Data-based neuro-fuzzy systems

Tung, Sau Wai

2012

Tung, S. W. (2012). Data-based neuro-fuzzy systems. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/48902>

<https://doi.org/10.32657/10356/48902>

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

DATA-BASED NEURO-FUZZY SYSTEMS

TUNG SAU WAI

SCHOOL OF COMPUTER ENGINEERING

2012

DATA-BASED NEURO-FUZZY SYSTEMS

TUNG SAU WAI

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2012

Acknowledgments

I would like to take this opportunity to express my gratitude to my main supervisor, Associate Professor Quek Hiok Chai, for his conscientious directions, suggestions, and guidance through the completion of this thesis. His profound knowledge in computational intelligence has greatly inspired me to explore further into this interesting area of research, and I have gained tremendously from him.

I am also grateful to my co-supervisor, Dr. Guan Cuntai, for providing invaluable guidelines and advices to mould my research directions. He has also introduced me to many fascinating and promising applications of brain-computer interface research, which has drawn me to want to investigate further into the boundless potential of the area.

Special thanks to the staff and students from the Centre for Computational Intelligence for their helpful discussions and rewarding interactions which contributed towards the success of this thesis.

I would like to extend my appreciation to Agency for Science, Technology and Research (A*STAR) for funding my research scholarship, giving me an opportunity to participate in this exciting journey of research works. I would also like to extend my appreciation to the School of Computer Engineering, Nanyang Technological University for nurturing me into an independent researcher.

Lastly, I would like to express my heartfelt gratitude to my family for their continuous and endless support in my studies.

Contents

Acknowledgments	ii
Table of Contents	iv
List of Figures	viii
List of Tables	xi
Abstract	xiii
1 Introduction	1
1.1 Motivation	2
1.1.1 Batch neuro-fuzzy systems	3
1.1.2 Evolving neuro-fuzzy systems	5
1.2 Organization of thesis	6
2 Literature Review	8
2.1 Neural networks	8
2.1.1 A neuron	9
2.1.2 Multilayer perceptron	10
2.1.3 Backpropagation	12
2.2 Fuzzy systems	14
2.2.1 Type-1 fuzzy sets and systems	14
2.2.2 Interval Type-2 fuzzy sets and systems	17
2.3 Neuro-fuzzy systems	22

2.3.1	Online versus offline learning	25
2.3.2	Batch neuro-fuzzy systems	27
2.3.3	Evolving neuro-fuzzy systems	32
2.4	Summary	36
3	Self-adaptive Fuzzy Inference Network	39
3.1	SaFIN: Self-adaptive fuzzy inference network	42
3.1.1	Architecture of SaFIN	42
3.1.2	Reasoning process of SaFIN	44
3.2	Learning mechanisms in SaFIN	46
3.2.1	Self-organizing clustering in SaFIN	48
3.2.2	Self-automated rule generation in SaFIN	55
3.3	Experimental results and analysis	57
3.3.1	Example 1 – Identification of a nonlinear system	58
3.3.2	Example 2 – Nakanishi dataset	62
3.3.3	Example 3 – UCI dataset	68
3.4	Summary	72
4	Transportation Case Study: Highway Traffic Modeling Using SaFIN	73
4.1	Motivation	74
4.2	Traffic flow analysis	76
4.2.1	Basic elements in traffic flow analysis	76
4.2.2	Classical mathematical models	78
4.2.3	Intelligent transportation systems	80
4.3	Data collection and experimental setup	82
4.4	Experimental results and analysis	84
4.5	Summary	91

5	Evolving Type-2 Neural Fuzzy Inference System	92
5.1	eT2FIS: Evolving Type-2 neural fuzzy inference system	97
5.1.1	Architecture of eT2FIS	97
5.1.2	Reasoning process of eT2FIS	99
5.1.3	Neural computations of eT2FIS	102
5.2	Learning mechanisms in eT2FIS	105
5.2.1	Initialization	107
5.2.2	Rule generation	107
5.2.3	Merger of fuzzy labels	113
5.2.4	Deletion of obsolete rules	115
5.3	Experimental results and analysis	116
5.3.1	Example 1 – Identification a nonlinear system	116
	Online identification of a non-time-varying system	116
	Online identification of a time-varying system	121
	Identification of a nonlinear noisy system	124
5.3.2	Example 2 – Prediction of a chaotic time-series	128
5.4	Summary	132
6	Financial Case Study: Stock Price Tracking Using eT2FIS	134
6.1	Motivation	135
6.2	Financial time-series analysis	138
6.2.1	Behavioral models	138
6.2.2	Neuro-fuzzy systems in financial research	141
6.3	Experiment: Artificially generated stock price data	142
6.4	Data collection and experimental setup	144
6.5	Experimental results and analysis	147
6.6	Summary	154

7	Conclusions and Future Works	155
7.1	Conclusions	155
7.2	Future works	161
7.2.1	Recommendations on designs of neuro-fuzzy systems	161
7.2.2	Recommendations on application areas	166
	Bibliography	171
A	Parameter Adaptation in SaFIN	185
B	Parameter Adaptation in eT2FIS	188
C	Publication List	193

List of Figures

2.1	A neuron u	10
2.2	A 3-layers MLP.	11
2.3	A Type-1 gaussian fuzzy set $\mu(c, \sigma; x)$	15
2.4	Architecture of a Type-1 fuzzy system.	16
2.5	An interval Type-2 gaussian fuzzy set with centre $c \in [c^L, c^R]$ and width σ	18
2.6	Architecture of a Type-2 fuzzy system.	19
2.7	An interval Type-2 gaussian fuzzy set, reproduced from Figure 2.5, with a singleton point.	21
2.8	Types of neuro-fuzzy systems in the literature.	24
2.9	Flowchart of batch learning in neuro-fuzzy systems.	27
2.10	Flowchart of online learning in neuro-fuzzy systems.	32
3.1	Architecture of the Self-adaptive Fuzzy Inference Network (SaFIN).	43
3.2	Flowchart of the self-organizing clustering technique CLIP adopted in SaFIN.	49
3.3	Initialization of the self-organizing clustering technique performed in an input-output dimension of SaFIN.	51
3.4	Clustering steps performed in SaFIN after initialization.	54
3.5	Flowchart of the self-automated rule generation mechanism adopted in SaFIN.	56
3.6	Input spaces fuzzy partitioning for the identification of a nonlinear system in: (A) SaFIN; (B) SONFIN <i>before</i> alignment; and (C) SONFIN <i>after</i> alignment.	58

