FUZZY NEURAL NETWORK THEORY AND APPLICATION
SERIES IN MACHINE PERCEPTION AND ARTIFICIAL INTELLIGENCE*

Editors:  
H. Bunke (Univ. Bern, Switzerland)  
P. S. P. Wang (Northeastern Univ., USA)

Vol. 43: Agent Engineering  

Vol. 44: Multispectral Image Processing and Pattern Recognition  
(Eds. J. Shen, P. S. P. Wang and T. Zhang)

Vol. 45: Hidden Markov Models: Applications in Computer Vision  
(Eds. H. Bunke and T. Caelli)

Vol. 46: Syntactic Pattern Recognition for Seismic Oil Exploration  
(K. Y. Huang)

Vol. 47: Hybrid Methods in Pattern Recognition  
(Eds. H. Bunke and A. Kandel)

Vol. 48: Multimodal Interface for Human-Machine Communications  
(Eds. P. C. Yuen, Y. Y. Tang and P. S. P. Wang)

Vol. 49: Neural Networks and Systolic Array Design  
(Eds. D. Zhang and S. K. Pal)

Vol. 50: Empirical Evaluation Methods in Computer Vision  
(Eds. H. I. Christensen and P. J. Phillips)

Vol. 51: Automatic Diatom Identification  
(Eds. H. du Buf and M. M. Bayer)

Vol. 52: Advances in Image Processing and Understanding  
A Festschrift for Thomas S. Huwang  
(Eds. A. C. Bovik, C. W. Chen and D. Goldgof)

Vol. 53: Soft Computing Approach to Pattern Recognition and Image Processing  
(Eds. A. Ghosh and S. K. Pal)

Vol. 54: Fundamentals of Robotics — Linking Perception to Action  
(M. Xie)

Vol. 55: Web Document Analysis: Challenges and Opportunities  
(Eds. A. Antonacopoulos and J. Hu)

Vol. 56: Artificial Intelligence Methods in Software Testing  
(Eds. M. Last, A. Kandel and H. Bunke)

Vol. 57: Data Mining in Time Series Databases  
(Eds. M. Last, A. Kandel and H. Bunke)

Vol. 58: Computational Web Intelligence: Intelligent Technology for Web Applications  
(Eds. Y. Zhang, A. Kandel, T. Y. Lin and Y. Yao)

Vol. 59: Fuzzy Neural Network Theory and Application  
(P. Liu and H. Li)

*For the complete list of titles in this series, please write to the Publisher.
FUZZY NEURAL NETWORK THEORY AND APPLICATION

Puyin Liu  
National University of Defense Technology  
Changsha, China

Hongxing Li  
Beijing Normal University  
Beijing, China
# Table of Contents

## Foreword ............................................................ ( xi )
## Preface .............................................................. ( xv )

### Chapter I Introduction ............................................. ( 1 )
   §1.1 Classification of fuzzy neural networks ......................... ( 1 )
   §1.2 Fuzzy neural networks with fuzzy operators ................... ( 2 )
   §1.3 Fuzzified neural networks ....................................... ( 4 )
      1.3.1 Learning algorithm for regular FNN’s ...................... ( 4 )
      1.3.2 Universal approximation of regular FNN’s ................ ( 6 )
   §1.4 Fuzzy systems and fuzzy inference networks .................. ( 7 )
      1.4.1 Fuzzy systems ............................................... ( 8 )
      1.4.2 Fuzzy inference networks ................................. ( 11 )
   §1.5 Fuzzy techniques in image restoration ......................... ( 12 )
      1.5.1 Crisp nonlinear filters .................................... ( 12 )
      1.5.2 Fuzzy filters ............................................... ( 13 )
   §1.6 Notations and preliminaries ................................... ( 14 )
   §1.7 Outline of the topics of the chapters ......................... ( 17 )
   References .......................................................... ( 21 )

### Chapter II Fuzzy Neural Networks for Storing and Classifying  (25 )
   §2.1 Two layer max–min fuzzy associative memory ................ ( 26 )
      2.1.1 FAM with threshold ......................................... ( 30 )
      2.1.2 Simulation example ......................................... ( 35 )
   §2.2 Fuzzy δ–learning algorithm .................................... ( 36 )
      2.2.1 FAM’s based on ‘∨ ~ ∧’ .................................... ( 36 )
      2.2.2 FAM’s based on ‘∨ ~ *’ .................................... ( 39 )
   §2.3 BP learning algorithm of FAM’s ............................... ( 45 )
      2.3.1 Two analytic functions ..................................... ( 45 )
2.3.2 BP learning algorithm .................................. (48)

§2.4 Fuzzy ART and fuzzy ARTMAP .......................... (50)
2.4.1 ART1 architecture .................................. (50)
2.4.2 Fuzzy ART ........................................... (53)
2.4.3 Fuzzy ARTMAP ...................................... (60)
2.4.4 Real examples ........................................ (62)
References .................................................. (65)

Chapter III Fuzzy Associative Memory—Feedback Networks (68)

§3.1 Fuzzy Hopfield networks .................................. (68)
3.1.1 Attractor and attractive basin .......................... (69)
3.1.2 Learning algorithm based on fault–tolerance ........ (73)
3.1.3 Simulation example .................................. (76)
§3.2 Fuzzy Hopfield network with threshold .................. (78)
3.2.1 Attractor and stability ................................ (79)
3.2.2 Analysis of fault–tolerance .......................... (83)
§3.3 Stability and fault–tolerance of FBAM ..................... (88)
3.3.1 Stability analysis .................................... (88)
3.3.2 Fault–tolerance analysis ............................... (91)
3.3.3 A simulation example ................................. (94)
§3.4 Learning algorithm for FBAM ............................... (96)
3.4.1 Learning algorithm based on fault–tolerance .......... (96)
3.4.2 A simulation example ................................ (98)
3.4.3 Optimal fault–tolerance ............................... (100)
3.4.4 An example ........................................... (105)
§3.5 Connection network of FBAM ............................... (106)
3.5.1 Fuzzy row–restricted matrix ........................... (106)
3.5.2 The connection relations of attractors ................. (107)
3.5.3 The elementary memory of \((R, L)\) ................ (111)
3.5.4 The transition laws of states .......................... (114)
§3.6 Equilibrium analysis of fuzzy Hopfield network .......... (117)
3.6.1 Connection relations of attractors ..................... (118)
3.6.2 Elementary memory of \(W\) ............................ (122)
3.6.3 The state transition laws .............................. (124)
Chapter IV Regular Fuzzy Neural Networks

§4.1 Regular fuzzy neuron and regular FNN
4.1.1 Regular fuzzy neuron
4.1.2 Regular fuzzy neural network
4.1.3 A counter example of universal approximation
4.1.4 An example of universal approximation

§4.2 Learning algorithms
4.2.1 Preliminaries
4.2.2 Calculus of $\vee - \wedge$ functions
4.2.3 Error function
4.2.4 Partial derivatives of error function
4.2.5 Learning algorithm and simulation

§4.3 Conjugate gradient algorithm for fuzzy weights
4.3.1 Fuzzy CG algorithm and convergence
4.3.2 GA for finding optimal learning constant
4.3.3 Simulation examples

§4.4 Universal approximation to fuzzy valued functions
4.4.1 Fuzzy valued Bernstein polynomial
4.4.2 Four-layer regular feedforward FNN
4.4.2 An example

§4.5 Approximation analysis of regular FNN
4.5.1 Closure fuzzy mapping
4.5.2 Learning algorithm

§4.6 Approximation of regular FNN with integral norm
4.6.1 Integrable bounded fuzzy valued functions
4.6.2 Universal approximation with integral norm

Chapter V Polygonal Fuzzy Neural Networks

§5.1 Uniformity analysis of feedforward networks
5.1.1 Uniform approximation of four-layer network
5.1.2 Uniformity analysis of three-layer neural network
§5.2 Symmetric polygonal fuzzy number .................................. (211)
  5.2.1 Symmetric polygonal fuzzy number space ................. (212)
  5.2.2 Polygonal linear operator ....................................... (217)
  5.2.3 Extension operations based on polygonal fuzzy numbers (220)
§5.3 Polygonal FNN and learning algorithm ............................ (223)
  5.3.1 Three–layer feedforward polygonal FNN ................. (223)
  5.3.2 Learning algorithm ................................................ (226)
  5.3.3 A simulation .......................................................... (231)
§5.4 Approximation of polygonal FNN ...................................... (233)
  5.4.1 I/O relationship analysis of polygonal FNN ............. (234)
  5.4.2 Approximation of polygonal FNN ............................... (240)
References ........................................................................ (247)

Chapter VI Approximation Analysis of Fuzzy Systems ....... (251)
§6.1 Piecewise linear function .................................................. (252)
  6.1.1 SPLF and its properties ............................................ (252)
  6.1.2 Approximation of SPLF’s ........................................... (254)
§6.2 Approximation of generalized fuzzy systems with integral norm (259)
  6.2.1 Generalized Mamdani fuzzy system ....................... (261)
  6.2.2 Generalized T–S fuzzy system ................................. (268)
§6.3 Hierarchical system of generalized T–S fuzzy system ....... (273)
  6.3.1 Hierarchical fuzzy system ......................................... (274)
  6.3.2 Generalized hierarchical T–S fuzzy system ............. (276)
§6.4 Approximation of hierarchical T–S fuzzy system .......... (282)
  6.4.1 Universal approximation with maximum norm .......... (282)
  6.4.2 Realization procedure of universal approximation ...... (285)
  6.4.3 Universal approximation with integral norm .......... (288)
References ........................................................................ (290)

Chapter VII Stochastic Fuzzy Systems and Approximations (296)
§7.1 Stochastic process and stochastic integral ....................... (296)
  7.1.1 Stochastic measure and stochastic integral .......... (297)
  7.1.2 Canonical representation of Brownian motion ...... (299)
§7.2 Stochastic fuzzy systems .................................................. (301)
7.2.1 Stochastic T–S system ........................................ (301)
7.2.2 Stochastic Mamdani fuzzy system .......................... (304)
§7.3 Universal approximation of stochastic process by T–S system (307)
7.3.1 Uniform approximation .................................... (307)
7.3.2 Approximation with mean square sense .................. (309)
§7.4 Universal approximation of stochastic Mamdani fuzzy system (315)
7.4.1 Approximation of stochastic Mamdani fuzzy system ... (315)
7.4.2 Example ...................................................... (318)
References ......................................................... (321)

Chapter VIII  Application of FNN to Image Restoration .... (326)
§8.1 Generalized fuzzy inference network .......................... (327)
8.1.1 Generalized defuzzification operator ....................... (327)
8.1.2 Fuzzy inference network .................................. (332)
8.1.3 Universal approximation of generalized FINN .......... (335)
8.1.4 Simulation of system identification ....................... (338)
§8.2 Representation of two-dimensional image by FINN ........ (343)
8.2.1 Local FNN representation of image ....................... (343)
8.2.2 Optimal FNN filter ....................................... (348)
8.2.3 Experiment results ....................................... (350)
§8.3 Image restoration based on FINN ............................. (353)
8.3.1 Fuzzy partition ........................................... (354)
8.3.2 Selection type FNN and its universal approximation .. (357)
8.3.3 A novel FNN filter ....................................... (361)
8.3.4 Learning algorithm ....................................... (362)
8.3.5 Simulation examples ..................................... (363)
References ......................................................... (366)

Indices ............................................................. (371)
§I.1 Symbols ....................................................... (371)
§I.2 Terminologies ................................................ (373)
This page is intentionally left blank
Authoring Professors P. Liu and H. Li, "Fuzzy Neural Network Theory and Application," or FNTTA for short, is a highly important work. Essentially, FNTTA is a treatise that deals authoritatively and in depth with the basic issues and problems that arise in the conception, design and utilization of fuzzy neural networks. Much of the theory developed in FNTTA goes considerably beyond what can be found in the literature.

Fuzzy neural networks, or more or less equivalently, neurofuzzy systems, as they are frequently called, have a long history. The embryo was a paper on fuzzy neurons by my former student, Ed Lee, which was published in 1975. Thereafter, there was little activity until 1988, when H. Takagi and I. Hayashi obtained a basic patent in Japan, assigned to Matsushita, which described systems in which techniques drawn from fuzzy logic and neural networks were employed in combination to achieve superior performance.

The pioneering work of Takagi and Hayashi opened the door to development of a wide variety of neurofuzzy systems. Today, there is an extensive literature and a broad spectrum of applications, especially in the realm of consumer products.

A question which arises is: Why is there so much interest and activity in the realm of neurofuzzy systems? What is it that neurofuzzy systems can do that cannot be done equally well by other types of systems? To understand the underlying issues, it is helpful to view neurofuzzy systems in a broader perspective, namely, in the context of soft computing.

What is soft computing? In science, as in many other realms of human activity, there is a tendency to be nationalistic-to commit oneself to a particular methodology and employ it exclusively. A case in point is the well-known Hammer Principle: When the only tool you have is a hammer, everything looks like a nail. Another version is what I call the Vodka Principle: No matter what your problem is, vodka will solve it.

What is quite obvious is that if A, B, ..., N are complementary methodologies, then much can be gained by forming a coalition of A, B, ..., N. In this perspective, soft computing is a coalition of methodologies which are tolerant of imprecision, uncertainty and partial truth, and which collectively provide a foundation for conception, design and utilization of intelligent systems. The principal numbers of the coalition are: fuzzy logic, neurocomputing, evolutionary computing, probabilistic computing, rough set theory, chaotic computing and machine learning. A basic credo which underlies soft computing is that, in
general, better results can be obtained by employing the constituent methodologies of soft computing in combination rather in a stand-alone mode.

In this broader perspective, neurofuzzy systems may be viewed as the domain of a synergistic combination of neurocomputing and fuzzy logic; inheriting from neurocomputing the concepts and techniques related to learning and approximation, and inheriting from fuzzy logic the concepts and techniques related to granulation, linguistic variable, fuzzy if-then rules and rules of inference and constraint propagation.

An important type of neurofuzzy system which was pioneered by Arabshahi et al starts with a neuro-based algorithm such as the backpropagation algorithm, and improves its performance by employing fuzzy if-then rules for adaptive adjustment of parameters. What should be noted is that the basic idea underlying this approach is applicable to any type of algorithm in which human expertise plays an essential role in choosing parameter-values and controlling their variation as a function of performance. In such applications, fuzzy if-then rules are employed as a language for describing human expertise.

Another important direction which emerged in the early nineties involves viewing a Takaga-Sugeno fuzzy inference system as a multilayer network which is similar to a multilayer neural network. Parameter adjustment in such systems is achieved through the use of gradient techniques which are very similar to those associated with backpropagation. A prominent example is the ANFIS system developed by Roger Jang, a student of mine who conceived ANFIS as a part of his doctoral dissertation at UC Berkeley. The widely used method of radical basis functions falls into the same category.

Still another important direction—a direction initiated by G. Bortolan—involves a fuzzification of a multilayer, feedforward neural network, resulting in a fuzzy neural network, FNN. It is this direction that is the principal concern of the work of Professors Liu and Li.

Much of the material in FNNTA is original with the authors and reflects their extensive experience. The coverage is both broad and deep, extending from the basics of FNN and FAM (fuzzy associate memories) to approximation theory of fuzzy systems, stochastic fuzzy systems and application to image restoration. What is particularly worthy of note is the author’s treatment of universal approximation of fuzzy-valued functions.

A basic issue that has a position of centrality in fuzzy neural network theory—and is treated as such by the authors—is that of approximation and, in particular, universal approximation. Clearly, universal approximation is an issue that is of great theoretical interest. A question which arises is: Does the theory of universal approximation come to grips with problems which arise in the design of fuzzy neural networks in realistic settings? I believe that this issue is in need of further exploration. In particular, my feeling is that the usual assumption about continuity of the function that is approximated, is too weak, and that the problem of approximation of functions which are smooth, rather than continuous, with smoothness defined as a fuzzy characteristic, that
is, a matter of degree, must be addressed.

FNNTA is not intended for a casual reader. It is a deep work which addresses complex issues and aims at definitive answers. It ventures into territories which have not been explored, and lays the groundwork for new and important applications. Professors Liu and Li, and the publisher, deserve our thanks and congratulations for producing a work that is an important contribution not just to the theory of fuzzy neural networks, but, more broadly, to the conception and design of intelligent systems.

Lotfi A. Zadeh
Professor in the Graduate School, Computer Science Division
Department of Electrical Engineering and Computer Sciences
University of California
Berkeley, CA 94720–1776
Director, Berkeley Initiative in Soft Computing (BISC)

March, 2004
As a hybrid intelligent system of soft computing technique, the fuzzy neural network (FNN) is an efficient tool to deal with nonlinearly complicated systems, in which there are linguistic information and data information, simultaneously. In view of two basic problems—learning algorithm and universal approximation, FNN's are thoroughly and systematically studied in the book. The achievements obtained here are applied successfully to pattern recognition, system modeling and identification, system forecasting, and digital image restoration and so on. Many efficient methods and techniques to treat these practical problems are developed.

As two main research objects, learning algorithms and universal approximations of FNN's constitute the central part of the book. The basic tools to study learning algorithms are the max–min (\(\lor - \land\)) functions, the cuts of fuzzy sets and interval arithmetic, etc. And the bridges to research universal approximations of fuzzified neural networks and fuzzy inference type networks, such as regular FNN's, polygonal FNN's, generalized fuzzy systems and generalized fuzzy inference networks and so on are the fuzzy valued Bernstein polynomial, the improved type extension principle and the piecewise linear functions. The achievements of the book will provide us with the necessary theoretic basis for soft computing technique and the applications of FNN's.

There have been a few of books and monographs on the subject of FNN's or neuro-fuzzy systems. There are several distinctive aspects which together make this book unique.

First of all, the book is a thorough summation and deepening of authors' works in recent years in the fields related. So the readers can get latest information, including latest research surveys and references related to the subjects through this book. This book treats FNN models both from mathematical perspective with the details of most proofs of the results included: only simple and obvious proofs are omitted, and from applied or computational aspects with the realization steps of main results shown, also many application examples included. So it is helpful for readers who are interested in mathematical aspects of FNN's, also useful for those who do not concern themselves with the details of the proofs but the applied aspects of FNN's.

Second, the perspective of the book is centered on two typical problems on FNN's, they are universal approximation and learning algorithm. The achievements about universal approximation of FNN's may provide us with the theoretic basis for FNN applications in many real fields, such as system modeling
and system identification, information processing and system optimization and so on. And learning algorithms for FNN's may lead to rational treatments of FNN's for their architectures, implementation procedures and all kinds practical applications, etc. So readers may easily enter through this book the fields related by taking the two subjects as leads. Also the book includes many well-designed simulation examples for readers' convenience to understand the results related.

Third, the arrangement of contents of the book is novel and there are few overlaps with other books related to the field. Many concepts are first introduced for approximation and learning of FNN's. The constructive proofs of universal approximations provide us with much convenience in modeling or identifying a real system by FNN's. Also they are useful to build some learning algorithms to optimize FNN architectures.

Finally almost all common FNN models are included in the book, and as many as possible references related are listed in the end of each chapter. So readers may easily find their respective contents that they are interested. Moreover, those FNN models and references make this book valuable to people interested in various FNN models and applications.


Now let us sketch out the main points of the book, and the details will be presented in Chapter I. This book consists of four primary parts: the first part focuses mainly on FNN's based on fuzzy operators 'V' and 'A', including FNN's for storing and classifying fuzzy patterns, dynamical FNN's taking fuzzy Hopfield networks and fuzzy bidirectional associative memory (FBAM) as typical models. They are dealt with in Chapter II and Chapter III, respectively. The second part is mainly contributed to the research of universal approximations of fuzzified neural networks and their learning algorithms. The fuzzified neural networks mean mainly two classes of FNN's, i.e. regular FNN's and polygonal FNN's. A series of equivalent conditions that guarantee universal approximations are built, and several learning algorithms for fuzzy weights are developed. Also implementations and applications of fuzzified neural networks are included. The third part focuses on the research of universal approximations, including ones of generalized fuzzy systems to integrable functions and ones of stochastic fuzzy systems to some stochastic processes. Also the learning algorithms for the stochastic fuzzy systems are studied. The fourth part is contributed to the applications of the achievements and methodology on FNN's
to digital image restoration. A FNN representation of digital images is built for reconstructing images and filtering noises. Based on fuzzy inference networks some efficient FNN filters are developed for removing impulse noise and restoring images.

When referring to a theorem, a lemma, a corollary, a definition, etc in the same chapter, we utilize the respective numbers as they appear in the statements, respectively. For example, Theorem 4.2 means the second theorem in Chapter IV, while Definition 2.4 indicates the fourth definition in Chapter II, and so on.

Although we have tried very hard to give references to original papers, there are many researchers working on FNN’s and we are not always aware of contributions by various authors, to which we should give credit. We have to say sorry for our omissions. However we think the references that we have listed are helpful for readers to find the related works in the literatures.

We are indebted to Professor Lotfi A. Zadeh of University of California, Berkeley who writes the preface of the book in the midst of pressing affairs at authors’ invitation. We are specially grateful to Professors Guo Guirong and He Xingui who read the book carefully and make their many of insightful comments. Thanks are also due to Professor Bunke Horst who accepts this book in the new book series edited by him. Finally we express our indebtedness to Dr. Seldrup Ian the editor of the book and the staff at the World Scientific Publishing for displaying a lot of patience in our final cooperation.

We were supported by several National Natural Science Foundation Grants of China (e.g., No.69974041, No.60375023 and No.60174013) during the years this book was written.

Puyin Liu and Hongxing Li

March 2004
CHAPTER I
Introduction

As information techniques including their theory and applications develop further, the studying objects related have become highly nonlinear and complicated systems, in which natural linguistic information and data information coexist [40]. In practice, a biological control mechanism can carry out complex tasks without having to develop some mathematical models, and without solving any complex integral, differential or any other types of mathematical equations. However, it is extremely difficult to make an artificial mobile robot to perform the same tasks with vague and imprecise information for the robot involves a fusion of most existing control techniques, such as adaptive control, knowledge-based engineering, fuzzy logic and neural computation and so on. To simulate biological control mechanisms, efficiently and to understand biological computational power, thoroughly a few of powerful fields in modern technology have recently emerged [30, 61]. Those techniques take their source at Zadeh’s soft data analysis, fuzzy logic and neural networks together with genetic algorithm and probabilistic reasoning [68–71]. The soft computing techniques can provide us with an efficient computation tool to deal with the highly nonlinear and complicated systems [67]. As a collection of methodologies, such as fuzzy logic, neural computing, probabilistic reasoning and genetic algorithm (GA) and so on, soft computing is to exploit the tolerance for imprecision, uncertainty and partly truth to achieve tractability, robustness and low solution cost. In the partnership of fuzzy logic, neural computing and probabilistic reasoning, fuzzy logic is mainly concerned with imprecision and approximate reasoning, neural computing with learning and curve fitting, and probabilistic reasoning with uncertainty and belief propagation.

§1.1 Classification of fuzzy neural networks

As a main ingredient of soft computing, fuzzy neural network (FNN) is a hybrid intelligent system that possess the capabilities of adjusting adaptively and intelligent information processing. In [36, 37] Lee S. C. and Lee E. T. firstly proposed the fuzzy neurons and some systematic results on FNN’s were developed by softening the McCulloch–Pitts neurons in the middle 1970s when the interest in neural networks faltered. So such novel neural systems had not attracted any attention until 1987 when Kosko B. developed a fuzzy associa-
tive memory (FAM) to deal with intelligent information by introducing some fuzzy operators in associative memory networks [32]. Since the early 1980s the research on neural networks has increased dramatically because of the works done by Hopfield J. J. (see [26]). The FNN models have also attracted many scholars’ attention. A lot of new new concepts, such as innovative architecture and training rules and models about FNN’s have been developed [30, 32, 61]. In practice FNN’s have found useful in many application fields, for instance, system modelling [16, 24], system reliability analysis [7, 42], pattern recognition [33, 56], and knowledge engineering and so on. Based on fuzziness involved in FNN’s developed since the late of 1980s, one may broadly classify all FNN models as three main types as shown in Figure 1.1:

\[
\begin{align*}
\text{FNN} & \quad \{ \\
\text{Based on fuzzy operators} & \quad \text{Feedforward neural networks (1980’s),} \\
\text{Fuzzified neural networks} & \quad \text{Feedback neural networks (1990’s),} \\
\text{Fuzzy inference networks} & \quad \text{Regular FNN’s (1990’s),} \\
& \quad \text{Improved FNN’s (1990’s),} \\
& \quad \text{Mamdani type (1990’s),} \\
& \quad \text{Takagi–Sugeno type (1990’s),} \\
& \quad \text{Generalized type (1990’s).}
\end{align*}
\]

Figure 1.1 Classification of FNN’s

§1.2 Fuzzy neural networks with fuzzy operators

FNN’s based on fuzzy operators are firstly studied by Lee and Lee in 1970s. Such FNN’s have become one of the foci in neural network research since Kosko introduced the fuzzy operators ‘\( \lor \)’ and ‘\( \land \)’ in associative memory to define fuzzy associative memory (FAM) in 1987. A FAM is a feedforward FNN whose information flows from input layer to output layer. It possesses the capability of storing and recalling fuzzy information or fuzzy patterns [32, 41]. So storage capability and fault-tolerance are two main problems we focus on in the research on FAM’s. In practice a applicable FAM should possess strong storage capability, i.e. as many fuzzy patterns as possible can be stored in a FAM. So far many learning algorithms, including the fuzzy Hebbian rule, the fuzzy delta rule and the fuzzy back propagation (BP) algorithm and so on have been developed to train FAM’s and to improve storage capability of a FAM [32]. Equally, fault-tolerance, i.e. the capability of a FAM to recall the right fuzzy pattern from a distorted input is of real importance, for in practice many fuzzy patterns to handle are inherently imprecise and distorted. The research on feedforward FAM’s, including their topological architecture designs, the selection of fuzzy operators for defining the internal operations, learning algorithms and so on attracts much attention [41]. The achievements related to FAM’s
have found useful in many applied areas, such as pattern recognition [33, 59], pattern classification [41, 58], system analysis [38], signal processing [27] and so on.

