1	Comparison between analog and digital implementations	9
1.2	Comparison between analog and digital design technology	9
2.1	The simulation result for different values of σ	26
2.2	FCNN result for the British towns data set using different number of nodes, M	26
2.3	Comparison of the performance of the FCNN and the FCM FCM algorithm on the Butterfly data set	27
2.4	Cluster centers produced by the FCNN and the FCM for the British towns data set	27
3.1	The state table for XOR network	35
5.1	Comparison of the building elements in the two architectures	64
7.1	The relation of Δw_{ij} , $\Delta(-1)$ and $\Delta(+1)$	108
7.2	The truth table of the digital sensor for the weight, w_{ij}	108
8.1	Comparison of 3-2 compressors between CPL and C^2PL	120
8.2	Comparison of 4-2 compressors	121
8.3	Comparison of 7-3 compressors	121
10.1	Comparison of the different architectures	156

LIST OF FIGURES

1.1	A prototype biological neuron	2
1.2	A typical model of a neuron	4
1.3	A three-layer feedforward network	4
1.4	Commonly used activation functions	5
1.5	Typical ANN architectures	6
1.6	ANN implementation approaches	8
1.7	A VLSI system design methodology	11
2.1	Learning vector quantization (LVQ) network model	14
2.2	Radial basis function (RBF) network model	15
2.3	Fuzzy clustering neural network (FCNN)	16
2.4	Parallel FCNN model with special feedforward and feedback paths	20
2.5	A data set with 50 points in 2 clusters centered at $\{(2.0, 2.0), (-2.0, -2.0)\}$	23
2.6	The quality of fit for different values of η ($\sigma = 0.7$)	23
2.7	The moving track for the weights in the input space	24
2.8	The simulation result of applying the FCNN to the Butterfly data set	25
3.1	A perceptron model with binary input / output	30
3.2	A LSL function using a hyperplane	33
3.3	The diagram of the two-layer X	33
3.4	Pattern combinations in XOR network	34
3.5	Two-layer network adaptation diagram	38
3.6	The standard input patterns	39
3.7	The translational and rotational input patterns	39
3.8	The diagram of the error changing with weight increment in the XOR network	40
3.9	The adaptation pattern distribution system at the kth hidden layer (\otimes : Selector)	43
3.10	The error generation system for DLP (\oplus : Adder ; $k = 1, 2, \dots, S$)	43
4.1	A basic ANN model	48
4.2	FRNN Mapping	48
4.3	The operator subnet and its symbol	52
4.4	Outputs of multiplier and iteration blocks	53
4.5	FRNN architecture of an RNS \rightarrow Binary conversion	55
5.1	The FCNN architecture built by processing cells	58
5.2	Block diagram of the output processing cell	59
5.3	Block diagram of the weight processing cell	61
5.4	Fuzzy c-means neural network (FCMN) architecture	62
5.5	(a) A basic fuzzy clustering ANN computing model and (b) its SA structure	66
5.6	(a) A typical feedforward subnet model and (b) its SA structure	66

5.7	(a) A typical feedback subnet model and (b) its SA structure	67
5.8	The SA implementation for Fig.5.1	68
5.9	Input / output data flows for the feedforward SA	69
5.10	Triangle SA with M adder PEs and $(M[M-1])/2$ registers	69
5.11	Weight PE in systolic implementation	70
5.12	The SA with the learning path	71
6.1	Window model built by input matrix in Layer s	74
6.2	(a) Pipelined ANN architecture model; (b) Pipelined neuron unit	75
6.3	Parallel PNN processing stage	77
6.4	Serial PNN processing stage	77
6.5	Synapse building unit structure	78
6.6	Summing building unit structure	78
6.7	Connection network with on-line learning in Stage s	80
6.8	Parallel data flow window computation	81
6.9	Serial data flow window computation	82
6.10	Computational element structure in window operation	83
6.11	Relation between layers for TDNN	84
6.12	Relation between layers for BWNN	85
6.13	(a) Serial processing stage for DYNM and (b) its PE structure	86
6.14	Window structures of the DYNM implementation	87
6.15	The window process for DYNM implementation	88
6.16	Selection of the window for Type 1	92
6.17	Selection of the window for Type 2	92
7.1	The block diagram of ANN-ROM structure design	93
7.2	Output ROM structure model	95
7.3	Input ROM structure model	95
7.4	Learning ROM structure model	95
7.5	Design procedure of full-adder using the reducing rules	98
7.6	A content-addressable memory (a) ANN structure and (b) with ROM Output	100
7.7	A pattern recognition system	102
7.8	The diagram of the implementation for the building element for Level 1	103
7.9	Perceptron structure with binary input / output	104
7.10	A digital learning mechanism design	105
7.11	The weight change range (a) general implementation (b) digital implementation	105
8.1	Basic CPL schematic structure	110
8.2	C ² PL layer connections (a) in serial and (b) in parallel	111
8.3	(a) 3-2 compressor logic structure and (b) its two cells	112
8.4	Two 3-2 compressors in (a) C ² PL(1) and (b) C ² PL(2)	113
8.5	The gate width ratio, $\zeta = W_{up} / W_{down}$	114
8.6	The gate width ratio, $\phi = W_p / W_n$	114
8.7	The output units (a) CMOS driver and (b) BiCMOS driver	115
8.8	Performance for the worst delay time	116
8.9	Performance for power dissipation	116
8.10	Performances with no buffer, CMOS & BiCMOS driver	117