3.7	(a) Identified fuzzy rulebase; and (b) Modeling results on testing dataset for the identification of a nonlinear system using SaFIN.	60
3.8	Semantic fuzzy clusters in the first input dimensions; and modeling results on testing datasets for the Nakanishi dataset using SaFIN.	63
3.9	Distribution of iris dataset.	70
3.10	Fuzzy clusters identified by SaFIN (with corresponding numerical data) for the iris classification task.	71
4.1	Fundamental principles of traffic flow analysis.	80
4.2	Location of site 29 along the Pan Island Expressway in Singapore.	82
4.3	Traffic flow densities of the three straight lanes along PIE at site 29.	83
4.4	Fuzzy clusters identified by SaFIN for the time domain, and for lanes L1–L3 (with distributions of raw data) for the training set of CV1 when $\tau = 5$ minutes.	84
4.5	Recall and prediction of traffic density on lane L1 when $\tau = 5$ minutes.	87
4.6	Traffic flow prediction results.	89
5.1	Architecture of the evolving Type-2 neural Fuzzy Inference System (eT2FIS).	97
5.2	Flowchart of the incremental learning process employed in eT2FIS.	106
5.3	Fuzzy partitioning steps performed in eT2FIS.	112
5.4	A. Computation of similarity match between two fuzzy clusters; and B. Merger process of two highly over-lapping fuzzy labels in eT2FIS.	113
5.5	(a)–(c): Fuzzy partitioning achieved for the online identification of a non-time-varying nonlinear system; and (d) Identified fuzzy rulebase in eT2FIS.	117
5.6	(a) Number of fuzzy rules identified; and (b) Modeling results for the online identification of a non-time-varying nonlinear system using eT2FIS.	118
5.7	Modeling performance of eT2FIS when disturbance is (a) added; and (b) removed.	122
5.8	Number of fuzzy rules identified by eT2FIS when: (a) no deletion is activated; and (b) deletion is activated.	122
5.9	(a)–(c) Fuzzy partitioning achieved for the identification of a nonlinear noisy system with noise level $[-0.1, 0.1]$; and (d) Identified fuzzy rulebase in eT2FIS.	126

5.10	Modeling performance for eT2FIS when noise level is (a) $[-0.1, 0.1]$; and (b) $[-0.5, 0.5]$	126
5.11	Input spaces fuzzy partitioning for the noisy chaotic series in eT2FIS.	130
5.12	Modeling performance for eT2FIS when noise level is (a) $STD = 0.1$; and (b) $STD = 0.3$	130
6.1	Relations between correlations in stock returns and stock price movement: a simplistic illustration.	136
6.2	A particular generation of the artificial stock data.	142
6.3	Daily closing stock prices of Alcoa Incorporation from 2-nd January 1962 – 30-th July 2010 (together with the daily stock price differences).	145
6.4	Trading decision model with forecasting.	147
6.5	Semantic fuzzy labels identified by eT2FIS in the input spaces for $\lambda = 1$ day.	148
6.6	Online tracking performances of eT2FIS for a day ahead forecast of the stock.	149
6.7	Stock price tracking performances.	151
6.8	Online trading performances.	152
7.1	A summarized view of research contributions in this thesis.	157
7.2	Flowchart of performing feature analysis as a pre-processing stage <i>prior</i> to the design of a neuro-fuzzy system.	162
7.3	Proposed flowchart for performing feature analysis <i>integrated</i> into the design of a neuro-fuzzy system.	163
7.4	Proposed flowchart for performing online feature analysis <i>integrated</i> into the design of an evolving neuro-fuzzy system.	164

List of Tables

2.1	A summary of evolving neuro-fuzzy systems in the literature.	34
3.1	A comparison between clustering techniques.	48
3.2	Sensitivity test for different values of α and β in SaFIN.	61
3.3	Mamdani-type fuzzy rules identified in SaFIN for the Nakanishi dataset. . .	65
3.4	Consolidated experimental results for the Nakanishi dataset using SaFIN. .	67
3.5	Consolidated experimental results for the UCI dataset using SaFIN.	69
4.1	Mamdani-type fuzzy rules identified in SaFIN for CV1 when $\tau = 5$ minutes.	86
4.2	Average performances for the traffic flow prediction.	90
5.1	A summary of evolving neuro-fuzzy systems in the literature (reproduced from Table 2.1).	94
5.2	Sensitivity test for different values of Λ in eT2FIS.	119
5.3	Sensitivity test for different values of Ω in eT2FIS.	120
5.4	Results of online identification of a non-time-varying system using eT2FIS.	120
5.5	Sensitivity test for different values of Γ in eT2FIS.	123
5.6	Performances of eT2FIS, eT1FIS-V1 and eT1FIS-V2 for the identification of a nonlinear noisy system.	124
5.7	Performances of eT2FIS and the benchmarking models for the identification of a nonlinear noisy system.	127
5.8	Average performances of the identification of a chaotic time-series using eT2FIS.	131

6.1	Average performances for the forecasting of the price difference on the artificial stock price data.	143
6.2	Average performances for the forecasting of the price difference on the artificial stock price data with injected uniform noise.	144
6.3	Mamdani-type fuzzy rules identified in eT2FIS for $\lambda = 1$ day.	150
6.4	Average performances for the stock price tracking task.	151
6.5	Performances for the online stock trading task.	152

Abstract

Neuro-fuzzy systems are hybrid systems that possess the functionalities of the two individual systems, where a fuzzy IF-THEN model can be extracted from given numerical data using neural learning approaches. Depending on the numerical data presented, there are generally two approaches to the design of a neuro-fuzzy system; namely, design via batch learning, and design via sequential learning. Currently, the main problems dogging existing neuro-fuzzy systems of the former approach are: (1a) an inconsistent rulebase; (1b) the need for prior knowledge such as the number of clusters to be computed; (1c) heuristically designed knowledge acquisition methodologies; and (1d) the stability-plasticity trade-off of the system. Such weaknesses are directly related to the techniques employed for the fuzzy partitioning and fuzzy rulebase identification in the systems. On the other hand, existing neuro-fuzzy systems of the latter approach are commonly plagued by the following problems: (2a) susceptibility to noisy data; (2b) an ever-expanding knowledge structure; and (2c) a lack of interpretability of the system. Such drawbacks are the consequences of online learning in an uncertain and time-varying environment.

This thesis consists of 2 sections. In the first section, the thesis addresses issues faced by batch learning neuro-fuzzy systems using a novel hybrid intelligent Self-adaptive Fuzzy Inference Network (SaFIN); while in the second section, the thesis focuses on improving the robustness of online learning in an uncertain and time-varying environment using a novel evolving Type-2 neural Fuzzy Inference System (eT2FIS). The proposed models address the problems with the following contributions:

A novel fuzzy partitioning technique named Categorical Learning Induced Partitioning (CLIP) is employed in SaFIN to obtain an initial fuzzy partitioning of the input and output spaces such that it is tailored to the numerical data presented. Inspired from the behavioral category learning process demonstrated in humans, the one-pass CLIP is able to incorporate new clusters in each input-output dimension when the existing clusters are not able to

give a satisfactory representation of the incoming data. This not only omits the need for prior knowledge regarding the number of clusters needed for each input-output dimension, it also allows SaFIN a flexibility to incorporate new knowledge with old knowledge in the system. In addition, a self-automated rule formation mechanism proposed within SaFIN ensures that it obtains a consistent resultant rulebase. The SaFIN model is subsequently evaluated using a real-life application involving the modeling of highway traffic flow density.