A main object of the research on FAM's is to improve the storage capability, which relates closely to fuzzy relational equation theory. Blanco et al in [4, 5] express a fuzzy system as a fuzzy relational equation, thus, the feedforward FAM's can identify a fuzzy relation by designing suitable learning algorithm. On the other hand, many methods for solving fuzzy relational equations are employed to improve the storage capability of FAM's. Li and Ruan in [38] use FAM's based on several fuzzy operator pairs including '∨ − ∧', '∨ − x', '+ − x' and '+ − ∧' etc to identify many fuzzy system classes by building a few of novel learning algorithms. Moreover, they show the convergence of fuzzy delta type iteration algorithms. Liu et al utilize the approaches for solving fuzzy relational equations to build a series of equivalent conditions that a given fuzzy pattern family can be stored by a FAM [41]. Furthermore, some learning algorithms for improving storage capability of FAM's are developed. These constitute one of the main parts in Chapter II. Pedrycz in [51] put forward two logic type fuzzy neurons based on general fuzzy operators, and such FAM's can be applied to realize fuzzy logic relations efficiently.

Adaptive resonance theory (ART) is an efficient neural model of human cognitive information processing. It has since led to an evolving series of real-time neural network models for unsupervised category learning and pattern recognition. The model families include ART1, which can process patterns expressed as vectors whose components are either 0 or 1 [8]; ART2 which can categorize either analog or binary input patterns [9], and ART3 which can carry out parallel search, or hypothesis testing, of distributed recognition codes in a multi-level network hierarchy [10]. The fuzzy ART model developed by Carpenter et al in [11] generalizes ART1 to be capable of learning stable recognition categories in response to both analog and binary input patterns. The fuzzy operators '∨' and '∧' are employed to define the operations between fuzzy patterns. The research related to fuzzy ART is mainly focused on the classification characteristics and applications to pattern recognition in all kinds of applied fields, which is presented in Chapter II.

Another important case of FNN's based on fuzzy operators is one of feedback FNN's. It imitates human brain in understanding objective world, which is a procedure of self-improving again and again. A feedback neural network as a dynamical system finishes information processing by iterating repeatedly from an initial input to the equilibrium state. In practice, an equilibrium state of a dynamical FNN turns out to be the right fuzzy pattern to recall. So the recalling procedure of a dynamical FNN is in nature a process that the FNN evolutes to its equilibrium state from an initial fuzzy pattern. In the book we focus mainly on two classes of dynamical FNN's, they are fuzzy Hopfield networks and fuzzy bidirectional associative memory (FBAM). By the comparison between the two dynamical FNN's and the corresponding crisp neural networks
we can see

1. The FNN's do not need the transfer function used in the crisp networks, for a main function of the transfer function in artificial neural networks lies in controlling output range, which may achieve by the fuzzy operators '∨', '∧', and '∧' is a threshold function [4, 5].

2. In practice, it is much insufficient to represent fuzzy information by the strings only consisting of 0, 1. We should utilize fuzzy patterns whose components belong to [0, 1] to describe fuzzy information. So the dynamical FNN's may be applied much more widely than the crisp networks may.

Similarly with crisp dynamical networks, in the research related to the dynamical FNN's, the stability analysis, including the global stability of dynamical systems and Lyapounov stability of the equilibrium state (attractor), the attractive basins of attractors and the discrimination of attractors and pseudo-attractors and so on are main subjects to study. Those problems will be studied thoroughly in Chapter III.

§1.3 Fuzzified neural networks

A fuzzified neural network means such a FNN whose inputs, outputs and connection weights are all fuzzy set, which is also viewed as a pure fuzzy system [61]. Through the internal relationships among fuzzy sets of a fuzzified neural network, a fuzzy input can determines a fuzzy output. One most important class of fuzzified neural networks is regular FNN class, each of which is the fuzzifications of a crisp feedforward neural network. So for a regular FNN, the topological architecture is identical to one of the corresponding crisp neural network, and the internal operations are based on Zadeh's extension principle [44] and fuzzy arithmetic [20]. Since regular FNN's were put forward by Buckley et al [6] and Ishibuchi et al [28] about in 1990s, the systematic achievements related have been built by focusing mainly on two basic problem—learning algorithm and universal approximation.

1.3.1 Learning algorithm for regular FNN's

There are two main approaches to design learning algorithms for regular FNN's, they are α—cut learning [28, 41], and the genetic algorithm (GA) for fuzzy weights [2]. The main ideas for the α—cut learning algorithm rest with the fact that for any \( \alpha \in [0, 1] \), we utilize the BP algorithm for crisp neural networks to determine the two endpoints of α—cut and consequently establish the α—cut of a fuzzy weight, and then define the fuzzy weight. Thus, the fuzzy connection weights of the regular FNN is trained suitably. However, the α—cut learning algorithm loses its effectiveness frequently since for \( \alpha_1, \alpha_2 \in [0, 1] : \alpha_1 < \alpha_2 \), by the BP algorithm we obtain the α—cuts \( \tilde{W}_{\alpha_1}, \tilde{W}_{\alpha_2} \), and if no constraint is added, the fact \( \tilde{W}_{\alpha_2} \subseteq \tilde{W}_{\alpha_1} \) can not be guaranteed. And therefore the fuzzy set \( \tilde{W} \) can not be defined. So in order to ensure the effectiveness of the algorithm
it is necessary to solve the following optimization problem:

\[
\begin{align*}
\min \{ E(\tilde{W}_1, ..., \tilde{W}_n) \mid \tilde{W}_1, ..., \tilde{W}_n \text{ are fuzzy sets} \}, \\
\text{s.t. } \forall \alpha_1, \alpha_2 \in [0, 1] : \alpha_1 < \alpha_2, \forall i \in \{1, ..., n\}, \ (\tilde{W}_i)_{\alpha_2} \subseteq (\tilde{W}_i)_{\alpha_1},
\end{align*}
\]

(1.1)

where \( E(\cdot) \) is an error function. If no constraint is added, (1.1) is generally insolvable. Even if we may find a solution of (1.1) in some special cases the corresponding solving procedure will be extremely complicated. Another difficulty to hinder the realization of the \( \alpha \)-cut learning algorithm is to define a suitable error function \( E(\cdot) \) [41], so that not only its minimization can ensure to realize the given input—output (I/O) relationship, approximately, but also its derivatives related are easy to calculate. To avoid solve (1.1), a common method to define \( E(\cdot) \) is to introduce some constraints on the fuzzy connection weights, for instance, we may choose the fuzzy weights as some common fuzzy numbers such as triangular fuzzy numbers, trapezoidal fuzzy numbers and Gaussian type fuzzy numbers and so on, which can be determined by a few of adjustable parameters. Ishibuchi et al utilize the triangular or, trapezoidal fuzzy numbers to develop some \( \alpha \)-cut learning algorithms for training the fuzzy weights of regular FNN's. And some successful applications of regular FNN's in the approximate realization of fuzzy inference rules are demonstrated in [28]. Park et al in [50] study the inverse procedure of the learning for a regular FNN, systematically. That is, using the desired fuzzy outputs of the FNN to establish conversely the conditions for the corresponding fuzzy inputs. This is a fuzzy version of the corresponding problem for crisp neural networks [39]. Solution of such a problem rest in nature with treating the \( \alpha \)-cut learning.

However, no matter how different fuzzy weights and error functions these learning algorithms have, two important operations 'V' and 'A' are often involved. An indispensable step to construct the fuzzy BP algorithm is to differentiate \( V - A \) operations by using the unit step function, that is, for the given real constant \( a \), let

\[
\frac{\partial (x \lor a)}{\partial x} = \begin{cases} 
1, & x \geq a, \\
0, & x < a;
\end{cases} \quad \frac{\partial (x \land a)}{\partial x} = \begin{cases} 
1, & x \leq a, \\
0, & x > a.
\end{cases}
\]

(1.2)

Above representations are only valid for special case \( x \neq a \). And if \( x = a \), they are no longer valid. Based on these two derivative formulas, the chain rules for differentiation of composition functions are only in form, and lack rigorous mathematical sense. Apply the results in [73] to analyze the \( V - A \) operations fully and to develop a rigorous theory for the calculus of \( V \) and \( A \) operations are two subsidiary results in Chapter IV.

The GA's for fuzzy weights are also developed for such fuzzy sets that they can be determined uniquely by a few of adjustable parameters when it is possible to code fuzzy weights and to ensure one to one correspondence between a code sequence in GA and fuzzy connection weights. For instance, Aliev et
al in [2] employ simple GA to train the triangular fuzzy number weights and biases of regular FNN’s. They encode all fuzzy weights as a binary string (chromosome) to complete the search process. The transfer function $\sigma$ related is assumed to be an increasing real function. The research in this field is at its infancy and many fundamental problems, such as, how to define a suitable error function? what more efficient code techniques can be employed and what are the more efficient genetic strategies? and so on remain to be solved.

Regardless of $\alpha$-cut learning algorithm and GA for the fuzzy weights of regular FNN’s they are efficient only for a few of special fuzzy numbers, such as triangular or trapezoidal fuzzy numbers, Gaussian type fuzzy numbers and so on. The applications of the learning algorithms are much restricted. And therefore it is meaningful and important to develop the BP type learning algorithms or, GA for fuzzy weights of regular FNN’s within a general framework, that is, we have to build learning algorithms for general fuzzy weights. The subject constitutes one of central parts of Chapter IV. To speed the convergence of the fuzzy BP algorithm we develop a fuzzy conjugate gradient (CG) algorithm [18] to train a regular FNN with general fuzzy weights.

1.3.2 Universal approximation of regular FNN’s

Another basic problem for regular FNN’s is the universal approximation, which can provide us with the theoretic basis for the FNN applications. The universal approximation of crisp feedforward neural networks means such a fact that for any compact set $U$ of the input space and any continuous function $f$ defined on the input space, $f$ can be represented with arbitrarily given degree of accuracy $\varepsilon > 0$ by a feedforward crisp neural network. The research related has attracted many scholars since the late 1980s. It is shown that a three-layer feedforward neural network with a given nonlinear activation function in the hidden layer is capable of approximating generic class of functions, including continuous and integrable ones [13, 14, 57]. Recently Scarselli and Tsoi [57] present a detail survey of recent works on the approximation by feedforward neural networks, and obtain some new results by studying the computational aspects and training algorithms for the approximation problem. The approximate representation of a continuous function by a three layer feedforward network can with the approximate sense solve the 13-th Hilbert problem with a simple approach [57], and Kolmogorov had to employ a complicated approach to solve the problem analytically in 1950s [57]. The achievements related to the field have not only solved the approximation representation of some multivariate functions by the combination of finite compositions of one-variable functions, but also found useful in many real fields, such as the approximation of structural synthesis [57], system identification [14], pattern classification [25], and adaptive filtering [52], etc.

Since the middle 1990s many authors have begun to paid their attentions to the similar approximation problems in fuzzy environment [6, 22, 28, 41]. Firstly Buckley et al in [6] study systematically the universal approximation
of FNN's and obtain such a fact: Hybrid FNN's can be approximator to fuzzy functions while regular FNN's are not capable of approximating continuous fuzzy functions to any degree of accuracy on the compact sets of a general fuzzy number space. Considering the arbitrariness of a hybrid FNN in its architectures and internal operations, we find such a FNN is inconvenient for realizations and applications. Corresponding to different practical problems the respective hybrid FNN's with different topological architectures and internal operations have to be constructed [6]. However, regular FNN's, whose topological architectures are identical to the corresponding crisp ones, internal operations are based on extension principle and fuzzy arithmetic, have found convenient and useful in many applications. Thus some important questions arise. What are the conditions for continuous fuzzy functions that can be arbitrarily closely approximated by regular FNN's? that is, which function class can guarantee universal approximation of regular FNN's to hold? Whether the corresponding equivalent conditions can be established? Since the inputs and outputs of regular FNN's are fuzzy sets, the common operation laws do not hold any more. It is difficult to employ similar approaches for dealing with crisp feedforward neural networks to solve above problems for regular FNN's.

Above problems attract many scholars' attention. At first Buckley and Hayashi [6] show a necessary condition for the fuzzy functions that can be approximated arbitrarily closely by regular FNN's, that is, the fuzzy functions are increasing. Then Feuring et al in [22] restrict the inputs of regular FNN's as trapezoidal fuzzy numbers, and build the approximate representations of a class of trapezoidal fuzzy functions by regular FNN's. Also they employ the approximation of the regular FNN's with trapezoidal fuzzy number inputs and connection weights to solve the overfitting problem. We establish some sufficient conditions for fuzzy valued functions defined on an interval \([0, T_0]\) that ensure universal approximation of three layer regular FNN's to hold [41]. However these results solve only the first problem partly, and do not answer the second problem. To solve the universal approximation of regular FNN's completely, Chapter IV and Chapter V develop comprehensive and thorough discussion to above problems. And some realization algorithms for approximating procedure are built.

In practice many I/O relationships whose internal operations are characterized by fuzzy sets are inherently fuzzy and imprecise. For instance, the natural inference of human brain, industrial process control, chemical reaction and natural evolution process and so on [17, 30]. Regular FNN's have become the efficient tools to model these real processes, for example fuzzy regression models [28], data fitting models [22], telecommunication networks [42] and so on are the successful examples of regular FNN applications.

§1.4 Fuzzy systems and fuzzy inference networks

Fuzzified neural networks as a class of pure fuzzy systems can deal with
natural linguistic information efficiently. In practice, in addition to linguistic information, much more cases relate to data information. From a real fuzzy system we can get a collection of data information that characterizes its I/O relationship by digital sensor or data surveying instrument, so it is of very real importance to develop some systematic tools that are able to utilize linguistic and data information, synthetically. Fuzzy systems take an important role in the research related. In a fuzzy system we can deal with linguistic information by developing a family of fuzzy inference rules such as 'IF...THEN...'. And data information constitutes the external conditions that may adjust system parameters, including the membership functions of fuzzy sets and defuzzification etc, rationally. Using fuzzy inference networks we may represent a fuzzy system as the I/O relationship of a neural system, and therefore fuzzy systems also possess the function of self-learning and self-improving.

Since recent twenty years, fuzzy systems and fuzzy inference networks have attracted much attention for they have found useful in many applied fields such as pattern recognition [30, 33, 56], system modelling and identification [16, 24], automatic control [30, 53, 61], signal processing [12, 35, 60], data compression [47] and telecommunication [42] and so on. As in the research of neural networks we study the applications of fuzzy systems and fuzzy inference networks by taking their universal approximation as a start point. Therefore, in the following let us take the research on approximating capability of fuzzy systems and fuzzy inference networks as a thread to present a survey to theory and application of this two classes of systems.

1.4.1 Fuzzy systems

In practice there are common three classes of fuzzy systems [30, 61], they are pure fuzzy systems, Mamdani fuzzy systems and Takagi–Sugeno (T–S) fuzzy systems. Pure fuzzy systems deal mainly with linguistic information while the latter two fuzzy systems can handle both linguistic information and data information [61]. We can distinguish a Mamdani fuzzy system and a T–S fuzzy system by their inference rule consequents. The rule consequent forms of a Mamdani fuzzy system are fuzzy sets while ones corresponding to a T–S fuzzy system are functions of the system input variables. As shown in
Figure 1.2 is the typical architecture of a fuzzy system, which consists of three parts: fuzzifizer, pure fuzzy system and defuzzifizer. The internal structures of the pure fuzzy system are determined by a sequence of fuzzy inference rules. Suppose the fuzzy rule base is composed of $N$ fuzzy rules $R_1, ..., R_N$. For a given input vector $x$, by fuzzifizer we can get a singleton fuzzy set $\tilde{x}$. Using the fuzzy rule $R_j$ and the implication relation we can establish a fuzzy set $\tilde{Y}_j$ defined on the output space [30]. By a $t$-conorm $S$ (generally is chosen as $S = \lor$) we synthesize $\tilde{Y}_1, ..., \tilde{Y}_N$ to determine the fuzzy set $\tilde{Y}$ defined on the output space:

$$\tilde{Y}(y) = S(\tilde{Y}_1(y), S(\tilde{Y}_2(y), ..., S(\tilde{Y}_N(y))), ...).$$

We call $\tilde{Y}$ a synthesizing fuzzy set [30]. Finally we utilize the defuzzifier $D_e$ to establish the crisp output $y_0 = D_e(\tilde{Y})$.

As one of main subjects related to fuzzy systems, universal approximation has attracted much attention since the early 1990s [61, 62, 65, 72]. We can classify the achievements in the field into two classes. One belongs to existential results, that is, the existence of the fuzzy systems is shown by the Stone-Weierstrass Theorem [30, 61]. Such an approach may answer the existence problem of fuzzy systems under certain conditions. However, its drawbacks are obvious, since it can not deal with many importantly practical problems such as, how can the approximating procedure of fuzzy systems express the given I/O relationship? How is the accuracy related estimated? With the given accuracy how can the size of the fuzzy rule base of the corresponding fuzzy system be calculated? and so on. Moreover, such a way gives the strong restrictions to the antecedent fuzzy sets, the inference composition rules and defuzzification. That is, the fuzzy sets are Gaussian type fuzzy numbers, the compositions are based on $\sum - x$ or $\lor - x$, and the defuzzifier usually means the method of center of gravity. Another is the constructive proving method, that is, we may directly build the approximating fuzzy systems related by the constructive procedures. Recent years the research related has attracted many scholars' attention. Ying et al in [65] employ a general defuzzification [23] to generalize Mamdani fuzzy systems and T-S fuzzy systems, respectively. Moreover, the antecedent fuzzy sets and the composition fuzzy operators can be general, that is, the fuzzy sets may be chosen as general fuzzy numbers with certain ranking order and the composition may be $\lor - T$, where $T$ is a $t$-norm. And some necessary conditions for fuzzy system approximation and their comparison are built. Zeng et al in [72] propose some accuracy analysis methods for fuzzy system approximation, and an approximating fuzzy system with the given accuracy may be established accordingly. So the constructive methods can be more efficient and applicable.

Up to the present the research on the related problems focuses on the approximations of the fuzzy systems to the continuous functions and the realization of such approximations. Although the related achievements are of much
real significance, their application areas are definitely restricted. There are many important and fundamental problems in the field remain to be solved.

First, in addition to continuous functions, how about the universal approximation of fuzzy systems to other general functions? For instance, in the control processes to many nonlinear optimal control models and the pulse circuits, the related systems are non-continuous, but integrable. Therefore the research in which the fuzzy systems are generalized within a general framework and more general functions, including integrable functions are approximately represented by the general fuzzy systems with arbitrary degree of accuracy, are very important both in theory and in practice.

Another problem is 'Rule explosion' phenomenon that is caused by so called 'curse of dimensionality', meaning that in a fuzzy system the number of fuzzy rules may exponentially increases as the number of the input variables of the system increases. Although the fuzzy system research has attracted many scholars' attention, and the achievements related have been successfully applied to many practical areas, particularly to the fuzzy control, the applications are usually limited to systems with very few variables, for example, two or at most four input variables [62]. When we increase the input variables, the scale of the rule base of the fuzzy system is immediately becoming overmuch, consequently the system not implementable. So 'rule explosion' do seriously hinder the applications of the fuzzy systems.

To overcome above drawbacks, Raju et al defined in [53] a new type of fuzzy system, that is the hierarchical fuzzy system. Such a system is constructed by a series of lower dimensional fuzzy systems, which are linked in a hierarchical fashion. To realize the given fuzzy inferences, the number of fuzzy rules needed in the hierarchical system is the linear function of the number of the input variables. Thus, we may avoid the 'rule explosion'. Naturally we may put forward an important problem, that is, how may the representation capability of the hierarchical fuzzy systems be analyzed? Kikuchi et al in [31] show that it is impossible to give the precise expression of arbitrarily given continuous function by a hierarchical fuzzy system. So we have to analyze the approximation capability of hierarchical fuzzy systems, i.e. whether are hierarchical fuzzy systems universal approximator or not? If a function is continuously differentiable on the whole space, Wang in [62] shows the arbitrarily close approximation of the function by hierarchical fuzzy systems; and he also in [63] gave the sensitivity properties of hierarchical fuzzy systems and designed a suitable system structure. For each compact set \( U \) and the arbitrarily continuous, or integrable function \( f \) on \( U \), how may we find a hierarchical fuzzy system to approximate \( f \) uniformly with arbitrary error bounds \( \epsilon \)?

The third important problem is the fuzzy system approximations in stochastic environment. Recently the research on the properties of the artificial neural networks in the stochastic environment attracts many scholars' attention. The approximation capabilities of a class of neural networks to stochastic processes and the problem whether the neural networks are able to learn stochastic pro-
cesses are systematically studied. It is shown that the approximation identity neural networks can with mean square sense approximate a class of stochastic processes to arbitrary degree of accuracy. The fuzzy systems can simultaneously deal with data information and linguistic information. So it is undoubtedly very important to study the approximation in the stochastic environment, that is, the approximation capabilities of fuzzy systems to stochastic processes.

The final problem is to estimate the size of the rule base of the approximating fuzzy system for the given accuracy. The research related is the basis for constructing the related fuzzy systems.

The systematic study on above problems constitutes the central parts of Chapter VI and Chapter VII. Also many well-designed simulation examples illustrate our results.

1.4.2 Fuzzy inference networks

Fuzzy inference system can simulate and realize natural language and logic inference mechanic. A fuzzy inference network is a multilayer feedforward network, by which a fuzzy system can be expressed as the I/O relationship of a neural system. So a fuzzy system and its corresponding fuzzy inference network are functionally equivalent [30]. As a organic fusion of inference system and neural network, a fuzzy inference network can realize automobile generation and automobile matching of fuzzy rules. Further, it can adaptively adjust to adapt itself to the changes of conditions and to self-improve. Since the early 1990s, many achievements have been achieved and they have found useful in many applied areas, such as system modeling [15, 16], system identification [46, 56], pattern recognition [58] and system forecasting [45] and so on. Moreover, fuzzy inference networks can deal with all kinds of information including linguistic information and data information, efficiently since they possess adaptiveness and fault-tolerance. Thus, they can successfully be applied to noise image processing, boundary detection for noise images, classification and detection of noise, system modeling in noise environment and so on.

Theoretically the research on fuzzy inference networks focuses mainly on three parts: First, design a feedforward neural network to realize a known fuzzy system, so that the network architecture is as simple as possible [30]; Second, build some suitable learning algorithms, so that the connection weights are adjusted rationally to establish suitable antecedent and consequent fuzzy sets; Third, within a general framework study the fuzzy inference networks, which is based on some general defuzzifier [54]. Defuzzification constitutes one important object to study fuzzy inference networks and it attracts many scholars’ attention. There are mainly four defuzzification methods, they are center of gravity (COG) method [33, 54], maximum of mean(MOM) method [61], $\alpha$-cut integral method, and $p$-mean method [55]. In addition, many novel defuzzifications to the special subjects are put forward in recent years. They have respective starting point and applying fields, also they have themselves advantages and disadvantages. For example, the COG method synthesizes all
actions of points in the support of the synthesizing fuzzy set to establish a crisp output while some special functions of some particular points e.g. the points with maximum membership, are neglected. The MOM method takes only the points with maximum membership under consideration while the other points are left out of consideration. The $\alpha$-cut integral and $p$-mean methods are two other mean summing forms for all points in the support of the synthesizing set. The research on how we can define defuzzifications and fuzzy inference networks within a general framework attracts much attention. Up to now many general defuzzifiers have been put forward [23, 54, 55, 61]. However, they possess respective drawbacks: Either the general principles are too many to be applied conveniently or, the definitions are too concrete to be generalized to general cases.

To introduce some general principle for defuzzidication and to build a class of generalized fuzzy inference network within a general framework constitute a preliminary to study FNN application in image processing in Chapter VIII.

§1.5 Fuzzy techniques in image restoration

The objective of image restoration is to reconstruct the image from degraded one resulted from system errors and noises and so on. There are two ways to achieve such an objective [3, 52]. One is to model the corrupted image degraded by motion, system distortion, and additive noises, whose statistic models are known. And the inverse process may be applied to restore the degraded images. Another is called image enhancement, that is, constructing digital filters to remove noises to restore the corrupted images resulted from noises. Originally, image restoration included the subjects related to the first way only. Recently many scholars put the second way into the field of image restoration [12, 35, 60]. Linear filter theory is an efficient tool to process additive Gaussian noise, but it can not deal with non-additive Gaussian noise. So the research on nonlinear filters has been attracting many scholars' attention [3, 60, 66].

In practice, it is imperative to bring ambiguity and uncertainty in the acquisition or transmission of digital images. One may use human knowledge expressed heuristically in natural language to describe such images. But this approach is highly nonlinear in nature and can not be characterized by traditional mathematical modeling. The fuzzy set and fuzzy logic can be efficiently incorporated to do that. So it is convenient to employ fuzzy techniques in image processing. The related discussions appeared about in 1981 [48, 49], but not until 1994 did the systematic results related incurred. Recently fuzzy techniques are efficiently applied in the field of image restoration, especially in the filtering theory to remove system distortion and impulse noises, smooth non-impulse noises and enhance edges or other salient features of the image.

1.5.1 Crisp nonlinear filters

Rank selection (RS) filter is a useful nonlinear filtering model whose sim-
plest form is median filter [52]. The guidance for building all kinds of RS type filters is that removing impulsive noise while keeping the fine image structure. By median filter, impulsive type noise can be suppressed, but it removes fine image structures, simultaneously. When the noise probability exceeds 0.5, median filter can result in poor filtering performance. To offer improved performance, many generalizations of median filter have been developed. They include weighted order statistic filter, center weighted median filter, rank conditioned rank selection (RCRS) filter, permutation filter, and stack filter, etc (see [3, 52]). The RCRS filter is built by introducing feature vector and rank selection operator. It synthesizes all advantages of RS type filters, also it can be generalized as the neural network filter. Moreover, as a signal restoration model, the RCRS filter possesses the advantage of utilizing rank condition and selection feature of sample set simultaneously. Thus all RS type filters may be handled within a general framework [3]. However, although RS type filters improve median filter from different aspects, their own shortcomings are not overcome, for the outputs of all these filters are the observation samples in the operating window of the image. For example a RCRS filter may change the fine image structure while removing impulsive noise; when the noise probability \( p > 0.5 \) it is difficult to get a restoration with good performance; the complexity of the RCRS filter increases exponentially with the order (the length of operating window). Such facts have spurred the development of fuzzy filters, which improve the performance of RS filters by extending output range, soft decision and adaptive structure.