8.11	The CPL 3-2 compressor	117
8.12	Simulated result for supply voltages	118
8.13	Simulated result for output loads	118
8.14	Layout results for full-adders without output driver	119
8.15	Two building blocks for (a) 4-2 compressor and (b) 7-3 compressor	120
8.16	Reduction process using 7-3 compressor to reduce 1010 row matrix	123
8.17	Reduction process using 4-2 compressor to reduce $M \times N$ matrix	124
9.1	A programmable analog neuron model	126
9.2	Schematics for comparing the effect of V_T perturbation	128
9.3	Synapse building block (a) schematic circuit and (b) its symbol representation	130
9.4	Neuron building block (a) schematic circuit and (b) its symbol representation	131
9.5	Connection network using the building blocks	132
9.6	Multilayer network implementation using BP algorithm	133
9.7	The simulation result of time delay caused by different number of weights	134
9.8	The simulation result of accuracy of binary-weighted current source	135
10.1	A basic operator subnet	141
10.2	Flow chart of the addition example	147
10.3	A ROM adder structure of the FRNN	149
10.4	The layout of the 3-bit cell	150
10.5	The structure of Type 1	151
10.6	The structure of Type 2	152
10.7	The diagram of a n -bit pipelined ripple-carry RNS adder	154
10.8	The diagram of a n -bit FRNN adder	155

INTRODUCTION

1.1 ANN FOR PATTERN RECOGNITION

Artificial Neural Networks (ANN) are massively parallel interconnected networks of simple (usually adaptive) nodes which are intended to interact with objects of the real world in the same way as biological nervous systems do [1].

The interest in these networks is due to the general opinion that they are able to perform some complicated and creative tasks, such as pattern recognition, similar to the way they are performed by human brains [2,10,35]. The implementations of these tasks by traditional computing methods have only reached relatively low performances in some limited aspects or environments. Nevertheless, as neural systems show some properties, like association, generalization, parallel searching, and adaptation to changes in the environment, which are analogous to human brain properties, they promise improved results.

The usage of ANNs for pattern recognition may be traced back to the perceptron models originated by Rosenblatt in 1950 [2]. The perceptron models used the concept of reward and punishment. In late 1960s, the progress in ANN models slowed down due to the limited capabilities of the early single layer perceptron models. In the mid-1970s and early 1980s, with the availability of enhanced computing power the progress in the development of ANN models accelerated. Researchers were able to model and test their theories about the functioning of the brain.

Today a number of well-developed theories and models of ANNs are available [3,34-36,40-47,55-64]. These networks consist of a large number of simple processing elements called nodes that represent the neurons. These nodes are interconnected by the synaptic connections. These models are capable of learning and making decisions; and are suitable for a variety of pattern recognition tasks [36,39].

Pattern recognition techniques can be grouped into two classes: supervised and unsupervised techniques. In supervised methods, certain number of samples are available for each category and these samples are used to train the classifier. In the case of unsupervised classification no training samples are available, and the network learns by detecting the similarity between the input patterns.