Most existing online sequential learning neuro-fuzzy systems are vulnerable under noisy learning environment because they are Type-1 models. On the other hand, eT2FIS synergizes an online learning system with a Type-2 fuzzy model, which is better known for its noise resistance ability. In addition, a three-steps online learning paradigm is proposed in eT2FIS to directly curb the ever-expanding knowledge structure and improve the interpretability of the system: A new rule is created only when a newly arrived data is novel; and an obsolete rule is deleted when it is no long relevant to the learning environment; while highly over-lapping clusters in the input-output spaces are merged to reduce the computational complexity and improve the overall interpretability of the system. Subsequently, the eT2FIS model is used in the modeling of a real-life application involving the tracking of stock price movement.

Chapter 1

Introduction

Neuro-fuzzy systems are a popular hybridization in the area of soft computing [Zadeh 94], which is formed from the synergy between a neural network and a fuzzy system. By combining the two individual systems into a homogenous architecture, a neuro-fuzzy system capitalizes on the functionalities of its predecessors [Nauck, Klawonn & Kruse 97]. This means that the system possesses high-level human-like reasoning mechanism where vague and imprecise information can be expressed as a set of human-interpretable linguistic IF-THEN fuzzy rules; while concurrently utilizing a low-level neural learning for the tuning of the system's parameters [Kecman 01].

This thesis is about the design of a neuro-fuzzy system, which involves two learning problems; namely, structure learning and parameter learning [Lin 94]. Structure learning tackles the issues of fuzzy partitioning in the input and output spaces, and the generation of fuzzy rules in the system; while parameter learning refers to the fine-tuning of the membership functions. To further elaborate, fuzzy partitioning for a neuro-fuzzy system determines the numbers, the positions and the spreads of the linguistic labels in each input-output space; while a rule generation scheme determines the set of fuzzy rulebase/knowledge base

governing the neuro-fuzzy system.

1.1 Motivation

The aim of developing a neuro-fuzzy system is to demonstrate its applicability, and ultimately develop it as a useful modeling tool for real-life applications [Margaliot & Langholz 00]. Thus, one approach to design a neuro-fuzzy system is to investigate the challenges faced in the field of modeling. As stated in [Castellano et. al. 05], a key issue in the area of modeling is to address the subjectivity-objectivity trade-off of the models. That is, under ideal conditions, modeling can be exploited through the following two extremes:

1. No initial subjective knowledge is involved and the modeling tool is extracted solely from objective numerical data (a process referred to as data-driven learning); and
2. The initial subjective knowledge is complete and no objective data is necessary for the design of the modeling tool (a process known as analytical learning).

Nevertheless, in real-world problems, learning is usually restricted by an incomplete prior knowledge and a finite dataset.

A spectrum of possibilities, therefore, exists between pure data-driven learning and pure analytical learning. Since the primary source of knowledge for analytical learning stems from the limited competence of each user, the result is often highly subjective and speculative. In addition, the difficulties in verbally formalizing interactions in a complex application may result in a loss of accuracy [Guillaume 01]. On the other hand, data-driven learning is gaining popularity because of technological advances that allow easy collection, storage and manipulation of data [Wang & Fu 05][Cherkassky & Mulier 07]. Hence, a learning model based on objective numerical data, with minimum prior subjective knowledge, is desirable.

Depending on the numerical data availability, there are generally two approaches to the design of a neuro-fuzzy system; namely, through batch learning; and through sequential learning [Tung & Quek 10]. Batch mode learning assumes that all numerical training data has been collected and is available before the design phase of a neuro-fuzzy system, where learning generally involves repetitive cycling of the numerical data over a number of epochs. Examples of applications using batch learning in neuro-fuzzy systems include classification tasks [Simpson 92][Wang & Lee 02], medical diagnosis [Aruna, Puviarasan & Palaniappan 05][Held et. al. 06][Tan et. al. 07], and control modeling [Pasquier, Quek & Toh 01][Li & Lee 03]. On the other hand, sequential learning assumes that the numerical training data is presented to a neuro-fuzzy system one at a time, where learning is performed based only on the current data tuple and it is discarded after learning. Some examples of applications using online sequential learning in neuro-fuzzy systems include financial trading instruments [Tung & Quek 05b][Tung & Quek 11], assistive biomedical instruments [Vidaurre et. al. 06][Shenoy et. al. 06], and modeling of physical phenomena [Horenko 10]. In general, neuro-fuzzy systems designed from the former approach are known as batch/offline systems; and those subscribing to the latter approach are commonly referred to as evolving/online systems.

1.1.1 Batch neuro-fuzzy systems

Existing batch neuro-fuzzy systems suffer from one or more of the following major problems; namely, (1) an inconsistent rulebase [Casillas et. al. 03]; (2) a need for prior knowledge such as the number of clusters to be computed; (3) heuristically designed knowledge acquisition methodologies; and (4) the stability-plasticity trade-off of the system [Grossberg 82]. Some examples from this class of neuro-fuzzy systems are [Wang & Langari 95][Quek & Zhou 96][Kim & Kasabov 99][Tung & Quek 02][Wang & Lee 02][Song et. al. 09].

Fuzzy rules are defined as *inconsistent* if they have very similar premise parts, but possess

rather different consequents [Jin, Von Seelen & Sendhoff 99]. Consistency of a fuzzy rulebase is a prevailing problem that affects the physical meaning of the resultant neuro-fuzzy system structure. Since one of the advantages of using a neuro-fuzzy system as a modeling tool is to benefit from its transparency where human-interpretable rules can be extracted to understand the application problem, a consistent rulebase is especially important as it allows a user to logically interpret the knowledge encoded within the system [Casillas et. al. 03].

On the other hand, some existing neuro-fuzzy systems [Wang & Langari 95][Quek & Zhou 96] adopt self-organizing numerical methods [Bezdek, Tsao & Pal 92][Bezdek 81][Kohonen 82] for their fuzzy partitioning, where prior knowledge regarding the number of clusters needed in the input-output spaces have to be pre-determined. Such information is heuristically defined initially for the design of a neuro-fuzzy system, which could lead to subsequent undesirable repercussions. This is because a neuro-fuzzy system that is restricted by the fixed number of clusters is unable to incorporate new clusters when the present partitioning of the system is found to be insufficient. As a result, the performance of the neuro-fuzzy system might suffer. This also leads to the issue of stability-plasticity trade-off of the system [Grossberg 82], where a regulated balance is necessary between the co-existence of past knowledge (as currently established in the neuro-fuzzy system) and any future knowledge (as introduced from new incoming data) for the modeling of an application environment. By fixing the number of input-output clusters, a neuro-fuzzy system is unable to expand its initial knowledge to incorporate any novel information from incoming data.