1.5.2 Fuzzy filters

The RCRS filter can not overcome the drawbacks of median filter thoroughly since its ultimate output is still chosen from the gray levels in the operating window. So fuzzy techniques may be used to improve the RS type filters from the following parts: extending output range, soft decision and fuzzy inference structure. Recent years fuzzy theory as a soft technique has been successfully applied in modeling degraded image and building noise filters.

Extending output range means generalizing crisp filters within a fuzzy framework. For example, by fuzzifying the selection function as a fuzzy rank, the RCRS filter can be generalized a new version—rank conditioned fuzzy selection (RCFS) filter [12]. It utilize natural language, such as ‘Dark’ ‘Darker’ ‘Medium’ ‘Brighter’ ‘Bright’ and so on to describe gray levels of the image related. And so the image information may be used more efficiently. Although the RCFS filter improves the performance of RCRS filter, as well as the filtering capability, the problems similar to RCRS filter arise still. So in more cases, soft decision or fuzzy inference structure are used to improve noise filters. Soft decision means that we may use fuzzy set theory to soften the constraint conditions for the digital image and to build the image restoration techniques. Civanlar et al in [17] firstly establish an efficient image restoration model by soft decision, in which the key part is to define suitable membership functions.
of fuzzy sets related. Yu and Chen in [66] generalize the stack filter as a fuzzy stack filter, by which the filtering performance is much improved. One of key steps to do that is fuzzifying a positive Boolean function (PBF) as a fuzzy PBF, by which we may estimate a PBF from the upper and the lower, respectively. And so the fuzzy stack filter concludes the stack filter as a special case. Fuzzy inference structure for image processing means that some fuzzy inference rules are built to describe the images to be processed, and then some FNN mechanisms are constructed to design noise filters. An obvious advantage for such an approach is that the fuzzy rules may be adjusted adaptively. The performance of the filters related may be advantageous in processing the high probability \((p > 0.5)\) noise images [35, 60]. A central part of Chapter VIII is to build some optimal FNN filters by developing suitable fuzzy inference rules and fuzzy inference networks.

Furthermore in [47], the fault-tolerance of fuzzy relational equations is the tool for image compression and reconstruction; And the classical vector median filter is generalized to the fuzzy one, and so on. Of course, the research of image restoration by fuzzy techniques has been in its infancy period, many elementary problems related are unsolved. Also to construct the systematic theory in the field is a main object for future research related to the subject.

§1.6 Notations and preliminaries

In the following let us present the main notations and terminologies used in the book, and account for the organization of the book.

Suppose \(\mathbb{N}\) is the natural number set, and \(\mathbb{Z}\) is the integer set. Let \(\mathbb{R}^d\) be \(d\)-dimensional Euclidean space, in which \(\| \cdot \|\) means the Euclidean norm. \(\mathbb{R} \triangleq \mathbb{R}^1\), and \(\mathbb{R}_+\) is the collection of all nonnegative real numbers. If \(x \in \mathbb{R}\), \(\text{Int}(x)\) means the maximum integer not exceeding \(x\).

By \(A, B, C, \ldots\) we denote the subsets of \(\mathbb{R}\), and \(\overline{A}\) is the closure of \(A\). For \(A, B \subseteq \mathbb{R}^d\), let \(d_H(A, B)\) be Hausdorff metric between \(A\) and \(B\), i.e.

\[
d_H(A, B) = \max \left\{ \bigvee_{x \in A} \bigwedge_{y \in B} \{\|x - y\|\}, \bigvee_{y \in B} \bigwedge_{x \in A} \{\|x - y\|\} \right\},
\]

(1.3)

where \(\bigvee\) means the supremum operator ‘sup’, and \(\bigwedge\) means the infimum operator ‘inf’. For the intervals \([a, b], [c, d] \subseteq \mathbb{R}\), define the metric \(d_E([a, b], [c, d])\) as follows:

\[
d_E([a, b], [c, d]) = \left\{ (a - c)^2 + (b - d)^2 \right\}^{\frac{1}{2}}.
\]

(1.4)

Give the intervals \([a, b], [c, d] \subseteq \mathbb{R}\), it is easy to show

\[
d_H([a, b], [c, d]) \leq d_E([a, b], [c, d]) \leq \sqrt{2} \cdot d_H([a, b], [c, d]),
\]

(1.5)

that is, the metrics \(d_E\) and \(d_H\) are equivalent. If \(X\) is universe, by \(\mathcal{F}(X)\) we denote the collection of all fuzzy sets defined on \(X\). Using \(\tilde{A}, \tilde{B}, \ldots\) we denote the
fuzzy sets defined on \( \mathbb{R} \). And \( \mathcal{F}_0(\mathbb{R}) \) means a subset of \( \mathcal{F}(\mathbb{R}) \) with the following conditions holding, i.e. for \( \tilde{A} \in \mathcal{F}_0(\mathbb{R}) \), we have

(i) The kernel of \( \tilde{A} \) satisfies, \( \text{Ker}(\tilde{A}) \triangleq \{ x \in \mathbb{R} | \tilde{A}(x) = 1 \} \neq \emptyset \);

(ii) \( \forall \alpha \in (0, 1] \), then \( \tilde{A}_\alpha \triangleq [a_\alpha^1, a_\alpha^2] \) is a bounded and closed interval;

(iii) The support of \( \tilde{A} \) satisfies, \( \text{Supp}(\tilde{A}) \triangleq \{ x \in \mathbb{R} | \tilde{A}(x) > 0 \} \) is a bounded and closed set of \( \mathbb{R} \).

We denote the support \( \text{Supp}(\tilde{A}) \) of a fuzzy set \( \tilde{A} \) by \( \tilde{A}_0 \). If \( \tilde{A}, \tilde{B} \in \mathcal{F}_0(\mathbb{R}) \), define the metric between \( \tilde{A} \) and \( \tilde{B} \) as \([19, 20]\):

\[
D(\tilde{A}, \tilde{B}) = \bigvee_{\alpha \in [0,1]} \{ d_H(\tilde{A}_\alpha, \tilde{B}_\alpha) \} = \bigvee_{\alpha \in [0,1]} \{ d_H(\tilde{A}_\alpha, \tilde{B}_\alpha) \}. \quad (1.6)
\]

By [19] it follows that \( (\mathcal{F}_0(\mathbb{R}), D) \) is a complete metric space. If we generalize the condition (ii) as

(ii)' \( \tilde{A} \) is a convex fuzzy set, that is the following fact holds:

\[
\forall x_1, x_2 \in \mathbb{R}, \forall \alpha \in [0, 1], \tilde{A}(\alpha x_1 + (1-\alpha)x_2) \geq \tilde{A}(x_1) \wedge \tilde{A}(x_2).
\]

Denote the collection of fuzzy sets satisfying (i)(ii)' and (iii) as \( \mathcal{F}_c(\mathbb{R}) \). If \( \tilde{A} \in \mathcal{F}_c(\mathbb{R}) \), then \( \tilde{A} \) is called a bounded fuzzy number. Obviously, \( \mathcal{F}_0(\mathbb{R}) \subset \mathcal{F}_c(\mathbb{R}) \). Also it is easy to show, (ii)' is equivalent to the fact that \( \forall \alpha \in [0, 1], \tilde{A}_\alpha \subset \mathbb{R} \) is a interval. Denote

\[
\mathcal{F}_0(\mathbb{R})^d = \mathcal{F}_0(\mathbb{R}) \times \cdots \times \mathcal{F}_0(\mathbb{R}).
\]

And for \( (\tilde{A}_1, \ldots, \tilde{A}_d), (\tilde{B}_1, \ldots, \tilde{B}_d) \in \mathcal{F}_0(\mathbb{R})^d \), we also denote for simplicity that

\[
D((\tilde{A}_1, \ldots, \tilde{A}_d), (\tilde{B}_1, \ldots, \tilde{B}_d)) = \sum_{i=1}^{n} D(\tilde{A}_i, \tilde{B}_i). \quad (1.7)
\]

It is easy to show, \( (\mathcal{F}_0(\mathbb{R})^d, D) \) is also a complete metric space. For \( \tilde{A} \in \mathcal{F}_0(\mathbb{R}), |\tilde{A}| \) means \( D(\tilde{A}, \{0\}) \), that is

\[
|\tilde{A}| = \bigvee_{\alpha \in [0,1]} \{ |a_\alpha^1| \vee |a_\alpha^2| \} \quad (\tilde{A}_\alpha = [a_\alpha^1, a_\alpha^2]).
\]

For a given function \( f : \mathbb{R}^d \longrightarrow \mathbb{R} \), we may extend \( f \) as \( \tilde{f} : \mathcal{F}_0(\mathbb{R})^d \longrightarrow \mathcal{F}(\mathbb{R}) \) by the extension principle [44]:

\[
\forall (\tilde{A}_1, \ldots, \tilde{A}_d) \in \mathcal{F}_0(\mathbb{R})^d, \tilde{f}(\tilde{A}_1, \ldots, \tilde{A}_d)(y) = \bigvee_{f(x_1, \ldots, x_d) = y} \left\{ \bigwedge_{i=1}^{d} \{ \tilde{A}_i(x_i) \} \right\}. \quad (1.8)
\]
For simplicity, we write also \( \tilde{f} \) as \( f \). And \( f \) is called an extended function.

\( C^1(\mathbb{R}) \) is the collection of all continuously differentiable functions on \( \mathbb{R} \); and \( C^1([a, b]) \) is the set of all continuously differentiable functions on the closed interval \( [a, b] \).

**Definition 1.1** [14] Suppose \( g : \mathbb{R} \rightarrow \mathbb{R} \), and \( F_N : \mathbb{R}^d \rightarrow \mathbb{R} \) is a three layer feedforward neural network whose transfer function is \( g \). That is

\[
\forall (x_1, ..., x_d) \in \mathbb{R}^d, \quad F_N(x_1, ..., x_d) = \sum_{j=1}^{p} v_j \cdot g \left( \sum_{i=1}^{d} w_{ij} \cdot x_i + \theta_j \right).
\]

If \( F_N(\cdot) \) constitute a universal approximator, then \( g \) is called a Tauber-Wiener function.

If \( g \) is a generalized sigmoidal function \( \sigma : \mathbb{R} \rightarrow \mathbb{R} \), that is, \( \sigma \) is bounded, and \( \lim_{x \to +\infty} \sigma(x) = 1, \lim_{x \to -\infty} \sigma(x) = 0 \). Then by [14], it follows that \( g \) is a Tauber-Wiener function.

We call \( g : \mathbb{R} \rightarrow \mathbb{R} \) a continuous sigmoidal function, if \( g \) is continuous and increasing, moreover, \( \lim_{x \to +\infty} \sigma(x) = 1, \lim_{x \to -\infty} \sigma(x) = 0 \). Obviously, a continuous sigmoidal function is a Tauber-Wiener function.

Let \( \mu \) be a Lebesgue measure on \( \mathbb{R}^d \), and \( f : \mathbb{R}^d \rightarrow \mathbb{R} \) be a measurable function. Give \( p \in [1, +\infty) \). If \( f \) is a \( p \)-integrable function on \( \mathbb{R}^d \), define the \( L_p(\mu) \)-norm of \( f \) as follows:

\[
\|f\|_{\mu,p} = \left\{ \int_{\mathbb{R}^d} |f(x)|^p d\mu \right\}^{\frac{1}{p}}.
\]

If \( A \subset \mathbb{R}^d \), and \( \mu \) is Lebesgue measure on \( A \), we let

\[
\|f\|_{A,p} = \left\{ \int_{A} |f(x)|^p d\mu \right\}^{\frac{1}{p}},
\]

\[
L^p(\mathbb{R}, \mathcal{B}, \mu) \triangleq L^p(\mu) = \{ f : \mathbb{R}^d \rightarrow \mathbb{R} | \|f\|_{\mu,p} < +\infty \},
\]

\[
L^p(\mathcal{A}) = \{ f : \mathbb{R}^d \rightarrow \mathbb{R} | \|f\|_{A,p} < +\infty \},
\]

where \( \mathcal{B} \) is a \( \sigma \)-algebra on \( \mathbb{R} \). And \( \mathcal{C}_F \) is a sub-class of collection of continuous fuzzy functions that \( \mathcal{F}_0(\mathbb{R})^d \rightarrow \mathcal{F}_0(\mathbb{R}) \). For \( n, m \in \mathbb{N} \), by \( \mu_{n \times m} \) we denote the collection of all fuzzy matrices with \( n \) rows and \( m \) columns. For \( x^1 = (x^1_1, ..., x^1_n), \ x^2 = (x^2_1, ..., x^2_n) \in [0, 1]^n \), we denote

\[
x^1 \lor x^2 = (x^1_1 \lor x^2_1, ..., x^1_n \lor x^2_n),
\]

\[
x^1 \land x^2 = (x^1_1 \land x^2_1, ..., x^1_n \land x^2_n).
\]

Other terminologies and notations not being emphasized here, the readers may find them in the respective chapters, or sections, in which they are utilized.
§1.7 Outline of the topics of the chapters

The book tries to develop FNN theory through three main types of FNN models. They are FNN's based on fuzzy operators which are respectively treated in Chapter II and Chapter III; Fuzzified neural networks taking regular FNN's and polygonal FNN's as main components, which are dealt with by Chapter IV and Chapter V, respectively; Fuzzy inference networks being able to realize the common fuzzy systems, such as Mamdani fuzzy systems, T–S fuzzy systems and stochastic fuzzy systems and so on, which are handled in Chapter VI, Chapter VII and Chapter VIII. In each chapter we take some simulation examples to illustrate the effectiveness of our results, especially the FNN models and the learning algorithms related.

Chapter II treats two classes of FNN models—feedforward fuzzy associative memory (FAM) for storing fuzzy patterns, which can also recall right fuzzy patterns stored, and fuzzy adaptive resonance theory (ART) for classifying fuzzy patterns. A fuzzy pattern related can be expressed as a vector whose components belong to \([0, 1]\). To improve the storage capability of a FAM, we in §2.1 build a novel feedforward FNN—FAM with threshold, that is, introduce a threshold to each neural unit in a FAM. Some equivalent conditions that a given family of fuzzy pattern pairs can be stored in the FAM completely are established. Moreover, an analytic learning algorithm for connection weights and thresholds, which guarantees the FAM to store the given fuzzy pattern pair family is built. To take advantage of the adaptivity of neural systems we build two classes of iteration learning algorithms for the FAM, which are called the fuzzy delta algorithm and the fuzzy BP algorithm in §2.2 and §2.3, respectively.

§2.4 focuses on a fuzzy classifying network—fuzzy ART. After recalling some fundamental concepts of ART1 we define a fuzzy version of ART1 through fuzzy operators 'V' and 'A'. We characterize the classifying procedure of the fuzzy ART and develop some useful properties about how a fuzzy pattern is classified. Finally, corresponding a crisp ARTMAP we propose its fuzzy version—fuzzy ARTMAP by joining two fuzzy ART’s together. Many simulation examples are studied in detail to illustrate our conclusions.

Chapter III deals with another type of FNN's based on fuzzy operators 'V' and 'A'—feedback FAM’s, which are dynamic FNN's. We focus on two classes of dynamic FAM’s, they are fuzzy Hopfield networks and fuzzy bidirectional associative memories (FBAM’s). §3.1 reports many useful dynamic properties of the fuzzy Hopfield networks by studying attractors and attractive basins. And based on fault-tolerance we develop an analytic learning algorithm, by which some correct fuzzy patterns may be recalled through some imprecise inputs. To improve the storage capability and fault-tolerance the fuzzy Hopfield networks with threshold are reported in §3.2. It is also shown that the dynamical systems are uniformly stable and their attractors are Lyapounov stable. In §3.3 and §3.4 the corresponding problems for FBAM’s are analyzed, systematically. At first we show the fact that the FBAM’s converge their equilibrium stables,
i.e. attractors or limit cycles. And then some learning algorithms based on fault-tolerance are built. Many simulation examples are shown to illustrate our conclusions. The transitive laws of attractors, the discrimination of the pseudo-attractors of this two dynamical FNN’s are presented in §3.5 and §3.6, respectively. The basic tools to do these include connection networks, fuzzy row-restricted matrices and elementary memories and so on.

Chapter IV develops the systematic theory of regular FNN’s by focusing mainly on two classes of important problems, they are learning algorithms for the fuzzy weights of regular FNN’s and approximating capability, i.e. universal approximation of regular FNN’s to fuzzy functions. To this end, we at first introduce regular fuzzy neurons, and present their some useful properties. Then we define regular FNN’s by connecting a group of regular fuzzy neurons. Here a regular FNN means mainly a multi-layer feedforward FNN. And give some results about the I/O relationships of regular FNN’s. Buckley’s conjecture ‘the regular FNN’s can be universal approximators of the continuous and increasing fuzzy function class’ is proved to be false by a counterexample. However it can be proven that regular FNN’s can approximate the extended function of any continuous function with arbitrarily given degree of accuracy on any compact set of $F_c(\mathbb{R})$. In §4.3 we introduce a novel error function related to three layer feedforward regular FNN’s and develop a fuzzy BP algorithm for the fuzzy weights. The basic tools to do that are the $\vee - \wedge$ function and the polygonal fuzzy numbers. Using the fuzzy BP algorithm we can employ a three layer regular FNN to realize a family of fuzzy inference rules approximately. To speed the convergence of the fuzzy BP algorithm, §4.4 develops a fuzzy CG algorithm for the fuzzy weights of the three layer regular FNN’s, whose learning constant in each iteration is determined by GA. It is also shown in theory that the fuzzy CG algorithm is convergent to the minimum point of the error function. Simulation examples also demonstrate the fact that the fuzzy CG algorithm improves indeed the fuzzy BP algorithm in convergent speed. In §4.5 we take the fuzzy Bernstein polynomial as a bride to show that the four layer feedforward regular FNN’s can be approximators to the continuous fuzzy valued function class. The realization steps of the approximating procedure are presented and illustrated by a simulation example. Taking these facts as the basis we in §4.6 develop some equivalent conditions for the fuzzy function class $C_F$, which can guarantee universal approximation of four layer regular FNN’s to hold. Moreover, an improved fuzzy BP algorithm is developed to realize the approximation with a given accuracy. Thus, the universal approximation problem for four layer regular FNN’s is solved completely. Finally in the chapter we in §4.7 employ a regular FNN to represent integrable bounded fuzzy valued functions, approximately with integral norm sense.

In Chapter V we proceed to analyze universal approximation of regular FNN’s. The main problem to solve is to simplify the equivalent conditions of the fuzzy function class $C_F$ in Chapter IV, which can ensure universal approximation of four layer regular FNN’s. The main contributions are to introduce a
novel class of FNN models—polygonal FNN's and to present useful properties of the FNN's, such as topological architecture, internal operations, I/O relationship analysis, approximation capability and learning algorithm and so on. To this end we in §5.1 at first develop uniformity analysis for three layer, and four layer crisp feedforward neural networks, respectively. For a given function family the crisp neural networks can approximate each function uniformly with a given accuracy. Also we can construct the approximating neural networks directly through the function family. §5.2 reports the topological and analytic properties of the polygonal fuzzy number space $\mathcal{F}^n_{bc}(\mathbb{R})$, for instance, the space is a completely separable metric space; also it is locally compact; a subset in the space is compact if and only if the set is bounded and closed; a bounded fuzzy number can be a limit of a sequence of polygonal fuzzy numbers, and so on. Moreover, Zadeh's extension principle is improved in $\mathcal{F}^n_{bc}(\mathbb{R})$, by developing a novel extension principle and fuzzy arithmetic. Thus, many extended operations such as extended multiplication and extended division and so on can be simplified strikingly. Based on the novel extension principle §5.3 defines the polygonal FNN, which is a three layer feedforward network with polygonal fuzzy number input, output and connection weights. Similarly with §4.3 a fuzzy BP algorithm for fuzzy weights of the polygonal FNN's is developed and it is successfully applied to the approximate realization of fuzzy inference rules. §5.4 treats universal approximation of the polygonal FNN's, and shows the fact that a fuzzy function class can guarantee universal approximation of the polygonal FNN's if and only if each fuzzy function in this class is increasing, which simplifies the corresponding conditions in §4.6, strikingly. So the polygonal FNN's are more applicable.

Chapter VI deals mainly with the approximation capability of generalized fuzzy systems with integral norm. The basic tool to do that is the piecewise linear function that is one central part in §6.1. Also a few of approximation theorems for the piecewise linear functions expressing each $L_p(\mu)$—integrable function are established. In §6.2 we define the generalized fuzzy systems which include generalized Mamdani fuzzy systems and generalized T–S fuzzy systems as special cases. And show the universal approximation of the generalized fuzzy systems to $L_p(\mu)$—integrable functions with integral norm sense; For a given accuracy $\in > 0$, a upper bound of the size of fuzzy rule base of a corresponding approximating fuzzy system is estimated. One main impediment to hinder the application of fuzzy systems is 'rule explosion' problem, that is, the size of the fuzzy rule base of a fuzzy system increases exponentially as the input space dimensionality increasing. To overcome such an obstacle we in §6.3 employ the hierarchy introduced by Raju et al to define hierarchical fuzzy systems, by which the 'rule explosion' problem can be solved successfully. Moreover, a hierarchical fuzzy system and the corresponding higher dimension fuzzy system are equivalent. So the hierarchical fuzzy systems can be universal approximators with maximum norm and with integral norm respectively, on which we main focus in §6.4. Thus, the fuzzy systems can also applied to the
cases of higher dimension complicated system. Many simulation examples are presented to illustrate the approximating results in the chapter.

Some further subjects about approximation of fuzzy systems are studied in Chapter VII, that is, we discuss the approximation capability of fuzzy systems in stochastic environment. To this end we in §7.1 recall some basic concepts about stochastic analysis, for instance, stochastic integral, stochastic measure and canonical representation of a stochastic process and so on. §7.2 introduces two class of stochastic fuzzy systems, they are stochastic Mamdani fuzzy systems and stochastic T–S fuzzy systems, which possess many useful properties. For example, their stochastic integrals with respect to an orthogonal incremental process exist, and the stochastic integrals can expressed approximately as an algebra summation of a sequence of random variables. Using the fundamental results in §6.2 the systematic analysis of approximating capability of stochastic fuzzy systems including stochastic Mamdani fuzzy systems and stochastic T–S fuzzy systems with mean square sense is presented, which is central part in §7.3 and §7.4, respectively. Learning algorithms for stochastic Mamdani fuzzy systems and stochastic T–S fuzzy systems are also developed, and approximating realization procedure of some stochastic processes including a class of non-stationary processes by stochastic fuzzy systems are demonstrated by some simulation examples.

Chapter VIII focuses mainly on the application of FNN's in image restoration. At first we treat fuzzy inference networks within a general framework, and so §8.1 introduces a general fuzzy inference network model by define generalized defuzzifier, in which includes the common fuzzy inference networks as special cases. In theory the generalized fuzzy inference networks can be universal approximators, which provides us with the theoretic basis for the applications of generalized fuzzy inference networks. In dynamical system identification we demonstrate by some real examples that the performance resulting from generalized fuzzy inference networks is much better than that from crisp neural networks or fuzzy systems with the Gaussian type antecedent fuzzy sets. In §8.2 we propose the FNN representations of a 2-D digital image by define the deviation fuzzy sets and coding the image as the connection weights of a fuzzy inference network. Such a representation is accurate when the image is uncorrupted, and the image can be completely reconstructed; when the image is corrupted, the representation may smooth noises and serve as a filter. Based on the minimum absolute error (MAE) criterion we design an optimal filter FR, whose filtering performance is much better than that of median filter. FR can preserve the uncorrupted structure of the image and remove impulse noise, simultaneously. However, when the noise probability exceeds 0.5, i.e. $p > 0.5$ FR may result in bad filtering performance. In order to improve FR in high probability noise, §8.3 develops a novel FNN—selection type FNN, which can be a universal approximator and is suitable for the design of noise filters. Also based on MAE criterion, the antecedent fuzzy sets of the selection type FNN are adjusted rationally, and an optimal FNN filter is built. It preserves the
uncorrupted structures of the image as many as possible, and also to a greatest extend it removes impulse noise. So by the FNN filter the restoration image with high quality may be built from the corrupted image degraded by high or low probabilities impulse noises. Further, the FNN filter also can suppress some hybrid type noises. By many real examples we demonstrate that the restoration images with good performances can be obtained through the filter FR, or the FNN filter. Especially the filtering performances of FNN filters to restore high probability \( p > 0.5 \) noise images may be much better than that of RS type filters, including RCRS filter.

References


CHAPTER II

Fuzzy Neural Networks for Storing and Classifying

If the fuzzy information handled by a FNN flows in one direction, from input to output, such a FNN is called a feedforward FNN. In this chapter, we focus mainly on such feedforward FNN's whose internal operations are based on the fuzzy operator pair ‘V - A’, which is called fuzzy associative memories (FAM’s). A FAM constitutes a fuzzy perceptron, which was firstly proposed by Kosko B. about in 1987. It is developed based on a crisp feedforward neural network by introducing the fuzzy operators ‘V’ and ‘A’. The fuzzy information can be described by vectors in [0, 1]ⁿ. An important subject related to FAM’s is the storage capacity of the network [9, 10, 20, 23, 48], since the hardware and computation requirements for implementing a FAM with good storage capacity can be reduced, significantly. So there exist a lot of researches about the storage capacity of FAM in recent years. At first Kosko [25] develops a fuzzy Hebbian rule for FAM’s, but it suffers from very poor storage capacity. To make up the defects of the fuzzy Hebbian rule, Fan et al improve Kosko’s methods with maximum solution matrix in [12] to develop some equivalent conditions, under which a family of fuzzy pattern pairs can be stored in a FAM, completely. Recent years FAM’s have been applied widely in many real fields, such as fuzzy relational structure modeling [18, 21, 39], signal processing [42], pattern classification [43, 45-47] and so on.