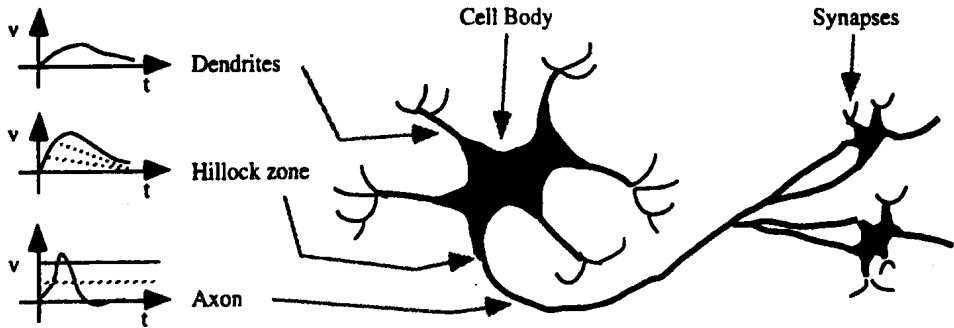


Fig. 1.1 A prototype biological neuron

Today many ANN models and algorithms for pattern recognition applications are available. They include Back-Propagation (BP) learning, Competitive learning, Kohonen learning, Adaptive Resonance Theory, Neocognitron models, Hopfield Networks, and Boltzman machines [5-13]. Applications of ANNs include character recognition, human face identification, speech recognition, multispectral image analysis, and expert systems [14-18]. The BP learning is essentially of the supervised type and the network learns with the help of training sets. The BP networks have been successfully used for many pattern recognition problems [5,15-17].

Another important class of neural networks is self-organizing neural networks. The networks with competitive learning algorithms are self-organizing networks. Early models of competitive learning were developed by Malsburg in his study of visual cortex [19]. Rumelhart and Zipser have suggested an algorithm for competitive learning [9]. The main disadvantage of competitive learning is that the network forgets its earlier learning with new learning and the network may get set into an unstable state with the spurious input patterns. To overcome this drawback, Grossberg developed an adaptive resonance architecture [10-11] and Fukushima proposed the Neocognitron models [12]. Kohonen developed a learning paradigm for self-organizing networks known as Kohonen learning [6-7]. These algorithms can be used for a variety of tasks in pattern recognition.

ANNs consist of parallel distributed processing (PDP) models. The PDP models are well described in the work of Rumelhart and McClelland [4]. The functional synthesis of these models consists of establishing a relationship between the several inputs and one or more outputs. In ANN, the nodes are connected to each other by the synaptic connections or the links. There is an associated synaptic strength or a weight with each connection. During the learning, the weights which represent the knowledge stored in the network are updated. The ANNs consist of two or several layers of nodes and each layer contains several nodes. The observed feature vector is presented to the input nodes. The input values may represent the probability that the discrete feature is present. Each possible decision or outcome can be represented by a node in the output layer.

1.2 NEURAL NETWORK MODEL

The basic element of neural networks of a brain is a neuron. The neurons consist of four basic parts: cell body, synapses, axons, and dendrites. The cell body essentially sums the membrane potential provided by the synapses. The synapses provide an output. Axons are the connections between the neurons that carry charge, and the dendrites are the branch-like structures which provide the sensory input to a cell body (See Fig.1.1). ANNs mimic the functioning of the neural networks of a brain. ANN consists of a large number of simple node. Each of the nodes is connected to another node(s) through a synaptic connection or a link [34].

Information processing takes place through the interaction between the nodes. Each node is associated with an activation value $\phi_j(t)$. The activation value passes through an activation function $f(\phi_j)$ to provide an actual output $y_j(t)$. These outputs pass through the unidirectional synaptic connections. There is an associated number, w_{ij} , called the weight or the connection strength, that determines the amount of effect node i can have on node j . For each node all the inputs are combined, and the total input, along with the current activation, determines the new activation value (Fig.1.2).

Usually ANNs consist of a number of layers and nodes in each layer. The most general model assumes the complete interconnections between all the nodes, and resolves the cases of the nonconnected nodes (i, j) by setting the weights $w_{ij} = 0$. A simple three-layer feedforward network is shown in Fig.1.3. The networks can be synchronous or asynchronous. The synchronous networks are controlled by clock pulses; whereas in asynchronous networks the nodes respond instantaneously to the incoming inputs.

The connections between the nodes can be bidirectional or unidirectional. The activation function and the activation values to be used in the network are often restricted in the range $[0, 1]$. In the case of discrete values, the activation values can take only two values – 0 or 1. The number of activation functions can be used to define the propagation law in the network. The commonly used activation functions are shown in Fig.1.4. For a sigmoid function the output at node j is

$$y_j = f(\phi_j) = \frac{1}{1 + \exp(-(\phi_j + \theta_j))} \quad (1.1)$$

where θ_j is a bias and the net input ϕ_j is represented as

$$\phi_j = \sum_{i=1}^n (x_i w_{ij}) \quad (1.2)$$