The above-stated deficiencies of batch neuro-fuzzy systems are directly related to the techniques employed for the fuzzy partitioning and fuzzy rulebase identification in the systems, and hence a neuro-fuzzy system that is free from these drawbacks is desirable.

1.1.2 Evolving neuro-fuzzy systems

Existing evolving neuro-fuzzy systems suffer from one or more of the following major problems; namely, (1) susceptibility to noisy data [Baraldi & Blonda 99a]; (2) an ever-expanding knowledge structure; and (3) a lack of interpretability of the system. Some examples from this class of neuro-fuzzy systems are [Lin & Lin 97][Frayman & Wang 98][Juang & Lin 98][Tzafestas & Zikidis 01][Kasabov 01][Wu, Er & Gao 01][Kasabov & Song 02][Angelov & Filev 04][Rong et. al. 06][Juang & Tsao 08a][Juang & Tsao 08b][Juang, Huang & Cheng 10][Tung & Quek 10].

Baraldi and Blonda [Baraldi & Blonda 99a] explain that online systems are generally more sensitive to the presence of noise because the effect of noisy data are not averaged over the entire training dataset, i.e. , a noisy training data is learned once it is presented to the system, and discarded immediately after the system has performed learning on it. As a result, an evolving neuro-fuzzy system that is robust under a noisy modeling environment is much sought after. While some existing evolving neuro-fuzzy systems [Juang & Tsao 08a][Juang & Tsao 08b][Juang, Huang & Cheng 10] have demonstrated promise of good robustness in the presence of noise, they are plagued by two major drawbacks; namely, an ever-expanding knowledge structure and a lack of interpretability of the system. These drawbacks are directly related to the life-long learning provided by the systems; yet, no attempt has been made to regulate the growth in the structure complexities of the systems as learning continues. This is because new rules and new clusters are continuously added when a neuro-fuzzy system continues to learn new knowledge from each incoming data tuple. Subsequently, there may be too many out-dated fuzzy rules and highly similar fuzzy labels created in the system, thus leading to degradation in the level of human interpretability of the neuro-fuzzy system. Interpretability of a neuro-fuzzy system follows two conditions stated in [Guillaume 01]; namely, (a) the fuzzy partition must be readable in the sense

that the fuzzy sets can be interpreted as linguistic labels; and (b) the set of rules must be compact with good generalization capability. Hence, an online learning procedure that is robust under noisy and time-varying learning environment is desirable in the design of an evolving neuro-fuzzy system.

1.2 Organization of thesis

This thesis consists of seven chapters:

Chapter 2 presents a brief review of some basic concepts on neural networks and fuzzy systems. This is followed by a literature review on neuro-fuzzy systems, where the need to design neuro-fuzzy systems from numerical data is motivated. Subsequently, two learning paradigms of learning from data is introduced, and the respective neuro-fuzzy systems adopting the two different learning paradigms are surveyed.

Chapter 3 presents the proposed batch learning *Self-adaptive Fuzzy Inference Network* (SaFIN). A novel fuzzy partitioning technique, named *Categorical Learning Induced Partitioning* (CLIP), employed in SaFIN is described. A novel rulebase identification procedure that ensures a consistent knowledge encryption in the SaFIN model is also introduced. The proposed model will be evaluated using three benchmarking experiments.

Chapter 4 presents experimental results of using SaFIN for a real-life highway traffic flow modeling application.

Chapter 5 presents the proposed evolving neuro-fuzzy system, named *evolving Type-2 neural Fuzzy Inference System* (eT2FIS). The online learning procedure of eT2FIS will be carefully described, highlighting on how the proposed model is able to overcome the current drawbacks in the existing evolving neuro-fuzzy systems. Subsequently, the proposed model

will be evaluated using two benchmarking experiments.

Chapter 6 presents experimental results of using eT2FIS for a real-life modeling application in stock price tracking.

Chapter 7 concludes this thesis and provides directions for future works.

Chapter 2

Literature Review

This chapter presents a brief review on some basic concepts in neural networks and fuzzy systems. These will serve as the foundation in understanding neuro-fuzzy systems, where the design paradigms of existing neuro-fuzzy systems are surveyed. Prevailing challenges of modeling using these neuro-fuzzy systems are also highlighted.

2.1 Neural networks

Inspired as a functionable model to mimic the information processing capability of the human brain, a definition for a neural network was given in [Haykin 98] as follows:

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

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

The basic building blocks in a neural network are the simple processing units, commonly known as *neurons* [McCulloch & Pitts 43]. Neurons that serve as inputs to the network are referred to as *input neurons*; those that serve as outputs to the network are called *output neurons*; while the remaining ones which are not directly influenced or observed from the environment are known as *hidden neurons*. The neurons are generally arranged in layers and are connected in a weighted directed graph manner that represents the architecture of the network.

This section shall review three basic concepts of a neural network; namely, (1) a *neuron*, the simplest processing unit in a neural network (see Section 2.1.1); (2) the *multilayer perceptron*, a particular class of feed-forward neural network (see Section 2.1.2); and (3) the *backpropagation algorithm*, which is a type of learning process for acquiring knowledge into a neural network (see Section 2.1.3). Formal mathematical definitions, based on [Nauck, Klawonn & Kruse 97], of these three concepts will be presented, and illustrations are subsequently provided.

2.1.1 A neuron

Definition 1 *A neuron u consists of three basic elements:*

1. *A set of synaptic weights, w_i , $i = 1 \dots n$, representing the interneuron connection strengths (or, the activation levels) between u and its neighbours (or, the environmen-*

Figure 2.1: A neuron u .

tal inputs), where n is the total number of inputs to u ;

- 2. An activation function, a , to compute the activation of u based on the net input to it;*
- 3. A computed output, o , of the neuron u .*

Figure 2.1 illustrates the configuration of a neuron u , where there are n input values to u . The input values x_i , $i = 1 \dots n$, either stem from other neighbouring neurons linked directly to u or from the environment. The set of synaptic weights are then denoted as w_i , $i = 1 \dots n$. Subsequently, the neuron u produces an output value o as in (2.1):

$$o = a(x_1, \dots, x_n; w_1, \dots, w_n) \quad (2.1)$$

where a is the activation function of u , which combines the information from the net input to u and produces an output based on this net input.