The classifying capability is another important subject related to the storage capacity of a FNN. The stronger the classifying ability of a FNN is, the more the FNN can store fuzzy patterns. By introducing the fuzzy operators ‘V’ and ‘A’ the crisp adaptive resonance theory (ART) can be generalized as a FNN model—fuzzy ART, which can provide us with much easier classification for a given fuzzy pattern family [7].

In the chapter we present further researches about FAM’s in storage capacity, learning algorithm for the connection weight matrices, associative space and so on. Some optimal connecting fashions of the neurons in a FAM, and some learning algorithms are developed based on storage capacity of the FAM. Finally we propose some systematic approaches to deal with the fuzzy ART, and its many classifying characteristics are developed. Some real examples show stronger classifying capability of the fuzzy ART.
§2.1 Two layer max–min fuzzy associative memory

Since the fuzzy operators '∨' and '∧' can adapt the outputs to prefix range, such as [0, 1], also the operation '∧' is a threshold function [2, 3], no transfer function in FAM's is considered in the following. Suppose the input signal \( \mathbf{x} \in [0, 1]^n \), and the output signal \( \mathbf{y} \in [0, 1]^m \). Thus the input–output (I/O) relationship of a two layer FAM can be expressed as: \( \mathbf{y} = \mathbf{x} \circ \mathbf{W} \), where '○' means '∨–∧' composition operation, \( \mathbf{W} = (w_{ij})_{n \times m} \in \mu_{n \times m} \) is the connection weight matrix, that is, if let \( \mathbf{x} = (x_1, \ldots, x_n) \), \( \mathbf{y} = (y_1, \ldots, y_m) \), then

\[
y_j = \bigvee_{i=1}^{n} \{ x_i \land w_{ij} \} \quad (j = 1, \ldots, m). \tag{2.1}
\]

Give a fuzzy pattern pair family as \( (\mathcal{X}, \mathcal{Y}) = \{(x_k, y_k) | k \in \mathcal{P} \} \), where \( \mathbf{x}_k = (x^k_1, \ldots, x^k_n) \), \( \mathbf{y}_k = (y^k_1, \ldots, y^k_m) \), and \( \mathcal{P} = \{1, \ldots, p\} \) (\( p \in \mathbb{N} \)). One of the main objects for studying (2.1) is to develop some learning algorithms for \( \mathbf{W} \), so that each pattern pair in \( (\mathcal{X}, \mathcal{Y}) \) can be stored in (2.1). Next let us present the topological architecture corresponding to (2.1), as shown in Figure 2.1.

For the fuzzy pattern pair family \( (\mathcal{X}, \mathcal{Y}) = \{(x_k, y_k) | k \in \mathcal{P} \} \), Kosko in [25] develops a fuzzy Hebbian learning algorithm for \( \mathbf{W} \), that is by the following formula \( \mathbf{W} \) can be established:

\[
\mathbf{W} = \bigvee_{k \in \mathcal{P}} \{ \mathbf{x}_k^T \circ \mathbf{y}_k \}, \tag{2.2}
\]

where \( \mathbf{x}_k^T \) means the transpose of \( \mathbf{x}_k \). The analytic learning algorithm (2.2) can not ensure each pattern pair \( (\mathbf{x}_k, \mathbf{y}_k) \) (\( k \in \mathcal{P} \)) to be stored in (2.1). To guarantee more pattern pairs in \( (\mathcal{X}, \mathcal{Y}) \) to be stored in (2.1), we improve the algorithm (2.2) in the following. Denote \( \mathcal{M} = \{1, \ldots, m\} \), \( \mathcal{N} = \{1, \ldots, n\} \), and

\[
\begin{align*}
G_{ij}(\mathcal{X}, \mathcal{Y}) &= \{ k \in \mathcal{P} | x^k_i > y^k_j \}, \\
E_{ij}(\mathcal{X}, \mathcal{Y}) &= \{ k \in \mathcal{P} | x^k_i = y^k_j \}, \\
GE_{ij}(\mathcal{X}, \mathcal{Y}) &= G_{ij}(\mathcal{X}, \mathcal{Y}) \cup E_{ij}(\mathcal{X}, \mathcal{Y}), \\
L_{ij}(\mathcal{X}, \mathcal{Y}) &= \{ k \in \mathcal{P} | x^k_i < y^k_j \}, \\
LE_{ij}(\mathcal{X}, \mathcal{Y}) &= L_{ij}(\mathcal{X}, \mathcal{Y}) \cup E_{ij}(\mathcal{X}, \mathcal{Y}).
\end{align*}
\]
By the following analytic learning algorithm we can re-establish the connection weight matrix $W = W_0 = (w_{ij}^0)_{n \times m}$ in (2.1):

$$w_{ij}^0 = \begin{cases} \bigwedge_{k \in G_{ij}(x, y)} \{y_j^k\}, & G_{ij}(x, y) \neq \emptyset, \\ 1, & G_{ij}(x, y) = \emptyset \end{cases} \quad (2.3)$$

For $i \in N$, $j \in M$, define the sets $S_{ij}^G(W_0, y)$ and $M^w$ respectively as follows:

$$S_{ij}^G(W_0, y) = \{k \in GE_{ij}(x, y) | y_j^k \leq w_{ij}^0\};$$
$$M^w = \{W \in \mu_{n \times m} | \forall k \in P, x_k \circ W = y_k\}.$$

**Theorem 2.1** For the given fuzzy pattern pair family $\{(x_k, y_k) | k \in P\}$, and $W_0 = (w_{ij}^0)_{n \times m}$, we have

(i) $\forall k \in P$, $x_k \circ W_0 \subset y_k$, and if the fuzzy matrix $W$ satisfies: $\forall k \in P, x_k \circ W \subset y_k$, then $W \subset W_0$;

(ii) If $M^w \neq \emptyset$, it follows that, $W_0 \in M^w$, and $\forall W = (w_{ij})_{n \times m} \in M^w$, $W \subset W_0$, i.e. $\forall i \in N, j \in M$, $w_{ij} \leq w_{ij}^0$;

(iii) The set $M^w \neq \emptyset$ if and only if $\forall j \in M$, $\bigcup_{i \in N} S_{ij}^G(W_0, y) = P$.

**Proof.** (i) By the definition of $W_0$, $\forall k \in P$, $\forall j \in M$, easily we can show, $k \in G_{ij}(x, y), \implies w_{ij}^0 \leq y_j^k$. Moreover

$$\bigvee_{i \in N} \{w_{ij}^0 \wedge x_i^k\} = \left( \bigvee_{i | k \in G_{ij}(x, y)} \{w_{ij}^0 \wedge x_i^k\} \right) \lor \left( \bigvee_{i | k \notin G_{ij}(x, y)} \{w_{ij}^0 \wedge x_i^k\} \right) \leq y_j^k.$$ 

Therefore, $\forall k \in P, x_k \circ W_0 \subset y_k$. Also if $W \in \mu_{n \times m}$ satisfies the given conditions, then we can conclude that

$$\bigvee_{i \in N} \{x_i^k \wedge w_{ij}\} \leq y_j^k, \implies \forall i \in N, w_{ij} \wedge x_i^k \leq y_j^k, \implies \forall k \in G_{ij}(x, y), w_{ij} \leq y_j^k.$$ 

So if $G_{ij}(x, y) \neq \emptyset$, $w_{ij} \leq \bigwedge_{k \in P} \{y_j^k\}$, by (2.3) we have, $\forall i \in N, j \in M$, $w_{ij} \leq w_{ij}^0$, that is, $W \subset W_0$. So (i) is true.

(ii) By the assumption we suppose $W = (w_{ij})_{n \times m} \in M^w$. For any $k \in P$, $j \in M$, we can conclude that

$$\bigvee_{i \in N} \{x_i^k \wedge w_{ij}\} = y_j^k, \implies \forall i \in N, w_{ij} \wedge x_i^k \leq y_j^k, \implies \forall k \in G_{ij}(x, y), w_{ij} \leq y_j^k.$$ 

Similarly with (i) we can show, $W \subset W_0$. Also for any $k \in P, j \in M$, it is easy to show the following fact:

$$\forall i \in N, w_{ij}^0 \wedge x_i^k \leq y_j^k, \implies y_j^k \geq \bigvee_{i \in N} \{x_i^k \wedge w_{ij}^0\} \geq \bigvee_{i \in N} \{x_i^k \wedge w_{ij}\} = y_j^k.$$
Thus, $V K f C A \subseteq \{y^k, i \in \mathbb{N}\}$, $W_0 \in M^w$. (ii) is proved.

(iii) Let $M^w \neq \emptyset$, and $W = (w_{ij})_{n \times m} \in M^w$. Then by (i), $W_0 = (w_{ij}^0)_{n \times m} \in M^w$, $W \subset W_0$. If there is $j_0 \in M$, satisfying $\bigcup_{i \in \mathbb{N}} S_{i,j_0}^G (W_0, \mathcal{Y}) \neq \emptyset$, then there exists $k_0 \in \mathcal{P}$, but $\forall i \in \mathbb{N}, k_0 \not\in S_{i,j_0}^G (W_0, \mathcal{Y})$, and hence, either $x_{i,k_0} < y_{j_0}^k$ or, $y_{j_0}^k > w_{ij_0}^0 \geq w_{ij}$. Therefore, $w_{ij_0} \land x_{i,k_0} < y_{j_0}^k$. So $\bigvee_{i \in \mathbb{N}} \{x_{i,k_0} \land w_{ij_0}^0\} < y_{j_0}^k$, which is a contradiction since $W \in M^w$. So $\forall j \in M$, $\bigcup_{i \in \mathbb{N}} S_{i,j}^G (W_0, \mathcal{Y}) = \emptyset$. On the other hand, let $\bigcup_{i \in \mathbb{N}} S_{i,j}^G (W_0, \mathcal{Y}) = \emptyset$ ($j \in \mathbb{M}$). For any $j \in \mathbb{M}$, $k \in \mathcal{P}$, there is $i_0 \in \mathbb{N}$, so that $k \in S_{i_0,j}$. Thus

$$x_{i_0} \geq y_j^k, w_{ij_0}^0 \geq y_j^k, \Rightarrow x_{i_0} \land w_{ij_0}^0 \geq y_j^k, \Rightarrow \bigvee_{i \in \mathbb{N}} \{x_{i}^k \land w_{ij}^0\} \geq y_j^k.$$ (2.4)

By the definition (2.3) for $w_{ij}^0$, it is easy to show

$$\forall i \in \mathbb{N}, j \in \mathbb{M}, k \in \mathcal{P}, x_{i}^k \land w_{ij}^0 \leq y_j^k, \Rightarrow \bigvee_{i \in \mathbb{N}} \{x_{i}^k \land w_{ij}^0\} \leq y_j^k.$$

Synthesizing (2.4) we get, $W_0 \in M^w$, $\Rightarrow M^w \neq \emptyset$. (iii) is true. □

![Figure 2.2 Topological architecture of three layer FAM](image)

**Figure 2.2** Topological architecture of three layer FAM

In the network (2.1), for the given connection weight matrix $W \in \mu_{n \times m}$, define

$$P_2^a (W) = \{(x, y) \in [0, 1]^n \times [0, 1]^m | x \circ W = y\},$$

The set $P_2^a (W)$ is called the associative space of (2.1). In practice, a main problem to study FAM is how to design the network (2.1) so that it can store as many fuzzy pattern pairs as possible, which can be viewed as such a problem that enlarging the associative space of (2.1). For the FAM's based on the fuzzy operator pair $\langle V - \Lambda \rangle$, it is impossible to treat the problem by increasing unit or, node layers of FAM's. To account for the fact, we propose a three layer FAM, as shown in Figure 2.2.
Suppose the output of the hidden unit \( k \) in Figure 2.2 is \( o_k \), then the corresponding I/O relationship can be expressed as follows:

\[
\begin{align*}
o_k &= \bigvee_{i=1}^{n} \{ x_i \wedge w^{(1)}_{ik} \} \quad (k = 1, \ldots, l), \\
y_j &= \bigvee_{k=1}^{l} \{ o_k \wedge w^{(2)}_{kj} \} \quad (j = 1, \ldots, m).
\end{align*}
\]

(2.5) stands for a three layer FAM. Next let us prove that the storage capacities of (2.1) and (2.5) are identical.

**Theorem 2.2** Let \( W_1 = (w^{(1)}_{ik})_{n \times l} \), \( W_2 = (w^{(2)}_{ik})_{l \times m} \), and \( P^a_3(W_1, W_2) \) is the associative space of the three layer FAM (2.5), i.e.

\[ P^a_3(W_1, W_2) = \{ (x, y) \mid x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_m) \text{ satisfy (2.5)} \}. \]

Then we can conclude that

(i) For given \( W_1, W_2 \), there is \( W \in \mu_{n \times m} \), so that \( P^a_3(W_1, W_2) \subset P^a_2(W) \);

(ii) If \( l \geq m \land n \), then for given \( W \in \mu_{n \times m} \), there are \( W_1 \in \mu_{n \times l}, W_2 \in \mu_{l \times m} \), so that \( P^a_2(W) \subset P^a_3(W_1, W_2) \).

**Proof.** (i) For any \((x, y) \in P^a_3(W_1, W_2) : x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_m)\), by the assumption we get

\[
y_j = \bigvee_{k=1}^{l} \left\{ \left( \bigvee_{i=1}^{n} \{ x_i \wedge w^{(1)}_{ik} \} \right) \wedge w^{(2)}_{kj} \right\} = \bigvee_{k=1}^{l} \left\{ \bigvee_{i=1}^{n} \{ x_i \wedge w^{(1)}_{ik} \wedge w^{(2)}_{kj} \} \right\},
\]

where, \( j \in M \). Define the connection weight matrix of (2.1), \( W = (w_{ij})_{n \times m} \) as follows:

\[
w_{ij} = \bigvee_{k=1}^{l} \{ w^{(1)}_{ik} \wedge w^{(2)}_{kj} \} \quad (i \in N, j \in M).
\]

Then by (2.6) it follows that, \( \forall (x, y) \in P^a_3(W_1, W_2) : x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_m) \), we have

\[
\forall j \in M, \ y_j = \bigvee_{i=1}^{n} \{ x_i \wedge w_{ij} \},
\]

i.e. \( x \circ W = y \iff (x, y) \in P^a_2(W) \). Therefore, \( P^a_3(W_1, W_2) \subset P^a_2(W) \). (i) is proved.

(ii) Let \( l \geq m \). For any \((x, y) \in P^a_2(W) : x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_m)\), then for \( j \in M \), \( y_j = \bigvee_{i=1}^{n} \{ x_i \wedge w_{ij} \} \). For \( i \in N, j \in M \) and \( k = 1, \ldots, l \), define

\[
w^{(1)}_{ik} = \begin{cases} w_{ik}, & k \leq m, \\ 0, & m < k \leq l \end{cases}, \quad w^{(2)}_{kj} = \begin{cases} 1, & k \leq m, k = j \\ 0, & \text{otherwise}. \end{cases}
\]
Then for $k = 1, \ldots, l$, by (2.5) easily we can get
\[
 o_k = \bigvee_{i=1}^{n} \{ x_i \wedge w_{ik}^{(1)} \} = \begin{cases} 
 \bigvee_{i=1}^{n} \{ x_i \wedge w_{ik} \}, & k \leq m, \\
 0, & m < k \leq l.
\end{cases}
\]
Thus, we may conclude that
\[
 \forall j \in M, \bigvee_{k=1}^{l} \{ o_k \wedge w_{kj}^{(2)} \} = o_j = \bigvee_{i=1}^{n} \{ x_i \wedge w_{ij} \} = y_j.
\]
So if let $W_1 = (w_{ik}^{(1)})_{n \times l}$, $W_2 = (w_{kj}^{(2)})_{k \times m}$, then $(x, y) \in P_3^a(W_1, W_2)$. Therefore, $P_2^a(W) \subset P_3^a(W_1, W_2)$. (ii) is proved. □

By Theorem 2.2, increasing unit layer of a FAM based on \(\lor - \land\) can not improve the storage capacity. To improve FAM's in their storage capacity or associative space, let us now aim at the optimization of the connecting fashions among the units.

2.1.1 FAM with threshold

In the FAM's as shown in Figure 2.1, we introduce thresholds $c_i$, $d_j$ to the input unit $i$ and output unit $j$, respectively, where $i \in N$, $j \in M$. Then the corresponding I/O relationship can be expressed as
\[
 y_j = \left( \bigvee_{i=1}^{n} \{(x_i \lor c_i) \land w_{ij}\} \right) \lor d_j = \bigvee_{i=1}^{n} \{(x_i \lor c_i \lor d_j) \land (w_{ij} \lor d_j)\}, \tag{2.7}
\]
(2.7) is called a FAM with threshold. Using the fuzzy matrix $W = (w_{ij})_{n \times m}$ and the fuzzy vector $c = (c_1, ..., c_n)$, $d = (d_1, ..., d_m)$ we re-write (2.7) as
\[
 y = ((x \lor c) \circ W) \lor d. \tag{2.8}
\]
For $j \in M$, we introduce the following set:
\[
 J_i(\mathcal{X}, \mathcal{Y}) = \{ j \in M | LE_{ij}(\mathcal{X}, \mathcal{Y}) \neq \emptyset \}.
\]
By the following (2.9) (2.10) we establish the connection weight matrix $W_0 = (w_{ij}^0)_{n \times m}$ and the threshold vectors $c_i^0 = (c_i^0, ..., c_n^0)$, $d_i^0 = (d_i^0, ..., d_m^0)$:
\[
 w_{ij}^0 = \begin{cases} 
 \land_{k \in G_{ij}(\mathcal{X}, \mathcal{Y})} \{ y_k \}, & G_{ij}(\mathcal{X}, \mathcal{Y}) \neq \emptyset, \\
 1, & G_{ij}(\mathcal{X}, \mathcal{Y}) = \emptyset;
\end{cases} \tag{2.9}
\]
\[
 c_i^0 = \begin{cases} 
 \lor_{k \in LE_{ij}(\mathcal{X}, \mathcal{Y}), j \in J_i(\mathcal{X}, \mathcal{Y})} \{ y_k \}, & J_i(\mathcal{X}, \mathcal{Y}) \neq \emptyset, \\
 0, & J_i(\mathcal{X}, \mathcal{Y}) = \emptyset.
\end{cases} \tag{2.10}
\]
And $d^0_j = \bigwedge_{k \in P} \{y^k_j\}$. For $i \in N$, $j \in M$, define the sets

$$TG_{ij}((X, c_0); d_0, Y) = \{k \in P \mid x^k_i \lor c^0_i \lor d^0_j > y^k_j\},$$

$$TE_{ij}((X, c_0); d_0, Y) = \{k \in P \mid x^k_i \lor c^0_i \lor d^0_j = y^k_j\},$$

$$TL_{ij}((X, c_0); d_0, Y) = \{k \in P \mid x^k_i \lor c^0_i \lor d^0_j < y^k_j\},$$

$$TGE_{ij}((X, c_0); d_0, Y) = TG_{ij}((X, c_0); d_0, Y) \cup TE_{ij}((X, c_0); d_0, Y),$$

$$TS_{ij}^G((W_0, d_0); Y) = \{k \in TGE_{ij}((X, c_0); d_0, Y) \mid y^k_j \leq w^0_{ij} \lor d^0_j\}.$$

Since $\forall i \in N, j \in M, TGE_{ij}((X, c_0); d_0, Y) \supset GE_{ij}(X, Y)$, and $d^0_j \leq w^0_{ij}$ are obviously true, we have

$$TS_{ij}^G((W_0, d_0); Y) = \{k \in TGE_{ij}((X, c_0); d_0, Y) \mid y^k_j \leq w^0_{ij}\},$$

$$TS_{ij}^G((W_0, d_0); Y) \supset S_{ij}^G(W_0, Y).$$

In (2.8) we give the connection weight matrix $W \in \mu_{n \times m}$, and the threshold vectors $c, d$. Define the set

$$TP^a(W, c, d) = \{(x, y) \in [0, 1]^n \times [0, 1]^m \mid ((x \lor c) \circ W) \lor d = y\}.$$  

We call $TP^a(W, c, d)$ the associative space of the FAM with threshold. Let $\{(x_k, y_k) \mid k \in P\}$ be a given fuzzy pattern pair family. define

$$M^{wcd} = \{(W, c, d) \mid \forall k \in P, ((x_k \lor c) \circ W) \lor d = y_k\}.$$  

**Theorem 2.3** Let $W = (w_{ij})_{n \times m} \in \mu_{n \times m}$, $c = (c_1, \ldots, c_n) \in [0, 1]^n$, and $d = (d_1, \ldots, d_m) \in [0, 1]^m$, $(W, c, d) \in M^{wcd}$. Then $\forall i \in N, j \in M, w_{ij} \leq w^0_{ij}$, $d_j \leq d^0_j$.

**Proof.** Let $a, b \in [0, 1]$, define `$\otimes$' as follows:

$$a \otimes b = \begin{cases} 1, & a \leq b, \\ b, & a > b. \end{cases}$$

Then we can show, $\forall a, b, c \in [0, 1]$, $a \otimes (a \land b) \geq b$. And $b > c, \implies a \otimes b \geq a \otimes c$. Since $(W, c, d) \in M^{wcd}$, it follows that $\forall j \in M, d_j \leq \bigwedge_{k \in P} \{y^k_j\} = d^0_j$.

So next it suffices to prove $w_{ij} \leq w^0_{ij}$. If $G_{ij}(X, Y) = \emptyset$, then $w^0_{ij} = 1 \geq w_{ij}$. If $G_{ij}(X, Y) \neq \emptyset$, there is $k_0 \in G_{ij}(X, Y)$, so that

$$y^k_j = \bigwedge_{k \in G_{ij}(X, Y)} \{y^k_j\}, \implies w^0_{ij} = \bigwedge_{k \in G_{ij}(X, Y)} \{y^k_j\} = y^k_j < x^k_i.$$
Using the definition of ‘\( \mathcal{S} \)’ and the assumptions easily we can show
\[
w_{ij}^0 = y_j^0 = x_i^0 \mathcal{S} y_j^0 = x_i^0 \mathcal{S} \left( \bigvee_{i' = 1}^n \left\{ (x_i^{k_0} \lor c_{i'} \lor d_j) \land (w_{i'j} \lor d_j) \right\} \right) \\
\geq x_i^{k_0} \mathcal{S} (x_i^{k_0} \lor w_{ij}) \geq w_{ij}.
\]
The theorem is therefore proved. \( \square \)

By Theorem 2.3, \( c_0 \), \( W_0 \) defined by (2.9) possess a maximality with the sense of storing fuzzy patterns.

**Theorem 2.4** Let \( (W, c, d) \in M^{wcd} \). Then there is a threshold vector \( c_1 = (c_1^1, \ldots, c_1^n) \), so that \( \forall i \in N, c_1^i \leq c_0^i \), and \( (W, c_1, d_0) \in M^{wcd} \).

**Proof.** For any \( i \in N \), define \( c_1^i \) as follows:
\[
c_1^i = \begin{cases} \\
\bigwedge_{k \in P, j \in M} \{y_j^k\}, & J_i(X, Y) \neq \emptyset, \\
0, & J_i(X, Y) = \emptyset.
\end{cases}
\]
If \( J_i(X, Y) = \emptyset \), then \( c_1^i = 0 \leq c_0^i \), and \( \forall k \in P, j \in M, x_i^k > y_j^k \). Then by (2.9) and Theorem 2.3 it follows that \( x_i^k > w_{ij}^0 \lor d_j^0 \geq w_{ij} \lor d_j \). Therefore
\[
J_i(X, Y) = \emptyset, \implies (x_i^k \lor c_i \lor d_j) \land (w_{ij} \lor d_j) = w_{ij} \lor d_j \\
\leq (x_i^k \lor c_1^i \lor d_j^0) \land (w_{ij} \lor d_j^0) \leq y_j^k. \tag{2.11}
\]
If \( J_i(X, Y) \neq \emptyset \), define the set \( KJ_i(c, Y) = \{(k, j) \in P \times M | c_i > y_j^k\} \). If \( KJ_i(c, Y) = \emptyset \), then we get
\[
\forall j \in M, k \in P, c_i \leq y_j^k, \ c_i \leq \bigwedge_{k \in P, j \in M} \{y_j^k\} = c_1^i.
\]
Therefore \( \forall j \in M, k \in P, c_i \leq y_j^k, \ c_i \leq \bigwedge_{k \in P, j \in M} \{y_j^k\} = c_1^i \).

Therefore we have
\[
(x_i^k \lor c_i \lor d_j) \land (w_{ij} \lor d_j) \leq (x_i^k \lor c_1^i \lor d_j^0) \land (w_{ij} \lor d_j^0) \leq y_j^k. \tag{2.12}
\]
If \( KJ_i(c, Y) \neq \emptyset \), then we can conclude that
\[
(k, j) \in KJ_i(c, Y), \implies w_{ij} \leq y_j^k, \implies w_{ij} \leq \bigwedge_{(k, j) \in KJ_i(c, Y)} \{y_j^k\}.
\]
Also it is easy to show the following facts:
\[
\bigwedge_{(k, j) \in KJ_i(c, Y)} \{y_j^k\} \leq \bigwedge_{(k, j) \notin KJ_i(c, Y)} \{y_j^k\}; \ w_{ij} \leq \bigwedge_{k \in P} \{y_j^k\}.
\]
That is, when \( KJ_i(c, Y) \neq \emptyset \), (2.12) holds also. So
\[
J_i(X, Y) \neq \emptyset, \implies (x_i^k \lor c_i \lor d_j) \land (w_{ij} \lor d_j) = w_{ij} \lor d_j \\
\leq (x_i^k \lor c_1^i \lor d_j^0) \land (w_{ij} \lor d_j^0) \leq y_j^k. \tag{2.13}
\]
Thus, by (2.11) (2.13) and \((W, c, d) \in M^{wcd}\), we get
\[
y_j^k = \bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i \lor d_j) \land (w_{ij} \lor d_j) \right\}
\]
\[
= \left( \bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i \lor d_j) \land (w_{ij} \lor d_j) \right\} \right) \lor
\]
\[
\bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i \lor d_j) \land (w_{ij} \lor d_j) \right\}
\]
\[
\leq \left( \bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij} \lor d_j^0) \right\} \right) \lor
\]
\[
\bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij} \lor d_j^0) \right\}
\]
\[
= \bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij} \lor d_j^0) \right\} \leq y_j^k.
\]
Hence \(\forall j \in M, k \in P, \bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij} \lor d_j^0) \right\} = y_j^k\). That is, \(y = ((x \lor c_1) \circ W) \lor d_0\). Consequently, \((W, c_1, d_0) \in M^{wcd}\).

**Lemma 2.1** Suppose \(j \in M, k \in P\). Then the following facts hold:
\[
y_j^k = \bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij}^0 \lor d_j^0) \right\}
\]
\[
> \bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij}^0 \lor d_j^0) \right\}.
\]

**Proof.** If \(k \not\in TS^G_{ij}((W_0, d_0); Y)\), then

either \(x_i^k \lor c_i^l \lor d_j^0 < y_j^k\) or, \(x_i^k \lor c_i^l \lor d_j^0 \geq y_j^k > w_{ij}^0 \lor d_j^0\),

which can implies, \((x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij}^0 \lor d_j^0) < y_j^k\). Therefore
\[
\bigvee_{i \in \mathbb{N}} \left\{ (x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij}^0 \lor d_j^0) \right\} < y_j^k.
\]

If \(k \in TS^G_{ij}((W_0, d_0); Y)\), then we have

either \(x_i^k \lor c_i^l \lor d_j^0 = y_j^k \leq w_{ij}^0 \lor d_j^0\) or, \(x_i^k \lor c_i^l \lor d_j^0 > y_j^k\).

And \(y_j^k \leq w_{ij}^0 \lor d_j^0\). Also by the definitions of \(w_{ij}^0\) and \(d_j^0\), it follows that
\(w_{ij}^0 \lor d_j^0 \geq y_j^k\). Hence
\[
w_{ij}^0 \lor d_j^0 = y_j^k, \quad (x_i^k \lor c_i^l \lor d_j^0) \land (w_{ij}^0 \lor d_j^0) = y_j^k.
That is, the following equality holds:

\[
\bigvee_{i,k \in TS_{ij}^{G}((W_0,d_0);\mathcal{Y})} \{(x_i^k \lor c_i^0 \lor d_j^0) \land (w_{ij}^0 \lor d_j^0)\} = y_j^k.
\]

And hence the lemma is proved. \(\square\)

**Theorem 2.5** For the given fuzzy pattern pair family \(\{(x_k, y_k) | k \in \mathcal{P}\}\), the set \(M^{wcd} \neq \emptyset\) if and only if \(\forall j \in \mathcal{M}, \bigcup_{i \in \mathcal{N}} TS_{ij}^{G}((W_0, d_0); \mathcal{Y}) = \mathcal{P}\).

**Proof.** Necessity: Let \(W = (w_{ij})_{n \times m} \in \mu_{n \times m}, \mathbf{c} = (c_1, ..., c_n) \in [0, 1]^n, \mathbf{d} = (d_1, ..., d_m) \in [0, 1]^m\), so that \((W, \mathbf{c}, \mathbf{d}) \in M^{wcd}\). If the conclusion is false, there is \(j_0 \in \mathcal{M}\), satisfying \(\bigcup_{i \in \mathcal{N}} TS_{ij_0}^{G}((W_0, d_0); \mathcal{Y}) \neq \mathcal{P}\). Thus, there is \(k \in \mathcal{P}\), so that \(\forall i \in \mathcal{N}, k \not\in TS_{ij_0}^{G}((W_0, d_0); \mathcal{Y})\). Therefore, \(\forall i \in \mathcal{N}\), either \(k \in TL_{ij_0}((\mathcal{X}, \mathbf{c}_0); d_0, \mathcal{Y})\) or, \(k \in TGE_{ij_0}((\mathcal{X}, \mathbf{c}_0); d_0, \mathcal{Y})\), \(y_{j_0}^k > w_{ij_0}^0 \lor d_j^0\). So

\[
\bigvee_{i \in \mathcal{N}} \{(x_i^k \lor c_i^0 \lor d_j^0) \land (w_{ij}^0 \lor d_j^0)\} = (\bigvee_{i,k \in TGE_{ij_0}((W_0,c_0);d_0,\mathcal{Y})} \{(x_i^k \lor c_i^0 \lor d_j^0) \land (w_{ij}^0 \lor d_j^0)\}) \lor \bigvee_{i,k \in TL_{ij_0}((W_0,c_0);d_0,\mathcal{Y})} \{(x_i^k \lor c_i^0 \lor d_j^0) \land (w_{ij}^0 \lor d_j^0)\} < y_{j_0}^k. \tag{2.14}
\]

By the assumption and Theorem 2.4, there is a fuzzy vector \(\mathbf{c}_1 = (c_{i_1}^1, ..., c_{i_n}^1)\) : \(\forall i \in \mathcal{N}, c_{i_1}^1 \leq c_i^0\), so that \((W, \mathbf{c}_1, \mathbf{d}_0) \in M^{wcd}\). So by (2.14) and Theorem 2.3 it follows that

\[
y_{j_0}^k = \bigvee_{i \in \mathcal{N}} \{(x_i^k \lor c_i \lor d_j) \land (w_{ij} \lor d_j)\} = \bigvee_{i \in \mathcal{N}} \{(x_i^k \lor c_i^0 \lor d_j^0) \land (w_{ij}^0 \lor d_j^0)\} \\
\leq \bigvee_{i \in \mathcal{N}} \{(x_i^k \lor c_i^0 \lor d_j^0) \land (w_{ij}^0 \lor d_j^0)\} < y_{j_0}^k,
\]

Which is a contradiction. The necessity is proved.

Sufficiency: \(\forall j \in \mathcal{M}, k \in \mathcal{P}\), there is \(i \in \mathcal{N}\), so that \(k \in TS_{ij}^{G}((W_0, d_0); \mathcal{Y})\). Hence if let \(w_{ij} = w_{ij}^0, c_i = c_i^0, d_j = d_j^0\), then Lemma 2.1 implies that

\[
\bigvee_{i \in \mathcal{N}} \{(x_i^k \lor c_i \lor d_j) \land (w_{ij} \lor d_j)\} = \bigvee_{i,k \in TS_{ij}^{G}((W_0,d_0);\mathcal{Y})} \{(x_i^k \lor c_i^0 \lor d_j^0) \land (w_{ij}^0 \lor d_j^0)\} \lor \\
\bigvee_{i,k \in TS_{ij}^{G}((W_0,d_0);\mathcal{Y})} \{(x_i^k \lor c_i^0 \lor d_j^0) \land (w_{ij}^0 \lor d_j^0)\} = y_j^k.
\]
Then put $W_0 = (w_{ij}^0), c_0 = (c_1^0, ..., c_n^0), d_0 = (d_1^0, ..., d_m^0)$. Thus, $(W_0, c_0, d_0) \in M_{wcd}$. That is, $M_{wcd} \neq \emptyset$. □

**Theorem 2.6** For a given fuzzy pattern family $\{(x_k, y_k)|k \in \mathcal{P}\}$, let $M_w \neq \emptyset$, i.e. there is $W \in \mu_{n \times m}$, so that $\forall k \in \mathcal{P}$, $x_k \circ W = y_k$. Let $W_0 = (w_{ij}^0), c_0 = (c_1^0, ..., c_n^0), d_0 = (d_1^0, ..., d_m^0)$. Then $(W_0, c_0, d_0) \in M_{wcd}$.

**Proof.** By Theorem 2.1 we get,

$$
\forall j \in \mathcal{M}, \bigcup_{i \in \mathcal{N}} S_{ij}^G(W_0, y) = P.
$$

Since $S_{ij}^G(W_0, y) \subset TS_{ij}^G((W_0, d_0); y)$, it follows that $\bigcup_{i \in \mathcal{N}} TS_{ij}^G((W_0, d_0); y) = P$. Theorem 2.4 implies, $M_{wcd} \neq \emptyset$, and $(W_0, c_0, d_0) \in M_{wcd}$. □

### 2.1.2 Simulation example

In the subsection we demonstrate that FAM (2.8) with threshold possesses good storage capacity by a simulation example. Let $\mathcal{N} = \{1, 2, 3, 4, 5\}, \mathcal{M} = \{1, 2, 3\}$, and $\mathcal{P} = \{1, ..., 8\}$. By Table 2.1 we give a fuzzy pattern pair family as $\{(x_k, y_k)|k \in \mathcal{P}\}$. Using the following steps we can realize the algorithm (2.9).

<table>
<thead>
<tr>
<th>$k$</th>
<th>$x_k$</th>
<th>$y_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0.5, 0.5, 0.4, 0.4, 0.3)</td>
<td>(0.5, 0.6, 0.3)</td>
</tr>
<tr>
<td>2</td>
<td>(0.1, 0.3, 0.3, 0.4, 0.4)</td>
<td>(0.5, 0.6, 0.4)</td>
</tr>
<tr>
<td>3</td>
<td>(0.8, 0.4, 0.6, 0.7, 0.4)</td>
<td>(0.6, 0.8, 0.4)</td>
</tr>
<tr>
<td>4</td>
<td>(0.3, 0.4, 0.4, 0.3, 0.4)</td>
<td>(0.5, 0.6, 0.4)</td>
</tr>
<tr>
<td>5</td>
<td>(0.6, 0.4, 0.7, 0.7, 0.5)</td>
<td>(0.7, 0.7, 0.5)</td>
</tr>
<tr>
<td>6</td>
<td>(0.1, 0.1, 0.2, 0.2, 0.1)</td>
<td>(0.5, 0.6, 0.3)</td>
</tr>
<tr>
<td>7</td>
<td>(0.7, 0.2, 0.4, 0.3, 0.2)</td>
<td>(0.5, 0.7, 0.3)</td>
</tr>
<tr>
<td>8</td>
<td>(0.8, 0.4, 0.3, 0.4, 0.2)</td>
<td>(0.5, 0.8, 0.3)</td>
</tr>
</tbody>
</table>

**Step 1.** For any $i \in \mathcal{N}, j \in \mathcal{M}$, calculate the sets $G_{ij}(\mathcal{X}, \mathcal{Y}), LE_{ij}(\mathcal{X}, \mathcal{Y})$, and establish $w_{ij}^0, c_i^0, d_j^0$;

**Step 2.** For $i \in \mathcal{N}, j \in \mathcal{M}$, calculate and determine the following sets:

$$
TG_{ij}((\mathcal{X}, c_0); d_0, y), TE_{ij}((\mathcal{X}, c_0); d_0, y),
$$

$$
TGE_{ij}((\mathcal{X}, c_0); d_0, y), TS_{ij}^G((W_0, d_0); y);
$$

**Step 3.** For any $j \in \mathcal{M}$, discriminate the following equality:

$$
\bigcup_{i \in \mathcal{N}} TS_{ij}^G((W_0, d_0); y) = P?
$$
if yes go to the following step, otherwise go to Step 5;

**Step 4.** Put \( W_0 = (w_{ij}^0) \), \( c_0 = (c_1^0, \ldots, c_n^0) \), \( d_0 = (d_1^0, \ldots, d_m^0) \);

**Step 5.** Stop.

By above steps we get, \( c_0 = (0.3, 0.3, 0.3, 0.3, 0.3) \), \( d_0 = (0.5, 0.6, 0.3) \), the threshold vectors of input units and output units, respectively, the connection weight matrix \( W_0 \) as follows:

\[
W_0^T = \begin{pmatrix}
0.5 & 1.0 & 1.0 & 0.6 & 1.0 \\
1.0 & 1.0 & 1.0 & 1.0 & 1.0 \\
0.3 & 0.3 & 0.3 & 1.0 & 1.0
\end{pmatrix}
\]

we may easily show that the given fuzzy pattern pair family \( \{(x_k, y_k)|k \in \mathcal{P}\} \) satisfies the conditions that for each \( j \in \mathcal{M}, \bigcup_{i \in \mathcal{N}} TS^G_{ij}((W_0, d_0); y) = \mathcal{P} \). So by Theorem 2.5, all fuzzy pattern pairs in Table 2.1 can be stored in FAM (2.8). If we use the Hebbian learning rule, easily we may imply that only \( (x_5, y_5) \) can be stored in FAM (2.1) [38]. Therefore, we can improve a FAM in storage capacity by introducing suitable thresholds and learning algorithms.

### §2.2 Fuzzy \( \delta \)-learning algorithm

Introducing threshold to units and designing analytic learning algorithm can improve a FAM as (2.1) in its storage capacity. However, analytic learning algorithm can not show the adaptivity and self-adjustability of FAM's. To overcome such defects we in this section develop a dynamic learning scheme, the fuzzy \( \delta \)-learning algorithm, and present its convergence.

#### 2.2.1 FAM's based on \( \vee \wedge \)

Give a fuzzy pattern pair family \( \{(x_k, y_k)|k \in \mathcal{P}\} \), and by the matrices \( X, Y \) we denote \( X = (x_1, \ldots, x_p)^T \), \( Y = (y_1, \ldots, y_p)^T \), that is

\[
X = \begin{pmatrix}
x_1^1 & x_2^1 & \cdots & x_n^1 \\
x_1^2 & x_2^2 & \cdots & x_n^2 \\
\vdots & \vdots & \ddots & \vdots \\
x_1^p & x_2^p & \cdots & x_n^p
\end{pmatrix}, \quad Y = \begin{pmatrix}
y_1^1 & y_2^1 & \cdots & y_m^1 \\
y_1^2 & y_2^2 & \cdots & y_m^2 \\
\vdots & \vdots & \ddots & \vdots \\
y_1^p & y_2^p & \cdots & y_m^p
\end{pmatrix}.
\]

Then all \( (x_1, y_1), \ldots, (x_p, y_p) \) can be stored in FAM (2.1) if and only if there is a fuzzy matrix \( W = (w_{ij})_{n \times m} \), satisfying

\[
X \circ W = Y.
\] (2.15)

(2.15) is a fuzzy relational equation based on \( \vee \wedge \) composition [27–29, 41, 42]. All \( (x_1, y_1), \ldots, (x_p, y_p) \) are memory patterns of FAM (2.1) if and only if the solutions of (2.15) exist. Moreover, using the following algorithm we
can demonstrate the learning procedure for the connection weight matrix $W$ of FAM (2.1), and establish a solution of (2.15).

**Algorithm 2.1** Fuzzy δ—learning algorithm. With the following steps we can realize the iteration of $w_{ij}$ for $i \in \mathbb{N}, j \in \mathbb{M}$:

1. **Initialization:** $\forall i \in \mathbb{N}, \ j \in \mathbb{M}$, put $w_{ij}(0) = 1$ and $t = 0$;
2. **Let $W(t) = (w_{ij}(t))_{n \times m}$**;
3. **Calculate the real output:** $Y(t) = X \circ W(t)$, that is
   $\forall k \in \mathbb{P}, \ \forall j \in \mathbb{M}, \ y^k_j(t) = \bigvee_{i=1}^{n} \{x^k_i \land w_{ij}(t)\}$.
4. **Adjust the connection weights:** Let $\eta \in (0, 1]$ be a learning constant, denote
   
   $w_{ij}(t + 1) = \begin{cases} 
   w_{ij}(t) - \eta \cdot (y^k_j(t) - y^k_j), & w_{ij}(t) \land x^k_i > y^k_j, \\
   w_{ij}(t), & \text{otherwise.}
   \end{cases}$

5. **Discriminate $w_{ij}(t + 1) = w_{ij}(t)$?** if yes stop; otherwise let $t = t + 1$, go to Step 2.

**Step 4.** Adjust the connection weights: Let $\eta \in (0, 1]$ be a learning constant, denote

\[
W_{ij}(t + 1) = \begin{cases} 
    w_{ij}(t) - \eta \cdot (y_j^k(t) - y_j^k), & w_{ij}(t) \land x_i^k > y_j^k, \\
    w_{ij}(t), & \text{otherwise.}
\end{cases}
\]

**Step 5.** $\forall i \in \mathbb{N}, \ j \in \mathbb{M}$, discriminate $w_{ij}(t + 1) = w_{ij}(t)$? if yes stop; otherwise let $t = t + 1$, go to Step 2.

Preceding to analyze the convergence of the fuzzy δ—learning algorithm 2.1, we present an example to demonstrate the realizing procedure of the algorithm. To this end let $P = N = \{1, 2, 3, 4\}$, and $M = \{1, 2, 3\}$. Give the fuzzy pattern pair family $\{(x_k, y_k)|k \in \mathbb{P}\}$ for training as follows:

\[
x_1 = (0.3, 0.4, 0.5, 0.6), \quad y_1 = (0.6, 0.4, 0.5), \\
x_2 = (0.7, 0.2, 1.0, 0.1), \quad y_2 = (0.7, 0.7, 0.7), \\
x_3 = (0.4, 0.3, 0.9, 0.8), \quad y_3 = (0.8, 0.4, 0.5), \\
x_4 = (0.2, 0.1, 0.2, 0.3), \quad y_4 = (0.3, 0.3, 0.3). 
\]

Then we can establish the fuzzy matrices $X, Y$ in (2.15) as

\[
X = \begin{pmatrix} 0.3 & 0.4 & 0.5 & 0.6 \\ 0.7 & 0.2 & 1.0 & 0.1 \\ 0.4 & 0.3 & 0.9 & 0.8 \\ 0.2 & 0.1 & 0.2 & 0.3 \end{pmatrix}, \quad Y = \begin{pmatrix} 0.6 & 0.4 & 0.5 \\ 0.7 & 0.7 & 0.7 \\ 0.8 & 0.4 & 0.5 \\ 0.3 & 0.3 & 0.3 \end{pmatrix}. 
\]

Choose $\eta = 0.8$, and with 40 iterations, the sequence of connection weight matrices $\{W(t)\}$ converges to the matrix $W$:

\[
W^{T} = \begin{pmatrix} 1.000000 & 1.000000 & 0.700000 & 1.000000 \\ 1.000000 & 1.000000 & 0.400000 & 0.400000 \\ 1.000000 & 1.000000 & 0.500000 & 0.500000 \end{pmatrix}. 
\]

Obviously (2.15) is true for $W$. 

Theorem 2.7 Suppose the fuzzy matrix sequence \( \{W(t) | t = 1, 2, \ldots \} \) is obtained by Algorithm 2.1. Then

(i) \( \{W(t) | t = 1, 2, \ldots \} \) is a non-increasing sequence of fuzzy matrices;

(ii) \( \{W(t) | t = 1, 2, \ldots \} \) converges.

Proof. (i) Let \( t \) mean the iteration step. For any \( i \in \mathbb{N}, j \in M, k \in P \), by (2.16), if \( x_i^k \land w_{ij}(t) > y_j^k \), we get, \( y_j^k(t) \geq x_i^k \land w_{ij}(t) > y_j^k \). Then \( w_{ij}(t + 1) = w_{ij}(t) - \eta(y_j^k(t) - y_j^k) < w_{ij}(t) \). If \( x_i^k \land w_{ij}(t) < y_j^k \), then \( w_{ij}(t + 1) = w_{ij}(t) \). Therefore, for \( i \in \mathbb{N}, j \in M, w_{ij}(t + 1) \leq w_{ij}(t) \), implying \( W(t + 1) \subseteq W(t) \). That is, \( \{W(t) | t = 1, 2, \ldots \} \) is a non-increasing fuzzy matrix sequence.

(ii) Since \( \forall t = 1, 2, \ldots, w_{ij}(t) \in [0, 1] \), we have, \( \forall i \in \mathbb{N}, j \in M \), the limit \( \lim_{t \to +\infty} w_{ij}(t) \) exists, that is, the matrix sequence \( \{W(t) | t = 1, 2, \ldots \} \) converges. \( \square \)

Theorem 2.7 can guarantee the convergence of Algorithm 2.1. If the solution set of (2.15) is non-empty, we in the following prove the limit matrix of the matrix sequence in Algorithm 2.1 is the maximum solution of (2.15).

Theorem 2.8 For a given fuzzy pattern pair family \( \{(x_k, y_k) | k \in P\} \), the fuzzy matrix sequence \( \{W(t) | t = 1, 2, \ldots \} \) defined by (2.16) converges to \( W_0 = (w_{ij}^0)_{n \times m} \), where \( w_{ij}^0 \) can be defined by (2.9). Moreover, if \( M^w \neq \emptyset \), then \( W_0 \in M^w \) is the maximum element of \( M^w \); if \( M^w = \emptyset \), then \( W_0 \) is the maximum element of the set \( \{W | X \circ W \subseteq Y\} \).

Proof. For any \( i \in \mathbb{N}, j \in M \), if \( G_{ij}(X, Y) = 0 \), then \( \forall k \in P, x_i^k \leq y_j^k \). In Algorithm 2.1, for any iteration step \( t \), we have, \( w_{ij}(t) \land x_i^k \leq y_j^k \). Then by (2.16) it follows that

\[
G_{ij}(X, Y) = 0 \quad \implies \quad w_{ij}(t + 1) = w_{ij}(t) = \cdots = w_{ij}(1) = w_{ij}(0) = 1. \quad (2.17)
\]

If \( G_{ij}(X, Y) \neq 0 \), there is \( k_0 \in G_{ij}(X, Y) \), so that \( y_j^{k_0} = \bigwedge_{k \in G_{ij}(X, Y)} \{y_j^k\} \). Then \( w_{ij}(0) \land x_i^{k_0} = x_i^{k_0} > y_j^{k_0} \). Next let us use (2.16) to show that

\[
\forall t \in \{1, 2, \ldots \}, w_{ij}(t) > y_j^{k_0}. \quad (2.18)
\]

In fact, if \( t = 1 \), considering \( 1 \geq x_i^{k_0} > y_j^{k_0} \) and (2.16) we get, \( w_{ij}(1) = w_{ij}(0) - \eta \cdot (y_j^{k_0}(0) - y_j^{k_0}) = 1 - \eta \cdot (x_i^{k_0} - y_j^{k_0}) > y_j^{k_0} \), where \( i_0 \in \mathbb{N} \) satisfies the following condition:

\[
y_j^{k_0} = \bigvee_{i' \in \mathbb{N}} \{w_{i'i}(0) \land x_i^{k_0}\} = \bigvee_{i' \in \mathbb{N}} \{x_i^{k_0}\} = x_i^{k_0}.
\]

If (2.18) is false, there is \( t' \in \mathbb{N} \), so that \( w_{ij}(t') \leq y_j^{k_0} \). Let \( t_0 \in \mathbb{N} : t_0 = \max\{t \in \mathbb{N} | w_{ij}(t) > y_j^{k_0}\} \). Then we get, \( t_0 \geq 1 \), and \( w_{ij}(t_0 + 1) \leq y_j^{k_0} \). So by (2.16) it follows that \( x_i^{k_0} \land w_{ij}(t_0 + 1) \leq y_j^{k_0} \), implying \( w_{ij}(t_0 + 2) = 1 > y_j^{k_0} \).
\( w_{ij}(t_0 + 1) \), which is a contradiction, since by (i) of Theorem 2.7, the fuzzy matrix sequence \( \{W(t)\}_{t = 1,2,...} \) is non-increasing. Hence (2.18) is true. Therefore, \( \lim_{t \to +\infty} w_{ij}(t) \geq y_j^{k_0} \). If \( \lim_{t \to +\infty} w_{ij}(t) \triangleq l_{ij} > y_j^{k_0} \), then

\[
y_j^{k_0}(t) = \bigvee_{i' \in \mathbb{N}} \{x_i^{k_0} \wedge w_{i'j}(t)\}, \implies \lim_{t \to +\infty} y_j^{k_0}(t) = \bigvee_{i' \in \mathbb{N}} \{x_i^{k_0} \wedge l_{i'j} \} > y_j^{k_0}.
\] (2.19)

Also by (2.16), \( w_{ij}(t + 1) = w_{ij}(t) - \eta \cdot (y_j^{k_0}(t) - y_j^{k_0}) \). Therefore, \( l_{ij} = l_{ij} - \eta \cdot \left( \lim_{t \to +\infty} y_j^{k_0}(t) - y_j^{k_0} \right) \), and \( \lim_{t \to +\infty} y_j^{k_0}(t) = y_j^{k_0} \), which contradicts (2.19). So \( G_{ij}(\mathcal{X}, \mathcal{Y}) \neq \emptyset \), \( \lim_{t \to +\infty} w_{ij}(t) = y_j^{k_0} \). Considering (2.17) we can conclude that

\[
\lim_{t \to +\infty} w_{ij}(t) = \left\{ \begin{array}{ll}
\bigwedge_{k \in G_{ij}(\mathcal{X}, \mathcal{Y})} \{y_j^k\}, & G_{ij}(\mathcal{X}, \mathcal{Y}) \neq \emptyset, \\
1, & G_{ij}(\mathcal{X}, \mathcal{Y}) = \emptyset,
\end{array} \right\} = w_{ij}^0.
\]

So the first part of the theorem holds. And the other part of the theorem is a direct result of Theorem 2.1. \( \square \)

In Algorithm 2.1 if we choose the learning constant \( \eta \) as an adjustable value changing with the iteration step \( t \), that is \( \eta(t) \), then the convergence speed of the algorithm can speed up, strikingly. We choose

\[
\eta = \eta(t) = \frac{w_{ij}(t) - w_{ij}(t) \wedge y_j^k}{y_j^k(t) - y_j^k}.
\]

Then (2.16) is transformed into the following iteration scheme:

\[
w_{ij}(t + 1) = \left\{ \begin{array}{ll}
w_{ij}(t) \wedge y_j^k, & w_{ij}(t) \wedge x_i^k > y_j^k, \\
w_{ij}(t), & \text{otherwise}.
\end{array} \right\} (2.20)
\]

By Theorem 2.8 the following theorem is trivial.

**Theorem 2.9** Let \( \{(x_k, y_k) | k \in \mathbb{P}\} \) be a fuzzy pattern pair family, and \( W(t) = (w_{ij}(t))_{n \times m} \) be a fuzzy matrix defined by (2.20). Then the sequence \( \{W(t) | t = 1,2,...\} \) converges to \( W_0 = (w_{ij}^0)_{n \times m} \) as \( t \to +\infty \), where \( w_{ij}^0 \) is defined by (2.9). Moreover, if \( M^w \neq \emptyset \), then \( W_0 \in M^w \) is a maximum element of \( M^w \); if \( M^w = \emptyset \), then \( W_0 \) is a maximum element of the \( \{W | X \circ W \subset Y\} \).