2.1.2 Multilayer perceptron

Definition 2 A multilayer perceptron (MLP) is a neural network as represented by the tuple (U, W, A, O) , where

Figure 2.2: A 3-layers MLP.

U is a finite set of processing units (neurons);

W , the network structure, is a mapping from the Cartesian product $U \times U$ to $\mathbb{R}$

where $W(u_i, u_j)$ is the synaptic weight from neuron u_i to neuron u_j ;

A is a collection of activation functions;

O is a collection of computed outputs.

It has the following properties:

1. $U = U_1 \cup \dots \cup U_M$, $M \geq 3$, where $U_m \neq \emptyset$ is the m -th layer of neurons in the MLP for $m = 1 \dots M$. Furthermore, $U_{m_1} \cap U_{m_2} = \emptyset$ for $m_1 \neq m_2$. Here, U_1 denotes the input layer (i.e. the collection of input neurons); while U_M denotes the output layer (i.e. the collection of output neurons).
2. There exists connections only between consecutive layers, i.e., if $u^{(m_1)} \in U_{m_1}$ such that $W(u^{(m_1)}, u^{(m_2)}) \neq 0$, then $u^{(m_2)} \in U_{m_1+1}$ for $m_1 = 1 \dots M-1$.

Figure 2.2 illustrates an example on the architecture of a 3-layers multilayer (feed-forward) perceptron, i.e., it consists of an input layer, a hidden layer and an output layer. The

inputs to the network is represented as the input vector $x = [x_1, \dots, x_P]^T$; while the outputs computed by the network is given as the output vector $y = [y_1, \dots, y_Q]^T$. Each neuron in the hidden layer is denoted as $u^{(2)}$; while each neuron in the output layer is denoted as $u^{(3)}$. Here, the MLP is *fully connected*, i.e. , every neuron from one layer is connected to every neuron in the successive layer. Subsequently, the activation levels of the environmental inputs are denoted as $W(x_p, u^{(2)})$; while the synaptic weights from layer 2 to layer 3 of the network are denoted as $W(u^{(2)}, u^{(3)})$.

When the input vector x is presented to the network, a neuron $u^{(2)}$ in layer 2 of the network computes an output value $o^{(2)}$ using (2.2):

$$o^{(2)} = a^{(2)} \left(x_1, \dots, x_P ; W(x_1, u^{(2)}), \dots, W(x_P, u^{(2)}) \right) . \quad (2.2)$$

Following that, an output neuron $u^{(3)}$ in layer 3 of the network computes its output value $o^{(3)}$ according to (2.3):

$$o^{(3)} = a^{(3)} \left(o_1^{(2)}, \dots, o_{|U_2|}^{(2)} ; W(u_1^{(2)}, u^{(3)}), \dots, W(u_{|U_2|}^{(2)}, u^{(3)}) \right) . \quad (2.3)$$

Each output neuron corresponds to one dimension of the output vector y such that $y = [o_1^{(3)}, \dots, o_{|U_3|}^{(3)}]^T$, i.e. , $|U_3| = Q$.

2.1.3 Backpropagation

Definition 3 Let MLP be a multilayer perceptron with $U = U_1 \cup \dots \cup U_M$, $M \geq 3$, and let L be a fixed learning problem. The backpropagation algorithm that determines the modifications for the network structure W of MLP after the presentation of an input vector

x is given as in (2.4):

$$\Delta W(u^{(m)}, u^{(m+1)}) = \eta \delta^{(m+1)} \frac{\partial o^{(m+1)}}{\partial W(u^{(m)}, u^{(m+1)})} \quad (2.4)$$

where $\eta > 0$ is the learning rate, and $\delta^{(m+1)}$ is the error signal of the neuron $u^{(m+1)}$ for $m = 1 \dots M - 1$.

As an illustration, consider the same 3-layers MLP architecture as shown in Figure 2.2. Here, the desired output vector corresponding to the input vector x is denoted as $d = [d_1, \dots, d_Q]^T$; while x_p is denoted as $u_p^{(1)}$. The sum of squared differences between the computed output and the desired output given by the fixed learning problem L is used as the error measure as defined in (2.5):

$$E = \frac{1}{2} \sum_q (d_q - y_q)^2. \quad (2.5)$$

The term *backpropagation* refers to the backward propagation of an error signal through the network. Since the backpropagation algorithm is based on a gradient descent approach, the goal of the learning is to minimize the error E by adjusting the weights of the network. Subsequently, the weight change is assumed as in (2.6):

$$\Delta W(u^{(m)}, u^{(m+1)}) \propto -\frac{\partial E}{\partial W(u^{(m)}, u^{(m+1)})}. \quad (2.6)$$

By chain rule, the derivative in (2.6) can be rewritten as in (2.7) as follows:

$$\begin{aligned} \frac{\partial E}{\partial W(u^{(m)}, u^{(m+1)})} &= \frac{\partial E}{\partial o^{(m+1)}} \frac{\partial o^{(m+1)}}{\partial W(u^{(m)}, u^{(m+1)})} \\ &= -\delta^{(m+1)} \frac{\partial o^{(m+1)}}{\partial W(u^{(m)}, u^{(m+1)})} \end{aligned} \quad (2.7)$$

where $\delta^{(m+1)} = -\partial E / \partial o^{(m+1)}$ denotes the error signal of the neuron $u^{(m+1)}$. Substituting (2.7) into (2.6), one can derive the equality relation as given in (2.4):

$$\Delta W(u^{(m)}, u^{(m+1)}) = \eta \delta^{(m+1)} \frac{\partial o^{(m+1)}}{\partial W(u^{(m)}, u^{(m+1)})}$$

where $\eta > 0$ is a learning constant.

2.2 Fuzzy systems

Fuzzy set theory was first introduced by Zadeh in 1965 [Zadeh 65], and he later extended his original works to introduce higher orders fuzzy sets in [Zadeh 75a]. This section shall review some background information on Type-1 and interval Type-2, which is a particular class of Type-2, fuzzy sets and systems.

2.2.1 Type-1 fuzzy sets and systems

Definition 4 A Type-1 fuzzy set, A , defined on the universe of discourse, X , is characterized by a membership function $\mu_A : X \rightarrow [0, 1]$ as defined in (2.8):

$$A = \int_X \mu_A(x) / x . \quad (2.8)$$

Here, $\int$ is the union of the elements $x \in X$ with the associated membership grade $\mu_A(x)$.

Figure 2.3 illustrates a Type-1 fuzzy set on the universe of discourse $X = [0, 5]$ where the membership function is defined as a gaussian function as described in (2.9):

$$\mu(c, \sigma; x) = e^{-((x-c)^2/\sigma^2)} \quad (2.9)$$

Figure 2.3: A Type-1 gaussian fuzzy set $\mu(c, \sigma; x)$.

such that c and σ are the centre and width of the function respectively.

Definition 5 A Type-1 fuzzy (logic) system uses only Type-1 fuzzy sets and its basic architecture is shown in Figure 2.4. There are four basic elements in the system:

1. A fuzzifier that transforms the crisp inputs to appropriate fuzzy inputs;
2. A knowledge base that stores the set of IF-THEN fuzzy rules, with their corresponding linguistic expressions as represented by the Type-1 fuzzy sets, containing the knowledge of the system;
3. A reasoning mechanism that performs inference, as specified by the reasoning method, using the inference engine where the inputs to the engine are the fuzzified inputs, fuzzy sets and fuzzy rules, to derive a fuzzy output;
4. A defuzzifier that converts the fuzzy output into a crisp output.

A general form of a fuzzy rule can be expressed as in (2.10):

$$\text{IF } \langle \text{antecedent} \rangle \text{ THEN } \langle \text{consequent} \rangle \quad (2.10)$$

where the *antecedent* of a fuzzy rule defines a set of conditions, and the *consequent* defines a set of conclusions that can be inferred from the rule when the conditions are met. Subsequently, a fuzzy system can be categorized into two classes depending on the consequences

Figure 2.4: Architecture of a Type-1 fuzzy system.

of the fuzzy rules adopted in the system as described in (2.11):

Mamdani :IF x_1 is A_1 and ... and x_P is A_P THEN y is C

Takagi-Sugeno-Kang :IF x_1 is A_1 and ... and x_P is A_P THEN y is $f(x_1, \dots, x_P)$

(2.11)

where x_p , $p = 1 \dots P$, and y are the input and output (linguistic) variables respectively; A_p , $p = 1 \dots P$, are the input labels for each of the input variables where each A_p is characterized by a Type-1 fuzzy set in the domain x_p . In the former class, the consequent part of the fuzzy rule is also fuzzy where C is the output label, as characterized by a Type-1 fuzzy set, in the domain y ; and such a system is known as a *Mamdani model* [Mamdani 77]. In the latter class, the consequent part of the fuzzy rule is expressed as a linear function f in terms of the input values; and such a system is commonly referred to as a *Takagi-Sugeno-Kang (TSK) model* [Takagi & Sugeno 85][Sugeno & Kang 88]. In general, Mamdani models are also known as *linguistic fuzzy models*, which are focused on the interpretability of the fuzzy systems; while TSK models are known as *precise fuzzy models*, which are focused on the accuracy of the fuzzy systems [Ang & Quek 05].

Unlike classical binary logic where inference in (2.10) holds when the antecedent conditions of the implication is entirely fulfilled, fuzzy inferencing is concerned with the partial fulfillment of the antecedent conditions in (2.10). Several reasoning methods have been proposed in the literature to compute such partial matching of conditions; namely, the *Compositional Rule of Inference* (CRI) [Zadeh 75b], the *Yager inference scheme* [Keller, Yager & Tahani 92], the *Truth Value Restriction* (TVR) [Mantaras 90], and the *Approximate Analogical Reasoning Schema* (AARS) [Turksen & Zhong 90].

2.2.2 Interval Type-2 fuzzy sets and systems

Interval Type-2 fuzzy sets [Liang & Mendel 00][Mendel & John 02], which are a particular class of Type-2 fuzzy sets [Zadeh 75a], are considered in this thesis for their reduced computational cost [Mendel, John & Liu 06]. Understanding that a Type-1 fuzzy set is certain in the sense that its membership grades are crisp, researchers have reported on the modeling limitations of using a Type-1 fuzzy set to minimize the effect of uncertainties [Hagras 04][Sepúlveda et. al. 07]. On the other hand, Type-2 fuzzy set theory [Mizumoto & Tanaka 76] has generated much interest because the membership functions are themselves fuzzy, and this potentially helps to overcome the modeling bottlenecks of its Type-1 counterparts [Mendel & John 02].

Definition 6 An interval Type-2 fuzzy set, $\tilde{A}$, defined on the universe of discourse, X , is characterized by a membership function $\mu_{\tilde{A}}(x)$ as defined in (2.12):

$$\begin{aligned}\tilde{A} &= \int_{x \in X} \mu_{\tilde{A}}(x) / x \\ &= \int_{x \in X} \left[\int_{u \in J_x} 1/u \right] / x\end{aligned}\tag{2.12}$$

where x is called the primary variable with domain X ; u is called the secondary variable

Figure 2.5: An interval Type-2 gaussian fuzzy set with centre $c \in [c^L, c^R]$ and width σ .

with domain $J_x \subseteq [0, 1]$ for each $x \in X$; J_x is called the primary membership of x ; and the secondary membership is unity for all $x \in X$ and all $u \in J_x$. The footprint of uncertainty (FOU) of $\tilde{A}$ is the union of all primary memberships, i.e. , $FOU(A) = \bigcup_{x \in X} J_x$.

Figure 2.5 illustrates an interval Type-2 gaussian fuzzy set on the domain $X = [0, 5]$ with varying centre, $c \in [c^L, c^R]$, and width, σ , whose FOU is defined as in (2.13):

$$\mu(x) = [\underline{\mu}(x), \bar{\mu}(x)] \quad (2.13)$$

where $\underline{\mu}(x)$ and $\bar{\mu}(x)$ are the lower and upper membership functions respectively. They are defined accordingly as in (2.14):

$$\underline{\mu}(x) = \begin{cases} \mu^R(c^R, \sigma; x) & \text{if } x \leq 1/2 (c^L + c^R) \\ \mu^L(c^L, \sigma; x) & \text{otherwise} \end{cases} ; \quad \bar{\mu}(x) = \begin{cases} \mu^L(c^L, \sigma; x) & \text{if } x \leq c^L \\ 1 & \text{if } c^L < x \leq c^R \\ \mu^R(c^R, \sigma; x) & \text{if } x > c^R \end{cases} \quad (2.14)$$

where $\mu^L(c^L, \sigma; x)$ and $\mu^R(c^R, \sigma; x)$ are the left and right formation functions respectively, such that they are given as gaussian functions as in (2.9). Thus, for a fixed $x \in X$, the primary membership J_x is bounded by the interval $[\underline{\mu}(x), \bar{\mu}(x)]$.

Definition 7 A Type-2 fuzzy (logic) system uses Type-2 (and Type-1) fuzzy sets and its basic architecture is shown in Figure 2.6. There are four basic elements in the system:

Figure 2.6: Architecture of a Type-2 fuzzy system.