### 2.2.2 FAM's based on ‘\( \lor - \cdot \)’

Since the fuzzy operator pair ‘\( \lor - \cdot \)’ can not treat many real problems, it is necessary to study the FAM’s based on other fuzzy operator pairs. To this end we at first present the following definition [18, 35, 42, 52].

**Definition 2.1** We call the mapping \( T : [0,1]^2 \to [0,1] \) a fuzzy operator, if the following conditions hold:
(1) $T(0, 0) = 0$, $T(1, 1) = 1$;
(2) If $a, b, c, d \in [0, 1]$, then $a \leq c$, $b \leq d$, $\implies T(a, b) \leq T(c, d)$;
(3) $\forall a, b \in [0, 1]$, $T(a, b) = T(b, a)$;
(4) $\forall a, b, c \in [0, 1]$, $T(T(a, b), c) = T(a, T(b, c))$.

If $T$ is a fuzzy operator, and $\forall a \in [0, 1]$, $T(a, 1) = a$, we call $T$ a t-norm; If the fuzzy operator $T$ satisfies: $\forall a \in [0, 1]$, $T(0, a) = a$, we call $T$ a t-conorm.

From now on, we denote $T(a, b) = a \ast b$, and write the t-norm $T$ as $\ast$.

For $a, b \in [0, 1]$, define $a \alpha b \in [0, 1]: a \alpha b = \sup \{x \in [0, 1]|aT x \leq b\}$.

Let us now present some useful properties of the operator $\alpha$, the further discussions can see [18, 40, 52].

Lemma 2.2 Let $a, b, a_1, b_1 \in [0, 1]$, and $T$ be a t-norm. Then

(i) $a \leq a_1, \implies a \alpha b \leq a_1 \alpha b$;
(ii) $b \leq b_1, \implies a \alpha b \leq a \alpha b_1$;

Proof. (i) By the definition of the operator $\alpha$ it follows that $a \ast (a \alpha b) \leq b$. Since $a \ast b \leq a \ast b$, also using the definition of $\alpha$, we get, $a \alpha (a \ast b) \geq a$. Moreover, considering that $a \ast (a \alpha b) \leq b, \implies (a \alpha b) \ast a \leq b$, we can conclude that, $(a \alpha b) \alpha b \geq a$. (i) is true. As for (ii), they are also the direct results of the definition of $\alpha$. □

In (2.1) we substitute the t-norm $\ast$ for $\vee$, and get a FAM based on the fuzzy operator pair $\vee - \ast$:

$$y_j = \bigvee_{i \in \mathbb{N}} \{x_i \ast w_{ij}\} (j \in M). \quad (2.21)$$

If choose $\oplus$ as the $\vee - \ast$ composition operation, then (2.21) becomes as $y = x \oplus W$. Similar with (2.1), we can in (2.21) develop some analytic learning algorithms and iterative learning algorithms for the connection weight matrix $W$. For a given fuzzy pattern pair family $\{(x_k, y_k)|k \in P\}$, we can obtain a conclusion for the FAM (2.21) being similar with Theorem 2.1. To this end we at first design an analytic learning algorithm for $W$. Define $W_\ast = (w^*_{ij})_{n \times m} \in \mu_{n \times m}$ as follows:

$$w^*_{ij} = \bigwedge_{k \in P} \{x^k_i \alpha y^k_j\} (i \in \mathbb{N}, j \in M). \quad (2.22)$$

Recalling $S^G_{ij}(W_0, \mathcal{Y})$ and $M^w$ we may introduce the sets $S^*_{ij}(W_\ast, \mathcal{X}, \mathcal{Y}) (i \in \mathbb{N}, j \in M)$ and $M^w_\ast$ respectively as follows:

$$S^*_{ij}(W_\ast, \mathcal{X}, \mathcal{Y}) = \{k \in P|x^k_i \ast w^*_{ij} \geq y^k_j\},$$

$$M^w_\ast = \{W \in \mu_{n \times m}|\forall k \in P, x_k \oplus W = y_k\}.$$  

Theorem 2.10 Given a fuzzy pattern pair family $\mathcal{X}, \mathcal{Y}\{(x_k, y_k)|k \in P\}$, and $W_\ast = (w^*_{ij})_{n \times m}$ is defined by (2.22). Then
(i) \( \forall k \in P, x_k \otimes W_+ \subset y_k \), and if the fuzzy matrix \( W \) satisfies: \( \forall k \in P, x_k \otimes W \subset y_k \), we have, \( W \subset W_* \);
(ii) If \( M_*^w \neq \emptyset \), it follows that \( W_* \in M_*^w \), and \( \forall W = (w_{ij})_{n \times m} \in M_*^w \), \( W \subset W_* \), i.e. \( \forall i \in N, j \in M, w_{ij} \leq w_+^* \);
(iii) The set \( M_*^w \neq \emptyset \) if and only if \( \forall j \in M, \bigcup_{i \in N} S_{ij}^G (W_*, X, Y) = P \).

Proof. (i) For any \( k \in P \), and \( j \in M \), by Lemma 2.2 and (2.22) it follows that the following inequalities hold:
\[
\bigvee_{i \in N} \{ x_i^k \ast w_{ij}^* \} = \bigvee_{i \in N} \{ x_i^k \ast \left( \bigwedge_{k' \in P} \{ x_i^{k'} \ast y_j^{k'} \} \right) \} \leq \bigvee_{i \in N} \{ x_i^k \ast (x_i^k \ast y_j^k) \} \leq y_j^k,
\]
that is, \( x_k \otimes W_* \subset y_k \). And if \( W = (w_{ij})_{n \times m} \in \mu \), satisfies: \( x_k \otimes W \subset y_k \), for \( k \in P, j \in M \), we can conclude that
\[
\bigvee_{i \in N} \{ x_i^k \ast w_{ij} \} \leq y_j^k, \quad \Rightarrow x_i^k \ast w_{ij} \leq y_j^k, \quad \Rightarrow w_{ij} \leq x_i^k \ast y_j^k.
\]
Therefore, \( w_{ij} \leq \bigwedge_{k \in P} \{ x_i^k \ast y_j^k \} = w_+^* \). So \( W \subset W_* \). Thus, (i) is true.

(ii) Let \( W = (w_{ij})_{n \times m} \in M_*^w \). Then \( \forall k \in P, x_k \otimes W = y_k \). Similarly with (i) we can show, \( W \subset W_* \). And for any \( j \in M, k \in P \), the following fact holds:
\[
\bigvee_{i \in N} \{ x_i^k \ast w_{ij} \} = y_j^k, \quad \Rightarrow \bigvee_{i \in N} \{ x_i^k \ast w_{ij}^* \} \geq y_j^k.
\]

Lemma 2.2 and (2.22) can imply the following conclusion:
\[
\bigvee_{i \in N} \{ x_i^k \ast w_{ij}^* \} = y_j^k. \quad \text{That is, } W_* \in M_*^w. \quad \text{(ii) is proved.}
\]

(iii) At first assume \( M_*^w \neq \emptyset \), and \( W = (w_{ij})_{n \times m} \in M_*^w \). By (i) we have, \( W_* = (w_+^*)_{n \times m} \in M_*^w \), moreover \( W \subset W_* \). Then suppose there is \( j_0 \in M \), satisfying \( \bigcup_{i \in N} S_{ij_0}^G (W_*, X, Y) \neq P \). So there is \( k_0 \in P \), so that \( \forall i \in N \), \( j \in M \), \( x_i^{k_0} \subset y_j^{k_0} \). By the definition of \( w_+^* \) and Lemma 2.2 it is easy to show, \( \forall i \in N, j \in M, k \in P \), we have \( x_i^k \ast w_{ij}^* = x_i^k \ast \left( \bigwedge_{k' \in P} \{ x_i^{k'} \ast y_j^{k'} \} \right) \leq x_i^k \ast (x_i^k \ast y_j^k) \leq y_j^k \). Thus, \( \bigvee_{i \in N} \{ x_i^k \wedge w_{ij}^* \} \leq y_j^k \). Therefore, \( W_0 \in M_*^w, \Rightarrow M_*^w \neq \emptyset \). (iii) is true. \( \square \)
Next let us illustrate the application of Theorem 2.10 by a simulation example. Using the analytic learning algorithm (2.22) to establish the connection weight matrix $W_*$. Let $P = N = \{1, 2, 3, 4\}$, $M = \{1, 2, 3\}$, and the t-norm $*$ is defined as follows: $\forall a, b \in [0, 1]$, $a * b = \max\{a + b - 1, 0\}$. Give the fuzzy pattern pair family $\{(x_k, y_k) | k \in P\}$:

$$x_1 = (0.6, 0.5, 0.4, 0.3), \quad y_1 = (0.3, 0.6, 0.1);$$
$$x_2 = (0.5, 0.7, 0.8, 0.6), \quad y_2 = (0.6, 0.5, 0.4);$$
$$x_3 = (0.4, 0.7, 0.6, 0.4), \quad y_3 = (0.4, 0.5, 0.2);$$
$$x_4 = (0.8, 0.9, 0.7, 0.3), \quad y_4 = (0.5, 0.8, 0.4).$$

For $i \in N$, $j \in M$, $k \in P$, by the definition of $*$ it is easy to show

$$x_i^k \alpha_* y_j^k = \sup \{x \in [0, 1]| x_i^k * x \leq y_j^k \} = \sup \{x \in [0, 1]| x_i^k + x - 1 \leq y_j^k \} = \min\{1 + y_j^k - x_i^k, 1\}. $$

Therefore, by (2.22), $w_{ij}^* = \bigwedge_{k \in P} \{x_i^k \alpha_* y_j^k \} = \bigwedge_{k \in P} \{\min\{1 + y_j^k - x_i^k, 1\}\}$. So we get $W_* = (w_{ij}^*)_{4 \times 3}$:

$$W_*^T = \begin{pmatrix} 0.7 & 0.6 & 0.8 & 1.0 \\ 1.0 & 0.8 & 0.7 & 0.9 \\ 0.5 & 0.5 & 0.6 & 0.8 \end{pmatrix}$$

Moreover, for $i \in N$, $j \in M$, $S_{ij}^G(W_*, \mathcal{X}, \mathcal{Y}) = \{k \in P|x_i^k * w_{ij}^* \geq y_j^k\}$. Easily we have, $\forall j \in M$, $\bigcup_{i \in N} S_{ij}^G(W_*, \mathcal{X}, \mathcal{Y}) = P$. Therefore, by Theorem 2.10, each pattern pair in $\{(x_k, y_k) | k \in P\}$ can be stored in the FAM (2.21), and the corresponding connection weight matrix is $W_*$.  

Similarly with Algorithm 2.1, we can develop an iteration scheme for learning the connection weight matrix $W$ of the FAM (2.21), that is

**Algorithm 2.2** Fuzzy $\delta$—learning algorithm based on t—norm. With the following steps we can establish the connection weights of FAM (2.21):

**Step 1.** Initialization: put $t = 0$, and $w_{ij}(t) = 1$;

**Step 2.** Let $W(t) = (w_{ij}(t))_{n \times m}$;

**Step 3.** Calculate the real output: $y_j(t) = x_\otimes W(t)$, i.e.

$$y_j(t) = \sqrt[\text{in}]{\{x_i^k * w_{ij}(t)\}} \quad (j \in M, k \in P).$$

**Step 4.** Iteration scheme: The connection weights iterate with the following scheme (where $\eta \in (0, 1]$ is a learning constant):

$$w_{ij}(t + 1) = \begin{cases} w_{ij}(t) - \eta \cdot (y_j(t) - y_j^k), & w_{ij}(t) * x_i^k > y_j^k, \\ w_{ij}(t), & \text{otherwise.} \end{cases} (2.23)$$
Step 5. Discriminate $W(t + 1) = W(t)$? if yes, stop; otherwise let $t = t + 1$ go to Step 2.

Similarly with the equation (2.15), the FAM (2.21) can store each fuzzy pattern pair in $\{(x_k, y_k)|k \in P\}$ if and only if the following equalities hold:

$$
\begin{pmatrix}
x_1^k & x_2^k & \cdots & x_n^k \\
x_1^p & x_2^p & \cdots & x_n^p \\
\vdots & \vdots & \ddots & \vdots \\
x_1^m & x_2^m & \cdots & x_n^m \\
\end{pmatrix} \otimes
\begin{pmatrix}
w_{11} & w_{12} & \cdots & w_{1m} \\
w_{21} & w_{22} & \cdots & w_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
w_{n1} & w_{n2} & \cdots & w_{nm} \\
\end{pmatrix} =
\begin{pmatrix}
y_1^1 & y_2^1 & \cdots & y_m^1 \\
y_1^2 & y_2^2 & \cdots & y_m^2 \\
\vdots & \vdots & \ddots & \vdots \\
y_1^p & y_2^p & \cdots & y_m^p \\
\end{pmatrix},
$$

(2.24)

that is, $X \otimes W = Y$. By Theorem 2.8 and Theorem 2.9, we can get the following result.

**Theorem 2.11** Let $\{(x_k, y_k)|k \in P\}$ be a given fuzzy pattern pair family, and $\{W(t)|t = 1, 2, \ldots\}$ be a sequence of fuzzy matrices obtained by Algorithm 2.2. The t-norm '*' is continuous as a two-variate function. Then

(i) $\forall t = 1, 2, \ldots, W(t + 1) \subset W(t)$, and so $\{W(t)|t = 1, 2, \ldots\}$ converges;

(ii) If there is $W \in \mu_{nxm}$, so that (2.24) holds, then $\{W(t)|t = 1, 2, \ldots\}$ converges to the maximum solution of (2.24), i.e. $\lim_{t \to +\infty} W(t) = W_* = (w_{ij}^*)_{nxm}$;

(iii) If $\forall W \in \mu_{nxm},$ (2.24) is false, then $\{W(t)|t = 1, 2, \ldots\}$ converges to the maximum solution of $X \otimes W \subset Y$ as $t \to +\infty$.

**Proof.** It suffices to prove (ii), since the proofs of (i) (iii) are similar with ones of Theorem 2.8 by Theorem 2.10.

There is $k_0 \in Gij(X, Y)$, satisfying $x_i^{k_0} \alpha_* y_j^{k_0} = \bigwedge_{k \in P} \{x_i^k \alpha_* y_j^k\}$. If $x_i^{k_0} \leq y_j^{k_0}$, then we can conclude that

$$
\forall t = 1, 2, \ldots, x_i^{k_0} \ast w_{ij}(t) \leq y_j^{k_0}, \implies w_{ij}(1) = w_{ij}(2) = \cdots = 1.
$$

(2.25)

if $x_i^{k_0} > y_j^{k_0}$, by the continuity of the t-norm 'ast', $x_i^{k_0} \alpha_* y_j^{k_0} < 1$. Similarly with (2.18), let us next to prove by (2.16) that

$$
\forall t \in \{0, 1, 2, \ldots\}, w_{ij}(t) > x_i^{k_0} \alpha_* y_j^{k_0}.
$$

(2.26)

In fact, if $t = 0$, then $w_{ij}(0) = 1 > x_i^{k_0} \alpha_* y_j^{k_0}$. And if there is $t_1 \in \mathbb{N}$, so that $w_{ij}(t_1) \leq x_i^{k_0} \alpha_* y_j^{k_0}$, let $t_0 = \max\{t \in \{0, 1, \ldots\}, w_{ij}(t) \leq x_i^{k_0} \alpha_* y_j^{k_0}\}$. Then $t_0 > 0$, and $w_{ij}(t_0 + 1) \leq x_i^{k_0} \alpha_* y_j^{k_0}$. By Lemma 2.2 we get, $w_{ij}(t_0 + 1) \ast x_i^{k_0} \leq x_i^{k_0} \ast (x_i^{k_0} \alpha_* y_j^{k_0}) \leq y_j^{k_0}$. So (2.23) may imply, $w_{ij}(t_0 + 2) = 1 > w_{ij}(t_0 + 1)$, which contradicts (i), i.e. $\{W(t)|t = 0, 1, \ldots\}$ is non-increasing. Therefore (2.26) is true. Thus, $\lim_{t \to +\infty} w_{ij}(t) \geq x_i^{k_0} \alpha_* y_j^{k_0} = w_{ij}^*$. If $\lim_{t \to +\infty} w_{ij}(t) \triangleq l_{ij} > x_i^{k_0} \alpha_* y_j^{k_0} = w_{ij}^*$, then by the definition of 'ast', $x_i^{k_0} \ast l_{ij} > y_j^{k_0}$, so

$$
y_j^{k_0}(t) = \bigvee_{i' \in \mathbb{N}} \{ x_i^{k_0} \ast w_{i'j}(t) \}, \implies \lim_{t \to +\infty} y_j^{k_0}(t) = \bigvee_{i' \in \mathbb{N}} \{ x_i^{k_0} \ast l_{i'j} \} > y_j^{k_0}.
$$

(2.27)
And by (2.23) (2.26) and the definition of \( \alpha_* \) it follows that

\[
w_{ij}(t + 1) = w_{ij}(t) - \eta \cdot (y_j^{k_0}(t) - y_j^{k_0}),
\]

\[\Rightarrow l_{ij} = l_{ij} - \eta \cdot \left( \lim_{t \to +\infty} y_j^{k_0}(t) - y_j^{k_0} \right), \Rightarrow \lim_{t \to +\infty} y_j^{k_0}(t) = y_j^{k_0},\]

which contradicts (2.27). Thus, \( x_i^{k_0} > y_j^{k_0}, \Rightarrow \lim_{t \to +\infty} w_{ij}(t) = x_i^{k_0} \alpha_* y_j^{k_0}. \]

Hence by (2.25), \( \lim_{t \to +\infty} w_{ij}(t) = x_i^{k_0} \alpha_* y_j^{k_0} = w_{ij}. \) □

Next we discuss an application of Algorithm 2.2. Define the t–norm \( * \) as follows [27–29]:

\[a*b = \max\{0, a + b - 1\} (a, b \in [0, 1]).\]

And \( P = \{1, 2, 3\}, N = \{1, 2, 3\}, M = \{1\}. \) Let

\[X = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0.8 & 0.0 & 0.4 \\ 0.5 & 0.1 & 0.3 \\ 1.0 & 0.2 & 0.1 \end{pmatrix}; \ Y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0.2 \\ 0.7 \end{pmatrix}.\]

By Theorem 2.10, the fuzzy pattern pair family \( \{(x_k, y_k)|k = 1, 2, 3\} \) can be stored in the FAM (2.21). So the maximum solution of \( X \otimes W = Y \) exists, that is \( W_* = (0.7, 1.0, 0.9)^T. \) Table 2.2 shows the iteration step number of Algorithm 2.2 with different learning constants \( \eta \)'s and the ultimate connection weight matrix \( W: \)

<table>
<thead>
<tr>
<th>No.</th>
<th>learning constant (( \eta ))</th>
<th>iteration (( t ))</th>
<th>converged matrix (( W ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.90000</td>
<td>5</td>
<td>(0.64000, 1.00000, 0.90000)( ^T )</td>
</tr>
<tr>
<td>2</td>
<td>0.80000</td>
<td>7</td>
<td>(0.68000, 1.00000, 0.90000)( ^T )</td>
</tr>
<tr>
<td>3</td>
<td>0.50000</td>
<td>16</td>
<td>(0.69531, 1.00000, 0.90000)( ^T )</td>
</tr>
<tr>
<td>4</td>
<td>0.30000</td>
<td>30</td>
<td>(0.69948, 1.00000, 0.90000)( ^T )</td>
</tr>
<tr>
<td>5</td>
<td>0.10000</td>
<td>86</td>
<td>(0.69999, 1.00000, 0.90000)( ^T )</td>
</tr>
<tr>
<td>6</td>
<td>0.01000</td>
<td>594</td>
<td>(0.70000, 1.00000, 0.90000)( ^T )</td>
</tr>
</tbody>
</table>

By Table 2.2 we may see, the larger the learning constant \( \eta \) is, the quicker the convergent speed of the matrix sequence \( \{W(t)|t = 1, 2, \ldots\} \) is. The limit value \( W \) and the maximum solution \( W_* \) of (2.20) are not completely identical, for example, when \( \eta = 0.9 \) and \( \eta = 0.8, \) the difference between \( W \) and \( W_* \) is obvious. As \( \eta \) becomes smaller and smaller, \( W \) is close to \( W_* \), gradually. When \( \eta = 0.01, \) we get \( W = W_* \). Therefore, a meaningful problem related to Algorithm 2.2 is how to determine \( \eta \) so that the convergent speed and the sufficient closeness between \( W \) and \( W_* \) can be guaranteed, simultaneously.
§2.3 BP learning algorithm of FAM’s

In the section we present the back propagation (BP) algorithm for the connection weight matrix \( W \) of the FAM (2.1). Since for a given \( a \in [0, 1] \), the functions \( a \lor x \) and \( a \land x \) are not differentiable on \([0, 1]\) (see [51]), as a preliminary for the BP algorithm of FAM’s we at first define the differentiable functions ‘La’ and ‘Sm’, by which the fuzzy operators ‘\( \lor \)’ and ‘\( \land \)’ can be approximated, respectively and the derivatives in the BP algorithm can be calculated.

2.3.1 Two analytic functions

Next we build the approximately analytic representations of the fuzzy operators ‘\( \lor \)’ and ‘\( \land \)’, respectively, and establish the partial derivatives related. Define \( d \)-variate functions \( La, Sm : \mathbb{R}_+ \times [0, 1]^d \rightarrow [0, 1] \) as

\[
La(s; x_1, ..., x_d) = \frac{\sum_{i=1}^{d} x_i \cdot \exp\{sx_i\}}{\sum_{i=1}^{d} \exp\{sx_i\}}, \quad Sm(s; x_1, ..., x_d) = \frac{\sum_{i=1}^{d} x_i \cdot \exp\{-sx_i\}}{\sum_{i=1}^{d} \exp\{-sx_i\}}.
\]

By (2.28) it is easy to show, \( \forall x_1, ..., x_d \in [0, 1] \), the following facts hold:

\[
\forall s > 0, \quad La(s; x_1, ..., x_d), \quad Sm(s; x_1, ..., x_d) \in [x_1 \land \cdots \land x_d, x_1 \lor \cdots \lor x_d].
\]

For \( x_1, ..., x_d \in [0, 1] \), denote \( x_{\min} \triangleq x_1 \land \cdots \land x_d, \quad x_{\max} \triangleq x_1 \lor \cdots \lor x_d \). Moreover

\[
x_\delta = \begin{cases} \max\{x \in \{x_1, ..., x_d\}| x < x_{\max}\}, & \text{this set nonempty,} \\ x_{\max}, & \text{this set empty}; \end{cases}
\]

\[
x_\rho = \begin{cases} \min\{x \in \{x_1, ..., x_d\}| x > x_{\min}\}, & \text{this set nonempty,} \\ x_{\min}, & \text{this set empty}; \end{cases}
\]

Lemma 2.3 Suppose \( d > 1, \quad s > 0, \) and \( x_1, ..., x_d \in [0, 1] \). Then we have the following estimations:

\[
\left| \bigvee_{i=1}^{d} \{x_i\} - La(s; x_1, ..., x_d) \right| \leq (d - 1) \cdot \exp\{-s(x_{\max} - x_\delta)\};
\]

\[
\left| \bigwedge_{i=1}^{d} \{x_i\} - Sm(s; x_1, ..., x_d) \right| \leq (d - 1) \cdot \exp\{-s(x_\rho - x_{\min})\}.
\]

Therefore,

\[
\lim_{s \rightarrow +\infty} La(s; x_1, ..., x_d) = \bigvee_{i=1}^{d} \{x_i\}, \quad \lim_{s \rightarrow +\infty} Sm(s; x_1, ..., x_d) = \bigwedge_{i=1}^{d} \{x_i\}.
\]

Proof. At first let \( x_{\max} > x_{\min} \), and \( x_1 = \cdots = x_q = x_{\max} \). Then \( \forall i \in \)
\{q + 1, \ldots, d\}, x_i < x_{\text{max}}. \text{ Thus, when } s > 0, \text{ we can conclude that }

\begin{align*}
\left| \bigvee_{i=1}^{d} \{x_i\} - L_a(s; x_1, \ldots, x_d) \right| &= \frac{\sum_{i=1}^{d} x_i \exp\{sx_i\} - x_{\text{max}}}{\sum_{i=1}^{d} \exp\{sx_i\}} \\
&\leq \left| \sum_{i=1}^{d} x_i - x_{\text{max}} \right| \exp\{sx_i\}
\end{align*}

\begin{align*}
&\frac{\sum_{i=q+1}^{d} \left| x_i - x_{\text{max}} \right| \exp\{s(x_i - x_{\text{max}})\}}{\sum_{i=1}^{d} \exp\{s(x_i - x_{\text{max}})\}} \\
&\leq \frac{\sum_{i=q+1}^{d} \exp\{-s \cdot (x_{\text{max}} - x_{\delta})\}}{\sum_{i=q+1}^{d} \exp\{s(x_i - x_{\text{max}})\}} \leq \frac{d - q}{q} \cdot \exp\{-s \cdot (x_{\text{max}} - x_{\delta})\}
\end{align*}

\begin{align*}
&\leq \frac{d - q}{q} \cdot \exp\{-s \cdot (x_{\text{max}} - x_{\delta})\} \leq (d - 1) \cdot \exp\{-s \cdot (x_{\text{max}} - x_{\delta})\}.
\end{align*}

So if $x_{\text{max}} > x_{\text{min}}$, then we can obtain the following limit:

\begin{align*}
\lim_{s \to +\infty} L_a(s; x_1, \ldots, x_d) = \bigvee_{i=1}^{d} \{x_i\} \leq \lim_{s \to +\infty} (d - 1) \cdot \exp\{-s \cdot (x_{\text{max}} - x_{\delta})\} = 0
\end{align*}

when $x_{\text{max}} > x_{\delta}$. And therefore, \( \lim_{s \to +\infty} L_a(s; x_1, \ldots, x_d) = \bigvee_{i=1}^{d} \{x_i\} \); If $x_{\text{max}} = x_{\text{min}}$, then $x_1 = \cdots = x_d \implies L_a(s; x_1, \ldots, x_d) = x_1 = \bigvee_{i=1}^{d} \{x_i\}$. Hence the limit of $L_a(s; x_1, \ldots, x_d)$ as $s \to +\infty$ exists, and \( \lim_{s \to +\infty} L_a(s; x_1, \ldots, x_d) = \bigvee_{i=1}^{d} \{x_i\} \).