1. A fuzzifier that transforms the crisp inputs to appropriate fuzzy inputs;
2. A knowledge base that stores the set of IF-THEN fuzzy rules, with their corresponding linguistic expressions as represented by the fuzzy sets, containing the knowledge of the system;
3. A reasoning mechanism that performs inference, as specified by the reasoning method, using the inference engine where the inputs to the engine are the fuzzified inputs, fuzzy sets and fuzzy rules, to derive a fuzzy output;
4. An output processing unit that converts the fuzzy output into a crisp output, by first passing through a type-reducer to obtain a Type-1 fuzzy set, and then through a defuzzifier to obtain a crisp output.

For an interval Type-2 fuzzy system, the system uses only interval Type-2 fuzzy sets.

Since a Type-1 fuzzy set can be seen as a special case of an interval Type-2 fuzzy set where the primary membership of x (i.e. J_x) is a crisp value, the fuzzy rules used in an interval

Type-2 fuzzy system shall be expressed generically as in (2.15):

$$\begin{aligned}
 &\textbf{Mamdani} && \text{:IF } x_1 \text{ is } \tilde{A}_1 \text{ and } \dots \text{ and } x_P \text{ is } \tilde{A}_P \text{ THEN } y \text{ is } \tilde{C} \\
 &\textbf{Takagi-Sugeno-Kang} && \text{:IF } x_1 \text{ is } \tilde{A}_1 \text{ and } \dots \text{ and } x_P \text{ is } \tilde{A}_P \text{ THEN } y \text{ is } f(x_1, \dots, x_P)
 \end{aligned}
 \tag{2.15}$$

where x_p , $p = 1 \dots P$, and y are the input and output (linguistic) variables respectively; $\tilde{A}_p$, $p = 1 \dots P$, are the input labels for each of the input variables where each $\tilde{A}_p$ is characterized by an interval Type-2 fuzzy set in the domain x_p ; $\tilde{C}$ is the output label, characterized by an interval Type-2 fuzzy set, in the domain y ; and f is a linear function defined in terms of the input values. Similar to a Type-1 fuzzy system (see (2.11)), one can categorize a Type-2 fuzzy system into a Mamdani model or a TSK model depending on the consequences of the fuzzy rules adopted in the system. Likewise, several reasoning methods have also been extended under the Type-2 context [Liang & Mendel 00][TUNG, Quek & Guan 09].

Comparing a Type-1 fuzzy system (Figure 2.4) with a Type-2 fuzzy system (Figure 2.6), one noticeable difference is the output processing of the fuzzy outputs performed by the two systems. Under a Type-2 fuzzy system, a process known as *type-reduction* transforms the Type-2 output set to a Type-1 fuzzy set, called the *type reduced set*, which is subsequently defuzzified to a crisp output through the defuzzifier. There are several type-reduction processes proposed in the literature [Karnik, Mendel & Liang 99]; namely, the *centroid type-reduction*, the *height type-reduction*, the *centre-of-sets type-reduction*, the *modified height type-reduction*, and the *centre-of-sums type-reduction*. To date, practical computations for a Type-2 fuzzy system are only possible when all the fuzzy sets used are interval Type-2 fuzzy sets [Mendel 07]. Subsequently, the iterative algorithm of Karnik-Mendel [Karnik, Mendel & Liang 99] for computing the height type-reduction [Driankov, Hellendoom & Reinfrank 96] is presented, where the type reduced set is an interval Type-1 set denoted by

Figure 2.7: An interval Type-2 gaussian fuzzy set, reproduced from Figure 2.5, with a singleton point.

the interval $[Z_l, Z_r]$:

Height type-reduction via KM iterative algorithm:

The height type-reducer replaces each Type-2 fuzzy set by a single point in the primary domain, referred to as a *singleton*, where it is denoted as z . Subsequently, for an interval Type-2 fuzzy set with a singleton z , the membership of z is an interval with bounds $[L, R]$. The notations can be explained using Figure 2.7, where the interval Type-2 fuzzy set of Figure 2.5 is reproduced. As seen, a possible singleton point for this fuzzy set is illustrated by the blue line.

For a set of L interval Type-2 fuzzy sets in a primary domain, let z_l , $l = 1 \dots L$, denote the L singletons, each with membership bounded as $[L_l, R_l]$, $l = 1 \dots L$. Assume that $z_1 \leq \dots \leq z_L$. For convenience, define the following function as in (2.16):

$$H(\theta_1, \dots, \theta_L) = \frac{\sum_{l=1}^L z_l \theta_l}{\sum_{l=1}^L \theta_l}. \quad (2.16)$$

To compute the upper bound Z_r of the type reduced set:

- Step 1: Set $\theta_l = (L_l + R_l) / 2$ for $l = 1 \dots L$. Compute $H_1 = H(\theta_1, \dots, \theta_L)$.
- Step 2: Determine l^* , where $1 \leq l^* \leq L - 1$, such that $z_{l^*} \leq H_1 \leq z_{l^*+1}$.
- Step 3: Set $\theta_l = L_l$ for $l = 1 \dots l^*$ and $\theta_l = R_l$ for $l = l^* + 1 \dots L$. Compute

$$H_2 = H(\theta_1, \dots, \theta_L).$$

- Step 4: If $H_1 = H_2$, then the upper bound is found and $Z_r = H_2$. Else, set H_1 as H_2 , and repeat by returning to Step 2.

To compute the lower bound Z_l of the type reduced set, the main difference is Step 3 of the above algorithm, where one sets $\theta_l = R_l$ for $l = 1 \dots l^*$ and $\theta_l = L_l$ for $l = l^* + 1 \dots L$.

2.3 Neuro-fuzzy systems

Neuro-fuzzy systems are hybrid systems that bring together the merits of the individual systems of fuzzy logic and neural networks [Lin & Lee 96][Nauck, Klawonn & Kruse 97]. By synergizing the workings of a neural network with that of a fuzzy system, a neuro-fuzzy system possesses both the advantages of a neural network (e.g. learning capabilities, parallel distributed processing abilities, and optimization techniques) and a fuzzy system (e.g. human-like reasoning abilities, ability to handle vague and imprecise concepts, and human-interpretation for complex systems). Furthermore, the synergized system helps to overcome the drawbacks faced by its individual predecessors; namely, the black-box phenomenon in a neural network, and the lack of learning abilities in a fuzzy system, and hence enjoy greater modeling superiority.

The architecture of a neuro-fuzzy system generally takes the form of a feed-forward network as illustrated in Figure 2.2, and incorporates the structure of a Type-1 or Type-2 fuzzy system as illustrated in Figures 2.4 or 2.6 respectively. To design a neuro-fuzzy system, one thus needs to consider a crucial conceptual component of a fuzzy system [Jang & Sun 95] – the *knowledge base*, which comprises of the two fundamental components of *fuzzy partitioning* and *fuzzy rules*.