The first part of the theorem is proved. Similarly we can prove the other conclusions. \( \square \)

**Lemma 2.4** The functions $L_a(s; x_1, \ldots, x_d)$ and $S_m(s; x_1, \ldots, x_d)$ are continuously differentiable on $[0, 1]^d$. Moreover, for $j \in \{1, \ldots, d\}$, we have
(i) \[
\frac{\partial L_a(s; x_1, \ldots, x_d)}{\partial x_j} = \frac{-\exp(s x_j)}{\left(\sum_{i=1}^{d} \exp(s x_i)\right)^2} \left\{ \sum_{i=1}^{d} (s x_i - s x_j - 1) \exp(s x_i) \right\};
\]

(ii) \[
\frac{\partial S_m(s; x_1, \ldots, x_d)}{\partial x_j} = \frac{-\exp(-s x_j)}{\left(\sum_{i=1}^{d} \exp(-s x_i)\right)^2} \left\{ \sum_{i=1}^{d} (s x_i - s x_j + 1) \exp(-s x_i) \right\}.
\]

Proof. It suffices to show (i) since the proof of (ii) is similar. By the definition of \( L_a(s; x_1, \ldots, x_d) \) we can prove

\[
\frac{\partial L_a(s; x_1, \ldots, x_d)}{\partial x_j} = \sum_{i=1, i \neq j}^{d} \frac{\partial}{\partial x_j} \left( \frac{x_i \cdot \exp(s x_i)}{\sum_{i=1}^{d} \exp(s x_i)} \right) + \frac{\partial}{\partial x_j} \left( \frac{x_j \cdot \exp(s x_j)}{\sum_{i=1}^{d} \exp(s x_i)} \right)
\]

\[
= \frac{-\exp(s x_j)}{\left(\sum_{i=1}^{d} \exp(s x_i)\right)^2} \left\{ s x_j \exp(s x_j) + \sum_{i=1, i \neq j}^{d} s x_i \exp(s x_i) - (1 + s x_j) \sum_{i=1}^{d} \exp(s x_i) \right\}
\]

\[
= \frac{-\exp(s x_j)}{\left(\sum_{i=1}^{d} \exp(s x_i)\right)^2} \left\{ \sum_{i=1}^{d} (s x_i - s x_j - 1) \exp(s x_i) \right\}.
\]

So (i) is true. \( \square \)

By Lemma 2.4, we may conclude that the following facts hold for the constant \( a \in [0, 1] \):

\[
\begin{align*}
\frac{d L_a(s; x, a)}{dx} &= \frac{1}{(1 + \exp(s(a - x)))^2} \left\{ 1 - (sa - sx - 1) \exp(s(a - x)) \right\}; \\
\frac{d S_m(s; x, a)}{dx} &= \frac{1}{(1 + \exp(-s(a - x)))^2} \left\{ 1 + (sa - sx + 1) \exp(-s(a - x)) \right\}.
\end{align*}
\]

Therefore, \( x > a, \Rightarrow \lim_{s \rightarrow +\infty} (d L_a(s; x, a)/dx) = 1, \lim_{s \rightarrow +\infty} (d S_m(s; x, a)/dx) = 0; \) and \( x < a, \Rightarrow \lim_{s \rightarrow +\infty} (d L_a(s; x, a)/dx) = 0; \lim_{s \rightarrow +\infty} (d S_m(s; x, a)/dx) = 1. \) So for a given constant \( a \in [0, 1], \) it follows that

\[
\lim_{s \rightarrow +\infty} \frac{d L_a(s; x, a)}{dx} = \begin{cases} 
\frac{d (a \lor x)}{dx}, & x \neq a, \\
\frac{1}{2}, & x = a
\end{cases}
\]

\[
\lim_{s \rightarrow +\infty} \frac{d S_m(s; x, a)}{dx} = \begin{cases} 
\frac{d (a \land x)}{dx}, & x \neq a, \\
\frac{1}{2}, & x = a
\end{cases}
\]
2.3.2 BP learning algorithm

To develop the BP learning algorithm for the connection weight matrix $W$ of (2.1), firstly we define a suitable error function. Suppose $\{(x_k, y_k)|k \in P\}$ is a fuzzy pattern pair family for training. And for the input pattern $x_k$ of (2.1), let the corresponding real output pattern be $o_k = (o^k_1, \ldots, o^k_m)$: $o_k = x_k \circ W$, that is

$$o^k_j = \bigvee_{i \in N} \{ x^k_i \land w_{ij} \} \ (k \in P, j \in M).$$

Define the error function $E(W)$ as follows:

$$E(W) = \frac{1}{2} \sum_{k=1}^{p} \| o_k - y_k \|^2 = \frac{1}{2} \sum_{k=1}^{p} \sum_{j=1}^{m} (o^k_j - y^k_j)^2. \quad (2.32)$$

Since $E(W)$ is non-differentiable with respect to $w_{ij}$, we cannot design the BP algorithm directly using $E(W)$. So we utilize the functions $La$ and $Sm$ to replace the fuzzy operators $\vee$ and $\land$, respectively. By Lemma 2.3, when $s$ is sufficiently large, we have

$$E(W) \approx e(W) \triangleq \frac{1}{2} \sum_{k=1}^{p} \sum_{j=1}^{m} (La(s; Sm(s; x^k_1, w_{ij}), \ldots, Sm(s; x^k_n, w_{nj}))-y^k_j)^2. \quad (2.33)$$

$e(W)$ is a differentiable function, so we can employ the partial derivative $\partial e(W)/\partial w_{ij}$ to develop a BP algorithm of (2.1).

**Theorem 2.12** Give the fuzzy pattern pair family $\{(x_k, y_k)|k \in P\}$. Then $e(W)$ is continuously differentiable with respect to $w_{ij}$ for $i \in N$, $j \in M$. And

$$\frac{\partial e(W)}{\partial w_{ij}} = \sum_{k=1}^{p} \frac{-\exp(s \cdot \triangle(i, k)) \Gamma(s)}{(\sum_{p=1}^{d} \exp(s \cdot \Delta(p, k)))^2} \cdot \frac{1 + (sx^k_i - sw_{ij} + 1) \exp(-s(x^k_i - w_{ij}))}{(1 + \exp(-s(x^k_i - w_{ij})))^2}.$$

where $\Gamma(s) = \sum_{p=1}^{d} \{ s \cdot \exp(s \cdot \Delta(p, k)) - \exp(s \cdot \Delta(i, k)) - 1 \} \exp(s \cdot \Delta(p, k))$, and $\triangle(i, k) = Sm(s; x^k_i, w_{ij})$.

**Proof.** By Lemma 2.4, considering $\triangle(i, k) = Sm(s; x^k_i, w_{ij})$ for $i \in N$, $j \in M$, we have

$$\frac{\partial La(s; Sm(s; x^k_1, w_{ij}), \ldots, Sm(s; x^k_n, w_{nj}))}{\partial Sm(s; x^k_i, w_{ij})} = -\frac{\exp(s \cdot \triangle(i, k)) \cdot \Gamma(s)}{(\sum_{p=1}^{d} \exp(s \cdot \Delta(p, k)))^2}.$$

And by (2.29) easily we can show

$$\frac{\partial Sm(s; x^k_i, w_{ij})}{\partial w_{ij}} = \frac{1 + (sx^k_i - sw_{ij} + 1) \exp(-s(x^k_i - w_{ij}))}{(1 + \exp(-s(x^k_i - w_{ij})))^2}.$$
By \( \frac{\partial e(W)}{\partial w_{ij}} = \sum_{k=1}^{p} \left( \frac{\partial e(W)}{\partial S_m(s; x_i^k, w_{ij})} \cdot (\frac{\partial S_m(s; x_i^k, w_{ij})}{\partial w_{ij}}) \right) \), it follows that
\[
\frac{\partial e(W)}{\partial w_{ij}} = \sum_{k=1}^{p} -\exp(s \cdot \Delta(i, k)) \Gamma(s) \cdot \frac{1+ (sx_i^k - sw_{ij} + 1) \exp(-sx_i^k - w_{ij}))}{(1 + \exp(-sx_i^k - w_{ij}))^2} .
\]

Therefore, \( e(W) \) is continuously differentiable with respect to \( w_{ij} \). \( \square \)

Using the partial derivatives in Theorem 2.12 we can design a BP algorithm for \( W \) of (2.1).

**Algorithm 2.3** BP learning algorithm of FAM's.

1. **Initialization.** Put \( w_{ij}(0) = 0 \), and let \( W(0) = (w_{ij}(0))_{n \times m} \), set \( t = 1 \).
2. **Denote** \( W(t) = (w_{ij}(t))_{n \times m} \).
3. **Iteration scheme.** \( W(t) \) iterates with the following law:
   \[
   \Omega = w_{ij}(t) - \delta \cdot \frac{\partial e(W(t))}{\partial w_{ij}(t)} + \alpha \cdot \Delta w_{ij}(t), \quad w_{ij}(t + 1) = (\Omega \lor 0) \land 1.
   \]
4. **Stop condition.** Discriminate \(|e(W(t + 1))| < \varepsilon\)? If yes, output \( w_{ij}(t + 1) \); otherwise, let \( t = t + 1 \) go to Step 2.

In the following we illustrate Algorithm 2.3 by a simulation to train FAM (2.1). To this end, Give a fuzzy pattern pair family as shown in Table 2.3.

<table>
<thead>
<tr>
<th>No.</th>
<th>Input pattern</th>
<th>Desired output</th>
<th>Real pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0.64, 0.50, 0.70, 0.60)</td>
<td>(0.64, 0.70)</td>
<td>(0.6400, 0.7000)</td>
</tr>
<tr>
<td>2</td>
<td>(0.40, 0.45, 0.80, 0.65)</td>
<td>(0.65, 0.80)</td>
<td>(0.6500, 0.7867)</td>
</tr>
<tr>
<td>3</td>
<td>(0.75, 0.70, 0.35, 0.25)</td>
<td>(0.75, 0.50)</td>
<td>(0.7250, 0.5325)</td>
</tr>
<tr>
<td>4</td>
<td>(0.33, 0.67, 0.35, 0.50)</td>
<td>(0.67, 0.50)</td>
<td>(0.6700, 0.5000)</td>
</tr>
<tr>
<td>5</td>
<td>(0.65, 0.70, 0.90, 0.75)</td>
<td>(0.75, 0.80)</td>
<td>(0.7500, 0.7867)</td>
</tr>
<tr>
<td>6</td>
<td>(0.95, 0.30, 0.45, 0.60)</td>
<td>(0.80, 0.60)</td>
<td>(0.7250, 0.6000)</td>
</tr>
<tr>
<td>7</td>
<td>(0.80, 1.00, 0.85, 0.70)</td>
<td>(0.80, 0.80)</td>
<td>(0.7864, 0.7867)</td>
</tr>
<tr>
<td>8</td>
<td>(0.10, 0.50, 0.70, 0.65)</td>
<td>(0.65, 0.70)</td>
<td>(0.6500, 0.7000)</td>
</tr>
<tr>
<td>9</td>
<td>(0.70, 0.70, 0.25, 0.56)</td>
<td>(0.70, 0.56)</td>
<td>(0.7000, 0.5600)</td>
</tr>
</tbody>
</table>

Choose \( \alpha = 0.05, \eta = 0.3 \). Let \( s = 100 \). With 1000 iterations, by Algorithm 2.3 we can establish the real output of (2.1), as shown Table 2.3. By comparison we know, Algorithm 2.3 possesses a quicker convergent speed and higher convergent accuracy.
The further subjects for FAM's include designing the learning algorithms related based on GA [5, 49], analysis on fault–tolerance of systems [22, 33–35, 37, 38], and applying the results obtained to many real fields, such as, signal processing [42], system modeling and identification [21, 44, 50], system control [31, 32] and so on. These researches are at their infancy, and so they have a great prospect for the future research.

§2.4 Fuzzy ART and fuzzy ARTMAP

Through the learning of a FAM, a given family of fuzzy patterns may be stored in the FAM, and the connection weight matrix $W$ is established. If a new fuzzy pattern is presented to the FAM and asked to be stored in $W$, the FAM has to be trained to violate the original $W$. Thus, FAM’s as competitive networks do not have stable learning in response to arbitrary input patterns. The learning instability occurs because of the network’s adaptivity, which causes prior learning to be eroded by more recent learning. How can a system be receptive to significant new patterns and yet remain stable in response to irrelevant patterns? Adaptive resonance theory (ART) developed by Carpenter et al addresses such a dilemma [6]. As each input pattern is presented to ART, it is compared with the prototype vector that it most closely matches. If the match between the prototype and the input vector is no adequate, a prototype is selected. In this way previously learned memories are nor eroded by new learning. ART1 can process patterns expressed as vectors whose components are either 0 or 1. Fuzzy ART is a fuzzy version of ART1 [7], so let us now recall ART1 and its architecture.

2.4.1 ART1 architecture

An ART1 network consists of five parts, two subsystems which are called the attentional subsystem $C$ (comparing layer) and the orienting subsystem $R$ (recognition layer), respectively, and three controllers, two gain controllers $G_1$ and $G_2$, which generate the controlling signals $G_1$, $G_2$, respectively. and reset controller 'Reset'. The five components act together to form an efficient pattern classifying model. The ART1 has an architecture as shown in Figure 2.3.

![Figure 2.3 Architecture of ART1](image)

![Figure 2.4 Attentional subsystem](image)
Let us now describe the respective functions of five parts of ART1 in Figure 2.3. When an input \( \mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n \) is presented the network, the gain controller \( G_2 \) tests whether it is 0, and the corresponding controlling signal \( G_2 = x_1 \land \cdots \land x_n \), that is, \( \forall i \in \{1,\ldots,n\}, x_i = 0, \implies G_2 = 0 \), otherwise \( G_2 = 1 \). Suppose the output of the recognition layer \( R \) is \( \mathbf{r} = (r_1, \ldots, r_m) \), and \( R_0 = r_1 \lor \cdots \lor r_m \). Then the controlling signal \( G_1 \) of the gain controller \( G_1 \) is the product of \( G_2 \) and the complementary of \( R_0 \), that is, \( G_1 = G_2 \cdot (1 - R_0) = G_2 \cdot R_0^c \). Therefore, \( \mathbf{r} = 0, \mathbf{x} \neq 0, \implies G_1 = 1 \); otherwise \( G_1 = 0 \). The reset controller ‘Reset’ makes the winning neuron in competition in layer \( R \) lose efficacy.

\[
P_j = \sum_{i=1}^{n} b_{ij} \cdot x_i \quad (j = 1, \ldots, m),
\]

We call \( P_j \) the matching degree between \( \mathbf{x} \) and \( \mathbf{b}_j = (b_{1j}, \ldots, b_{nj}) \). Choose such
a node \( j^* \), whose matching degree is maximum, that is, \( P_{j^*} = \bigvee_{1 \leq j \leq m} \{P_j\} \). The node is called a winning node, and so \( r_{j^*} = 1, r_j = 0 \) (\( j \neq j^* \)).

**Second step—comparing.** The output vector \( r = (r_1, ..., r_m) \) of \( R \) return to \( C \) through the connection weight matrix \( T = (t_{ij})_{n \times m} \). The fact that \( r_{j^*} = 1 \) results in the weight vector \( t_{j^*} = (t_{1j^*}, ..., t_{nj^*}) \) being active, and others being inactive. Thus, \( R_0 = 1, \implies G_1 = G_2 \cdot R_0^c = 0 \). The output \( c = (c_1, ..., c_n) \) of \( C \) characterizes the matching degree \( M_0 \) between \( t_{j^*} \) and the input \( x \):

\[
M_0 = \langle x, t_{j^*}^T \rangle = \sum_{i=1}^{n} t_{ij^*} \cdot x_i = \sum_{i=1}^{n} c_i.
\]

Since \( x_i \in \{0,1\} \), \( M_0 \) is the number of overlapping nonzero components between \( t_{j^*} \) and \( x \). Suppose there exist \( M_1 \) nonzero components in \( x \), i.e. \( M_1 = x_1 + \cdots + x_n \). \( M_0/M_1 \) also reflect the similarity between \( x \) and \( t_{j^*} \).

Give \( \rho \in [0, 1] \) as a minimum similarity vigilance of the input pattern \( x \) and the template \( t_j \) corresponding to a winning node. If \( M_0/M_1 < \rho \), then \( x \) and \( t_j \) can not satisfy the similarity condition, and through the reset signal ‘Reset’ let the match finished in the first step lose its efficacy, and the winning node become invalid. Go to third step, searching; If \( M_0/M_1 > \rho \), then \( x \) and \( t_j \) are close enough, and the ‘resonance’ between \( x \) and \( t_j \) takes place, the match in the first step is effective. Go to fourth step, learning.

**Third step—searching.** Reset signal makes the winning node established by the first step keep restrained, and the restraining state is kept until the ART1 network receives a new pattern. And we have, \( R_0 = 0, G_1 = 1 \). The network returns the matching state in the first step, and go to first step. If the circulating procedure does not stop until all patterns in \( R \) are used, then \((m+1)\)-th node have to be added to store the current pattern as a new template, and let \( t_{i(m+1)} = 1, b_{i(m+1)} = x_i \) \( (i = 1, ..., n) \).

**Fourth step—learning.** \( b_{ij} \) and \( t_{ij} \) iterate according to the following Algorithm 2.4, so that the stronger ‘resonance’ between \( x \) and \( t_{j^*} \) takes places.

**Algorithm 2.4**

The connection weight matrices \( B = (b_{ij})_{n \times m}, T = (t_{ij})_{n \times m} \) iterate with the following steps:

**Step 1.** Initialization: let \( t = 0 \), and choose the initial values of \( b_{ij} \) and \( t_{ij} \):

\[
b_{ij}(0) = \frac{1}{n+1}; \quad t_{ij}(0) = 1 \quad (i = 1, ..., n; \ j = 1, ..., m).
\]

**Step 2.** Receive an input: Give the input pattern \( x = (x_1, ..., x_n) \in \{0, 1\}^n \).

**Step 3.** Determine the winning node \( j^* \): Calculate the matching degree \( P_j \), and compute \( j^*: P_{j^*} = \bigvee_{j=1}^{m} \{P_j\} \).

**Step 4.** Compute the similarity degree. By \( M_0 = \sum_{i=1}^{n} x_i \cdot t_{i,j^*} = \sum_{i=1}^{n} c_i \) we may establish the similarity degree between the vector \( t_{j^*} = (t_{1j^*}, ..., t_{nj^*}) \), corresponding to the winning node and the input pattern \( x \).
Chapter II Fuzzy Neural Networks for Storing and Classifying

Step 5. Vigilance test. Calculate $M_1 = \sum_{i=1}^{n} x_i$. If $M_0/M_1 < \rho$, let the winning node $j^*$ invalid, and put $r_{j^*} = 0$. Go to Step 6; if $M_0/M_1 > \rho$, we classify $x$ into the pattern class that includes $t_{j^*}$, and go to Step 7.

Step 6. Search pattern class. If the winning node number is less than $m$, go to Step 3; If the invalid node number equals to $m$, then in $R$ add the $(m+1)$th node, and let $b_i(m+1) = x_i$, $t_i(m+1) = 1$ ($i = 1, ..., n$). Put $m = m + 1$, go to Step 2.

Step 7. Adjust connection weights. For $i = 1, ..., n$, $t_{ij^*}$ and $b_{ij^*}$ are adjusted with the following scheme:

$$
\begin{align*}
t_{ij^*}(t + 1) &= t_{ij^*} \cdot x_i, \\
b_{ij^*}(t + 1) &= \frac{t_{ij^*}(t) \cdot x_i}{0.5 + \sum_{i'=1}^{n} t_{i'j^*}(t) \cdot x_{i'}} = \frac{t_{ij^*}(t + 1)}{0.5 + \sum_{i'=1}^{n} t_{i'j^*}(t + 1)}.
\end{align*}
$$

Set $t = t + 1$, go to Step 2.

Algorithm 2.4 for the ART1 network is a on-line learning. In the ART1 network, the nodes stand for the classified pattern classes respectively, each of which includes some similar patterns. By the vigilance $\rho$ we can establish the number of the patterns classified. The larger $\rho$ is, the more the classified patterns are.

2.4.2 Fuzzy ART

Like an ART1 network, a fuzzy ART consist also of two subsystems [7, 13, 14], one is the attentional subsystem, and another is the orienting subsystem, as shown in Figure 2.6. The attentional subsystem is a two-layer network architecture, in which $F_1^X$ is the input layer, accepting the input fuzzy patterns, and $F_2^X$ is a pattern expressing layer. All fuzzy patterns in $F_2^X$ constitute a classification of the input patterns. Orienting subsystem consists of a reset node 'Reset', which accepts all information coming from $F_1^X$ layer, $F_2^X$ layer and $F_0^X$ layer using for transforming the input patterns.

By $F_0^X$ layer we can also complete the complement code of the input fuzzy pattern $x$, that is, the output $I$ of $F_0^X$ is determined as follows:

$$
I = (x, x^c) = (x_1, ..., x_n, x_1^c, ..., x_n^c) = (x_1, ..., x_n, 1 - x_1, ..., 1 - x_n),
$$

where $x_i \in [0, 1]$ ($i = 1, ..., n$). From now on, we take the pattern $I$ as an input of a fuzzy ART. Thus, $F_1^X$ includes $2n$ nodes, and suppose $F_2^X$ includes $m$ nodes. Let the connection weight between the node $i$ in $F_1^X$ and the node $j$ in $F_2^X$ be $W_{ij}^X$, and the connection weight between the node $j$ in $F_2^X$ and the node $i$ in $F_1^X$ be $w_{ji}^X$. Denote

$$
W_j^X = (W_{1j}^X, ..., W_{(2n)j}^X), \quad w_j^X = (w_{j1}^X, ..., w_{j(2n)}^X),
$$
We call $w_j^x$ a template, where $j = 1, \ldots, m$. Suppose the initial values of the connection weights $W_{ij}^x$ and $w_j^x$ are $W_{ij}^x(0), w_j^x(0)$, respectively:

$$W_{ij}^x(0) = \frac{1}{\alpha_x + M_x}, \quad w_j^x(0) = 1 \quad (i = 1, \ldots, 2n; \quad j = 1, \ldots, m),$$

where $\alpha_x, M_x$ are parameters of the fuzzy ART, $\alpha_x \in (0, +\infty)$ is a selection parameter, and $M_x \in [2n, +\infty)$ is a uncommitted node parameter [14]. Denote $W_j^x(0) = (W_{1j}^x(0), \ldots, W_{(2n)j}^x(0)), \quad w_j^x(0) = (w_{1j}^x(0), \ldots, w_{j(2n)}^x(0))$. Then $W_j^x(0), w_j^x(0)$ correspond, respectively to the $j$-th connection weight vectors before the input fuzzy pattern $x$ is expressed in $F_2^x$.

Before discussing the I/O relationship of the fuzzy ART, we introduce some notations. Suppose $y_q = (y_1^q, \ldots, y_{2n}^q) \in [0, 1]^{2n} (q = 1, 2)$, denote $|y^1| = \sum_{i=1}^{2n} y_i^1$, and we call $\text{dis}(y_1, y_2) = \sum_{i=1}^{2n} |y_i^1 - y_i^2|$ the metric between the fuzzy patterns $y_1$ and $y_2$. Denote

$$y_1 \vee y_2 = (y_1^1 \vee y_2^1, \ldots, y_{2n}^1 \vee y_{2n}^1); \quad y_1 \wedge y_2 = (y_1^1 \wedge y_1^2, \ldots, y_{2n}^1 \wedge y_{2n}^2).$$

We call the node $j$ a uncommitted node in $F_2^x$, if $w_j^x = w_j^x(0) = (1, \ldots, 1)$, otherwise the node $j$ is called a committed node. For $j \in \{1, \ldots, m\}$, define

$$\begin{align*}
W_j^{x, \text{old}} &= W_j^x, \quad \text{I is not classified into } F_2^x, \\
W_j^{x, \text{new}} &= W_j^x, \quad \text{I is classified into } F_2^x,
\end{align*}$$

$$\begin{align*}
w_j^{x, \text{old}} &= w_j^x, \quad \text{I is not classified into } F_2^x, \\
w_j^{x, \text{new}} &= w_j^x, \quad \text{I is classified into } F_2^x.
\end{align*}$$
For the input $I$ of $F^x$, by the upward connection weight matrix $(W^x_1, ..., W^x_{2n})^T$, we can establish an input of $F^x_2$ layer as $t(I) = (t_1(I), ..., t_m(I))$:

$$t_j(I) = \begin{cases} \frac{|I|}{\alpha_x + M_x}, & \text{node } j \text{ is an uncommitted node}, \\ \frac{|I \land w^x_{j,old}|}{\alpha_x + |w^x_{j,old}|}, & \text{node } j \text{ is a committed node}, \end{cases}$$ (2.34)

where $j \in \{1, ..., m\}$. Let $j^* \in \{1, ..., m\} : t^{*\text{th}}(I) = \bigvee_{j=1}^m \{t_j(I)\}$. And in $F^x_2$ only the node $j^*$ is active, and others are inactive. $w^x_{j^*}$ may be taken as a candidacy of a standard pattern class, into which the input pattern $I$ will be classified. The matching degree between $w^x_{j^*}$ and $I$ is $A^x = (|I \land w^x_{j^*}|)/|I|$. For a given vigilance $\rho \in [0,1]$, if $A^x > \rho$, then $I$ is classified into $w^x_{j^*}$. Similarly with Algorithm 2.4, the connection weight vectors can be trained as follows:

$$\begin{align*}
w^x_{j^*,\text{new}} &= I \land w^x_{j^*,\text{old}}, \\
w^x_{j^*,\text{new}} &= \frac{I \land w^x_{j^*,\text{old}}}{\alpha_x + |I \land w^x_{j^*,\text{old}}|} = \frac{w^x_{j^*,\text{new}}}{\alpha_x + |w^x_{j^*,\text{new}}|}.
\end{align*}$$

If $\lambda_x < \rho$, then the reset node ‘Reset’ in the orienting subsystem generates a signal to enable the node $j^*$ inhibitory. And we choose a second-maximum $t_j(I)$ as the new active node $j^*$. Repeat this procedure. If one by one each $j \in \{1, ..., m\}$ is taken as $j^*$, and it can not ensure $A^x > \rho$, then $I$ is stored in $F^x_2$ as a representative of a new fuzzy pattern class, and in $F^x_2$ we add a new node according to $I$.

Next let us show the geometry sense of fuzzy pattern in the fuzzy ART [14]. For $j \in M$, the weight vector $w^x_j$ can be expressed by two $n$ dimensional vectors $u^x_j$ and $v^x_j$:

$$w^x_j = (u^x_j, \{v^x_j\}^c).$$

where $u^x_j \leq v^x_j$. If let $n = 2$, then $w^x_j$ can be established by two vertices $u^x_j, v^x_j$ of the rectangle $R^x_j$, as shown in Figure 2.7. Give an input pattern $I^x = (x, x^c)$, at first it is not classified by the fuzzy ART. The rectangle $R^x_j$ is a geometry representation of the template vector $w^x_{j,old}$. If $I^x \in R^x_{j,old}$, then $u^x_{j,old} \leq x \leq v^x_{j,old}$, and the the template vector keeps unchanged, that is

$$w^x_{j,\text{new}} = w^x_{j,\text{old}} \land I^x = (u^x_{j,\text{old}} \land x, \{v^x_{j,\text{old}} \lor x\}^c) = w^x_{j,\text{old}}.$$

If $I^x \not\in R^x_{j,old}$, then easily we can show, $w^x_{j,\text{new}} \neq w^x_{j,\text{old}}$. And the weight vectors change, and the number of the rectangles increases, correspondingly. Its maximum value is determined by the vigilance $\rho$. For the input fuzzy pattern $I^x$, if $|I^x| = M$, and

$$|I^x \land w^x_{j,\text{old}}| \geq M \cdot \rho,$$ (2.35)
then in $F^x_2$ layer, $I^x$ can be classified into a pattern class that includes the weight vector $w^x_{j,old}$. By computation we can see

$$
|I^x \land w^x_{j,old}| = |(u^x_{j,old} \land x, \{v^x_{j,old} \lor x\})| = \sum_{i=1}^{n} (x_i \land u^x_{ij}) + \sum_{i=1}^{n} (a_i \lor v^x_{ij})^c
$$

(2.36)

$$
= n - |(x \lor v^x_{j,old}) - (x \land u^x_{j,old})| = n - |R^x_{j,new}|
$$

So by (2.35) (2.36) it follows that $|R^x_{j,new}| \leq n(1 - \rho)$. Therefore, if $\rho$ is very small, i.e. $\rho \approx 0$, then the input space is filled with small rectangles; If $\rho \approx 1$, then there are only a few of rectangles.

Now we present the classifying order of the input pattern $I = (x, x^c)$ by the fuzzy ART when choosing the parameter $\alpha_x$ as 'very small' 'medium' and 'very large', respectively. To this end we at first give three lemmas.

**Lemma 2.5** Let $I = (x, x^c)$ be an input fuzzy pattern of the fuzzy ART, and $I \in R^x_{j,1} \cap R^x_{j,2}$, $|R^x_{j,1}| < |R^x_{j,2}|$. Then the rectangle $R^x_{j,1}$ is chosen precede $R^x_{j,2}$, that is, $t_{j,1}(I) > t_{j,2}(I)$.

**Proof.** By the assumption we have, $j_1, j_2$ are committed nodes in $F^x_2$ layer. So by (2.34) it follows that

$$
\begin{align*}
\begin{cases}
t_{j,1}(I) = \frac{|I \land w^x_{j,1}|}{\alpha_x + |w^x_{j,1}|} = \frac{|w^x_{j,1}|}{\alpha_x + |w^x_{j,1}|} = \frac{n - |R^x_{j,1}|}{\alpha_x + n - |R^x_{j,1}|};
\end{cases}
\end{align*}
$$

(2.37)

$$
\begin{align*}
\begin{cases}
t_{j,2}(I) = \frac{|I \land w^x_{j,2}|}{\alpha_x + |w^x_{j,2}|} = \frac{|w^x_{j,2}|}{\alpha_x + |w^x_{j,2}|} = \frac{n - |R^x_{j,2}|}{\alpha_x + n - |R^x_{j,2}|}.
\end{cases}
\end{align*}
$$

Using the assumption we get, $n - |R^x_{j,1}| > n - |R^x_{j,2}|$. Hence (2.37) implies that $t_{j,1}(I) > t_{j,2}(I)$. Thus, by the classifying rule of the fuzzy ART, the rectangle $R^x_{j,1}$ is chosen precede $R^x_{j,2}$. □

**Lemma 2.6** Suppose an input fuzzy pattern $I = (x, x^c)$ is presented to the fuzzy ART, and $I \in R^x_{j,2} \setminus R^x_{j,1}$. Then the rectangle $R^x_{j,1}$ is chosen precede $R^x_{j,2}$ if and only if

$$
\text{dis}(I, R^x_{j,1}) < (n + \alpha_x - |R^x_{j,1}|)\left\{\frac{n - |R^x_{j,1}|}{\alpha_x + n - |R^x_{j,1}|} - \frac{n - |R^x_{j,2}|}{\alpha_x + n - |R^x_{j,2}|}\right\}.
$$

**Proof.** By the assumption and (2.34) easily we can show

$$
\begin{align*}
\begin{cases}
t_{j,1}(I) = \frac{|I \land w^x_{j,1}|}{\alpha_x + |w^x_{j,1}|} = \frac{|w^x_{j,1}|}{\alpha_x + |w^x_{j,1}|} = \frac{n - |R^x_{j,1}|}{\alpha_x + n - |R^x_{j,1}|};
\end{cases}
\end{align*}
$$

(2.38)

$$
\begin{align*}
\begin{cases}
t_{j,2}(I) = \frac{|I \land w^x_{j,2}|}{\alpha_x + |w^x_{j,2}|} = \frac{|w^x_{j,2}|}{\alpha_x + |w^x_{j,2}|} = \frac{n - |R^x_{j,2}|}{\alpha_x + n - |R^x_{j,2}|}.
\end{cases}
\end{align*}
$$
Then
\[ t_{j_1}(\mathbf{I}) > t_{j_2}(\mathbf{I}) \iff \frac{n - |R_{j_1}^{x,\text{new}}|}{\alpha_x + n - |R_{j_1}^{x,\text{old}}|} > \frac{n - |R_{j_2}^{x,\text{old}}|}{\alpha_x + n - |R_{j_2}^{x,\text{old}}|}. \] (2.38)

But \(|R_{j_1}^{x,\text{new}}| = |R_{j_1}^{x,\text{old}}| + \text{dis}(\mathbf{I}, R_{j_1}^{x,\text{old}})|\), which is replace into (2.38) we get,
\[ t_{j_1}(\mathbf{I}) > t_{j_2}(\mathbf{I}) \] if and only if the following fact holds:
\[ \text{dis}(\mathbf{I}, R_{j_1}^{x,\text{old}}) < (n + \alpha_x - |R_{j_1}^{x,\text{old}}|) \left\{ \frac{n - |R_{j_1}^{x,\text{old}}|}{\alpha_x + n - |R_{j_1}^{x,\text{old}}|} - \frac{n - |R_{j_2}^{x,\text{old}}|}{\alpha_x + n - |R_{j_2}^{x,\text{old}}|} \right\}, \]
which implies the lemma. □

**Lemma 2.7** Suppose \( \mathbf{I} = (x, x^c) \) is an input pattern presented to the fuzzy ART, and \( \mathbf{I} \in R_{j_1}^{x,\text{old}} \cap R_{j_2}^{x,\text{old}} \). Then the rectangle \( R_{j_1}^{x,\text{old}} \) is chosen precede \( R_{j_2}^{x,\text{old}} \) if and only if
\[ \text{dis}(\mathbf{I}, R_{j_1}^{x,\text{old}}) < \text{dis}(\mathbf{I}, R_{j_2}^{x,\text{old}}) \cdot \frac{n + \alpha_x - |R_{j_1}^{x,\text{old}}|}{\alpha_x + n - |R_{j_1}^{x,\text{old}}|} + \alpha_x (|R_{j_1}^{x,\text{old}}| - |R_{j_2}^{x,\text{old}}|) \]
\[ \text{dis}(\mathbf{I}, R_{j_2}^{x,\text{old}}) < \text{dis}(\mathbf{I}, R_{j_2}^{x,\text{old}}) \cdot \frac{n + \alpha_x - |R_{j_1}^{x,\text{old}}|}{\alpha_x + n - |R_{j_1}^{x,\text{old}}|} + \alpha_x (|R_{j_1}^{x,\text{old}}| - |R_{j_2}^{x,\text{old}}|) \].

**Proof.** Using the assumption and (2.34) we can conclude that
\[ t_{j_1}(\mathbf{I}) = \frac{|\mathbf{I} \wedge w_{j_1}^{x,\text{old}}|}{\alpha_x + |w_{j_1}^{x,\text{old}}|} = \frac{|w_{j_1}^{x,\text{new}}|}{\alpha_x + |w_{j_1}^{x,\text{old}}|} = \frac{n - |R_{j_1}^{x,\text{new}}|}{\alpha_x + n - |R_{j_1}^{x,\text{old}}|}. \]
\[ t_{j_1}(\mathbf{I}) = \frac{|\mathbf{I} \wedge w_{j_2}^{x,\text{old}}|}{\alpha_x + |w_{j_2}^{x,\text{old}}|} = \frac{|w_{j_1}^{x,\text{new}}|}{\alpha_x + |w_{j_2}^{x,\text{old}}|} = \frac{n - |R_{j_2}^{x,\text{new}}|}{\alpha_x + n - |R_{j_2}^{x,\text{old}}|}. \]
Therefore the following fact holds:
\[ t_{j_1}(\mathbf{I}) > t_{j_2}(\mathbf{I}) \iff \frac{n - |R_{j_1}^{x,\text{new}}|}{\alpha_x + n - |R_{j_1}^{x,\text{old}}|} > \frac{n - |R_{j_2}^{x,\text{new}}|}{\alpha_x + n - |R_{j_2}^{x,\text{old}}|}. \] (2.40)

But for \( k = 1, 2, |R_{j_1}^{x,\text{new}}| = |R_{j_2}^{x,\text{old}}| + \text{dis}(\mathbf{I}, R_{j_2}^{x,\text{old}}) \). Consequently by (2.40) it follows that \( t_{j_1}(\mathbf{I}) > t_{j_2}(\mathbf{I}) \) if and only if (2.39) is true. □

In order to utilize the parameter \( \alpha_x \) to establish the classifying order of the input fuzzy pattern \( \mathbf{I} \) by the fuzzy ART, we define the function
\[
\begin{align*}
\phi_0(x) &= (n + x - |R_{j_1}^{x,\text{old}}|) \left\{ \frac{n - |R_{j_1}^{x,\text{old}}|}{x + n - |R_{j_1}^{x,\text{old}}|} - \frac{n - |R_{j_2}^{x,\text{old}}|}{x + n - |R_{j_2}^{x,\text{old}}|} \right\}; \\
\phi_1(x) &= \text{dis}(\mathbf{I}, R_{j_2}^{x,\text{old}}) \cdot \frac{n + x - |R_{j_1}^{x,\text{old}}|}{x + n - |R_{j_1}^{x,\text{old}}|} + x \left( \frac{|R_{j_2}^{x,\text{old}}| - |R_{j_1}^{x,\text{old}}|}{x + n - |R_{j_2}^{x,\text{old}}|} \right). 
\end{align*}
\]
Theorem 2.13 Suppose an input pattern \( I = (x, x^c) \) is presented to the fuzzy ART, and the parameter \( \alpha_x \approx 0 \). Then

(i) If \( I \in R_{j1}^{x,\text{old}} \cap R_{j2}^{x,\text{old}} \), then \( I \) chooses first the rectangle between \( |R_{j1}^{x,\text{old}}| \) and \( |R_{j2}^{x,\text{old}}| \) with smaller size;

(ii) If \( I \in R_{j2}^{x,\text{old}} \setminus R_{j1}^{x,\text{old}} \), then \( I \) chooses first \( R_{j1}^{x,\text{old}} \);

(iii) If \( I \notin R_{j1}^{x,\text{old}} \cup R_{j2}^{x,\text{old}} \), then the pattern \( I \) chooses first \( R_{j1}^{x,\text{old}} \) if and only if

\[
\text{dis}(I, R_{j1}^{x,\text{old}})(n - |R_{j2}^{x,\text{old}}|) < \text{dis}(I, R_{j2}^{x,\text{old}})(n - |R_{j1}^{x,\text{old}}|).
\]

Proof. (i) is a direct corollary of Lemma 2.5, so it suffices to show (ii) and (iii).

(ii) Since \( \lim_{x \to 0} \phi_0(x) = 0 \), when \( \alpha_x \approx 0 \), \( \phi_0(\alpha_x) \approx 0 \). Therefore, \( \phi_0(\alpha_x) \leq \text{dis}(M, I, R_{j1}^{x,\text{old}}) \). Then by Lemma 2.6 it follows that \( I \) chooses the rectangle \( R_{j2}^{x,\text{old}} \) firstly.

(iii) Obviously, the following fact holds:

\[
\lim_{x \to 0} \phi_1(x) = \phi_1(0) = \frac{(\text{dis}(I, R_{j2}^{x,\text{old}})(n - |R_{j2}^{x,\text{old}}|))}{n - |R_{j2}^{x,\text{old}}|}.
\]

And when \( \alpha_x \approx 0 \), we have, \( \text{dis}(I, R_{j1}^{x,\text{old}}) < \phi_1(\alpha_x) \iff \text{dis}(I, R_{j1}^{x,\text{old}}) < \phi_1(0) \).

So by Lemma 2.7, \( I \) will first choose \( R_{j1}^{x,\text{old}} \) if and only if \( \text{dis}(I, R_{j1}^{x,\text{old}}) < \phi_1(0) \), that is,

\[
\text{dis}(I, R_{j1}^{x,\text{old}})(n - |R_{j2}^{x,\text{old}}|) < \text{dis}(I, R_{j2}^{x,\text{old}})(n - |R_{j1}^{x,\text{old}}|). \]

Next let us proceed to discuss the choosing order of the input fuzzy pattern by the fuzzy ART when the parameter \( \alpha_x \) is ‘medium’ or ‘sufficiently large’. By Lemma 2.7 and Theorem 2.13, if \( I \in R_{j1}^{x,\text{old}} \cap R_{j2}^{x,\text{old}} \), the conclusion (i) of Theorem 2.13 holds when \( \alpha_x \) is ‘medium’ or ‘sufficiently large’. So it suffices to solve our problem for \( I \in R_{j2}^{x,\text{old}} \setminus R_{j1}^{x,\text{old}} \) and \( I \notin R_{j1}^{x,\text{old}} \cup R_{j2}^{x,\text{old}} \).

Theorem 2.14 Suppose the fuzzy pattern \( I = (x, x^c) \) is presented to the fuzzy ART, and \( \alpha_x \) is ‘medium’, i.e. \( 0 < \alpha_x < +\infty \), moreover, \( |R_{j1}^{x,\text{old}}| < |R_{j2}^{x,\text{old}}| \). Then

(i) If \( I \in R_{j2}^{x,\text{old}} \setminus R_{j1}^{x,\text{old}} \), we have, \( I \) will first choose \( R_{j1}^{x,\text{old}} \) if and only if

\[
\text{dis}(I, R_{j1}^{x,\text{old}}) < \phi_0(\alpha_x).
\]

Moreover, the function \( \phi_0(\cdot) \) is nondecreasing on \( \mathbb{R} \), and

\[
0 < \phi_0(\alpha_x) < |R_{j2}^{x,\text{old}}| - |R_{j1}^{x,\text{old}}|.
\]

(ii) If \( I \notin R_{j2}^{x,\text{old}} \cup R_{j1}^{x,\text{old}} \), then \( I \) will first choose \( R_{j1}^{x,\text{old}} \) if and only if

\[
\text{dis}(I, R_{j1}^{x,\text{old}}) < \phi_1(\alpha_x).
\]
And the function $\phi_1(\cdot)$ is nondecreasing on $\mathbb{R}$, moreover

$$\text{dis}(I, R_{j_2}^{x_{\text{old}}}) \cdot \frac{n - |R_{j_1}^{x_{\text{old}}}|}{n - |R_{j_2}^{x_{\text{old}}}|} < \phi_1(\alpha_x) < \text{dis}(I, R_{j_2}^{x_{\text{old}}}) + |R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|. \tag{2.44}$$

Proof. (i) At first, by Lemma 2.6, $I$ will first choose $R_{j_2}^{x_{\text{old}}}$ if and only if (2.41) is true. And we can conclude by computation that

$$\phi_0(x) = \frac{d\phi_0(x)}{dx} = \frac{(n - |R_{j_2}^{x_{\text{old}}}|)(|R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|)}{(x + n - |R_{j_2}^{x_{\text{old}}}|)^2}.$$}

Using the fact $|R_{j_2}^{x_{\text{old}}}| \leq n$, and the assumption we get, $|R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}| > 0$. Then $\phi_0(x) \geq 0$, that is, $\phi_0(\cdot)$ is nondecreasing on $\mathbb{R}$. Moreover, $\phi_0(0) = 0$, $\phi_0(+\infty) \equiv \lim_{x \to +\infty} \phi_0(x) = |R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|$. Therefore by $\alpha_x > 0$, $0 < \phi_0(\alpha_x) < \phi_0(+\infty)$ it follows that (2.42) is true.

(ii) Using Lemma 2.7 we imply the first part of (ii) holds. it follows by computation that

$$\phi_1(x) = \frac{d\phi_1(x)}{dx} = \frac{(n - |R_{j_2}^{x_{\text{new}}}|)(|R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|)}{(x + n - |R_{j_2}^{x_{\text{old}}}|)^2}.$$}

By assumption and the fact $|R_{j_2}^{x_{\text{new}}}| \leq n$ we get, $\phi_1(x) \geq 0$. So $\phi_1(\cdot)$ is nondecreasing on $\mathbb{R}$. Denote $\phi_1(+\infty) \equiv \lim_{x \to +\infty} \phi_1(x)$. It is easy to show

$$\phi_1(0) = \text{dis}(I, R_{j_2}^{x_{\text{old}}}) \cdot \frac{n - |R_{j_1}^{x_{\text{old}}}|}{n - |R_{j_2}^{x_{\text{old}}}|}, \quad \phi_1(+\infty) = \text{dis}(I, R_{j_2}^{x_{\text{old}}}) + |R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|,$$

moreover $\phi_1(0) < \phi_1(\alpha_x) < \phi_1(+\infty)$. Thus, (2.44) is true. □

When $\alpha_x \to +\infty$, we have, $\phi_0(\alpha_x) \to |R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|$, and $\phi_1(\alpha_x) \to \text{dis}(I, R_{j_2}^{x_{\text{old}}}) + |R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|$. Then by Lemma 2.6 and Lemma 2.7 easily we can obtain the following conclusion.

**Theorem 2.15** Suppose the fuzzy pattern $I = (x, x^c)$ is presented to the fuzzy ART. The parameter $\alpha_x \approx +\infty$. Then

(i) If $I \in R_{j_2}^{x_{\text{old}}} \setminus R_{j_1}^{x_{\text{old}}}$, then $I$ will first choose $R_{j_2}^{x_{\text{old}}}$ if and only if $\text{dis}(I, R_{j_1}^{x_{\text{old}}}) < |R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|$.

(ii) If $I \not\in R_{j_2}^{x_{\text{old}}} \cup R_{j_1}^{x_{\text{old}}}$, then $I$ will first choose $R_{j_2}^{x_{\text{old}}}$ if and only if $\text{dis}(I, R_{j_1}^{x_{\text{old}}}) < \text{dis}(I, R_{j_2}^{x_{\text{old}}}) + |R_{j_2}^{x_{\text{old}}}| - |R_{j_1}^{x_{\text{old}}}|$.

In the subsection we discuss how to choose committed nodes in the learning of the fuzzy ART. The main results come from [7, 14]. As for how to choose
the uncommitted nodes by the fuzzy ART some tentative researches are presented in [14], which is also a meaningful problem for future research in the field related. Based on geometric interpretations of the vigilance test and the $F^x_2$ layer competition of committed nodes with uncommitted ones, Anagnostopoulo and Georgioupolos in [1] build a geometric concept related to fuzzy ART, category regions. It is useful for analyzing the learning of fuzzy ART, especially the stability of learning in fuzzy ART. Lin, Lin and Lee utilize the fuzzy ART learning algorithm as a main component to addresses the structure and the associated on-line learning algorithms of a feedforward multilayered network for realizing the basic elements and functions of a traditional fuzzy logic controller [30–32]. The network structure can be constructed from training examples by fuzzy ART learning techniques to find proper fuzzy partitions, membership functions, and fuzzy logic rules. Another important problem related fuzzy ART is how to find the appropriate vigilance range to improve its performance. We may build some robust and invariant pattern recognition models by solving such a problem [24].

2.4.3 Fuzzy ARTMAP

As a supervised learning neural network model the fuzzy ARTMAP consist of three modules, the fuzzy ART$_X$, fuzzy ART$_Y$ and the inter-ART module, shown as in Figure 2.8.

Both fuzzy ART$_X$ and fuzzy ART$_Y$ are the fuzzy ART’s, accepting the inputs $x$, $y$, respectively. The inter-ART module consists of the field $F_{xy}$ and the reset node ‘Reset’. The main propose of $F_{xy}$ is to classify a fuzzy pattern into the given class, or re-begin the matching procedure. For instance, when the fuzzy ART$_Y$ generates a wrong match, by the fuzzy ART$_X$ the vigilance $\rho_X$ increases, so that the maximum matching degree between the resonance fuzzy pattern and the input pattern can achieve.

![Figure 2.8 Fuzzy ARTMAP architecture](image)
As the respective input patterns of fuzzy ART\(X\) and fuzzy ART\(Y\), \(I^X = (x, x^c)\), \(I^Y = (y, y^c)\) are two complement code, where \(x\) is a stimulus fuzzy pattern, and \(y\) is a response fuzzy pattern, which is a prediction of the fuzzy pattern \(I^X\). In the fuzzy ART\(X\), the output of layer \(F^X_f\) is \(a^X = (a^X_1, ..., a^X_{2n^X})\), and \(b^X = (b^X_1, ..., b^X_{m^X})\) is the output of layer \(F^X_y\). Let the \(j\)-th connection weight vector from \(F^X_y\) down to \(F^X_x\) be \(w^X_j = (w^X_{j1}, ..., w^X_{j(2n^X)})\). Similarly, in the fuzzy ART\(Y\), suppose \(a^Y = (a^Y_1, ..., a^Y_{2n^Y})\) and \(b^Y = (b^Y_1, ..., b^Y_{m^Y})\) are the output patterns of \(F^Y_f\) and \(F^Y_y\), respectively. And let \(w^Y_k = (w^Y_{k1}, ..., w^Y_{k(2n^Y)})\) be the \(k\)-th connection weight vector from \(F^Y_y\) down to \(F^Y_x\). Also we suppose \(a^{XY} = (a^{XY}_1, ..., a^{XY}_{m^{XY}})\) is a output pattern of field \(F^{XY}\), and \(w^{XY}_j = (w^{XY}_{j1}, ..., w^{XY}_{j(2m^{XY})})\) is the \(j\)-th connection weight vector of \(F^X_x\) to \(F^{XY}\).

In the inter-ART, field \(F^{XY}\) is called a map field, which accepts the outputs coming from the fuzzy ART\(X\) and fuzzy ART\(Y\). Map field activation is governed by the activity of the fuzzy ART\(X\) and the fuzzy ART\(Y\), in the following way:

\[
\begin{aligned}
\text{If in } F^X_x \text{ the } j^* - \text{th node is active, then its output can be transported to the field } F^{XY} \text{ through the weight vector } w^{XY}_j. \text{ And } w^{XY}_j \text{ may be classified into a defined fuzzy pattern class. If } F^Y_x \text{ is active, then only when a identical fuzzy pattern class is obtained by fuzzy ART}_X \text{ and fuzzy ART}_Y, \text{ respectively } F^{XY} \text{ is active. If a mis-match between } b^Y \text{ and } w^{XY}_j \text{ takes place, } a^{XY} = 0, \text{ and the search procedure is active.}
\end{aligned}
\]

Searching match. When the system accepts an input pattern, the vigilance \(\rho_x\) of ART\(X\) equals to the minimum value \(\bar{p}_x\), and the vigilance of \(F^{XY}\) is \(\rho\). If \(|a^{XY}| < \rho \cdot |b^Y|\), then we increase \(\rho_x\), so that \(\rho_x > |I^X \wedge w^X_{j^*}| \cdot |I^X|^{-1}\), and thus

\[
|a^X| = |I^X \wedge w^X_{j^*}| < \rho_x \cdot |I^X|,
\]

where \(j^*\) means an active node in \(F^X_x\). Thus, through the search procedure of ART\(X\) we can obtain the fact: either there is an active node \(j^*\) in \(F^X_x\), satisfying

\[
\begin{aligned}
\begin{cases}
|a^X| = |I^X \wedge w^X_{j^*}| \geq \rho_x \cdot |I^X|;
|a^{XY}| = |b^Y \wedge w^{XY}_{j^*}| \geq \rho \cdot |b^Y|,
\end{cases}
\end{aligned}
\]

or there is no such a node, then \(F^X_x\) stop the expressing procedure of the input patterns.

Learning of map field. The connection weight \(w^{XY}_{jk}\) of \(F^X_x \rightarrow F^{XY}\) is trained with the following scheme:

**Step 1.** Initialize: \(w^{XY}_{jk}(0) = 1;\)