To derive the knowledge base (i.e. the identification of linguistic expressions given by the fuzzy partitioning of the input-output spaces, and the identification of the set of IF-THEN fuzzy rules via fuzzy rulebase generation, see Figures 2.4 and 2.6), existing neuro-fuzzy systems generally adopt two different approaches; namely, design by expert knowledge, and design from data. In the former class, the computational structure of a neuro-fuzzy system is manually crafted by the human experts. Prior knowledge for the design of fuzzy partitioning in the input-output spaces and the fuzzy rulebase of the system is determined by the human experts. Some examples of neuro-fuzzy systems in this class include the ANFIS [Jang 93] and the GARIC [Berenji & Khedkar 92] models. In the latter class, the neuro-fuzzy system is based on a self-reliant and automated design from numerical data. Safe from a set of user-defined parameters, the neuro-fuzzy system is able to derive the fuzzy partitioning and fuzzy rules from the numerical dataset. Some examples of neuro-fuzzy systems in this class include the family of POPFNN [Quek & Zhou 96][Quek & Zhou 99][Ang, Quek & Pasquier 03][Quek & Singh 05] models, the SONFIN [Juang & Lin 98] model, the family of GenSoFNN [Tung & Quek 02][Tung & Quek 05a][Oentaryo, Pasquier & Quek 08] models, the SAFIS [Rong et. al. 06] model, etc. Figure 2.8 provides a brief categorization of some existing neuro-fuzzy systems in the literature.

From this classification, it can be noted that the issue of subjectivity-objectivity trade-off is common to both the area of neuro-fuzzy systems and the field of modeling. More specifically, modeling takes place between two extremes, which can be identified in the following ideal conditions: (1) the initial knowledge is nil and the model is extracted by a learning process based on a set of data; and (2) the initial knowledge is complete and correct, and the model can be learned without the need of any data [Castellano et. al. 05]. In the former case, inductive methods are applied to generalize the knowledge derived from a set of data to a wider context – a process known as *data-driven learning*; while in the latter case, a deductive inference is performed where the general purpose prior knowledge is employed –

Figure 2.8: Types of neuro-fuzzy systems in the literature.

a process known as *analytical learning* [Cloete & Zurada 00]. Unfortunately, learning based on prior knowledge often leads to a less desirable accuracy due to high complexities in large systems [Guillaume 01]. Availability of complete and correct prior knowledge is also not possible in most application environments. On the other hand, technological advances are making numerical data more readily available for deriving models [Wang & Fu 05], where useful relationships between the input-output variables can be estimated. Thus, there is currently a paradigm shift from classical modeling based on first principles to developing models from data [Cherkassky & Mulier 07]. Correspondingly, such a shift is also observed in the approaches used for designing a neuro-fuzzy system as seen from Figure 2.8, where data-based neuro-fuzzy systems are preferred in recent times.

This section shall discuss on the following: Section 2.3.1 discusses on the two learning paradigms of learning from data; subsequently, Section 2.3.2 and Section 2.3.3 shall review

existing neuro-fuzzy systems designed from the two data-based learning paradigms.

2.3.1 Online versus offline learning

When system parameters are learned from training data, there are two classes of learning situations, depending on how data are presented to the system; namely, batch/offline learning, and online learning [Baraldi & Blonda 99a]. Under batch learning, data is available as a block; while data arrives sequentially under online learning. As pointed out in [Baraldi & Blonda 99a], when a sequential stream of data can be stored for analysis as a block, or a block of data can be analyzed sequentially, one is free to adopt either the batch or online learning paradigms. Nevertheless, there are situations when one learning setting is desirable over the other:

Online learning performs knowledge extraction with the arrival of each training data tuple and it is discarded after learning, thus avoids storage of the entire dataset. This is especially important when there are constraints to the computational time, or memory requirements during the learning process [Jain, Murty & Flynn 99]. In addition, incoming data may come in a continuous stream of unlimited length, which makes it impossible to apply batch learning [Bishop 95]. Online learning is also desirable when a learning system is required to respond in real time to new data samples, where the dynamics of the data samples changes with time. Despite such advantages, online learning possesses several issues; namely, (1) order-dependency of training data, and (2) susceptibility to noise. Firstly, since the training tuples are considered one at a time, the order of the training tuples affects the resultant structure of the learning system [Jain, Murty & Flynn 99]. Secondly, online systems are generally more sensitive to the presence of noise as their learning do not average the noise over the entire dataset, i.e. , they tend to provide high oscillatory outputs that perform poorly in terms of generalization abilities [Baraldi & Blonda 99a].

On the other hand, the goal of batch learning is to use a finite dataset to generalize the behaviour of an application problem to a generic situation. It is thus preferred when the main interest is the final answer, i.e. , the best generalization that one can obtain from a finite set of data [Baraldi & Blonda 99a]. Since batch learning enjoys the luxury of repeated presentations of the training dataset, convergence of learning is ensured. In addition, exact closed form solution to a learning problem is also possible for small datasets [Baraldi & Blonda 99a]. However, batch learning risks the danger of over-fitting when the training data is repeatedly presented to the learning system. This means that the resultant structure might be more complex than required from repeated learning of outliers present in the training data, thus leading to a decrease in the training error on the training data, but an increase in the generalization error on unseen data [Yang, Murata & Amari 98].

Both online and offline learning have been widely adopted in designing a neuro-fuzzy system from data. This can be seen from Figure 2.8. Batch learning in neuro-fuzzy systems typically involves repetitive cycling of the block of numerical data over a number of epochs, and they are commonly referred to as batch/offline neuro-fuzzy systems; on the other hand, online learning in neuro-fuzzy systems is carried out in a sequential learning mode such that learning is performed on a single data tuple at each time instance, and they are commonly known as evolving/online neuro-fuzzy systems. Section 2.3.2 reviews the design paradigms of existing batch neuro-fuzzy systems, where issues dogging this class of neuro-fuzzy systems will be highlighted. Following that, Section 2.3.3 reviews the design paradigms of existing evolving neuro-fuzzy systems, where issues plaguing this class of neuro-fuzzy systems will also be highlighted.

Figure 2.9: Flowchart of batch learning in neuro-fuzzy systems.

2.3.2 Batch neuro-fuzzy systems

Under batch learning, the block of training data is collected and is available before the design phase of the neuro-fuzzy system. This means that the knowledge from the numerical data can be used in every stage of designing the system. Figure 2.9 shows a generic flowchart of designing a neuro-fuzzy system via batch learning. As seen, the various stages of designing the neuro-fuzzy system are decoupled, i.e., one stage has to be completed through at least one cycle of the block of training data before the next stage proceeds, where parameter learning follows structure learning, and fuzzy partitioning is performed prior to fuzzy rules generation. The block of training data is also accessible at each decoupled stage, such that the relevant knowledge from the numerical data can be extracted. The neuro-fuzzy system is finally ready for use after parameter learning has been completed for the whole block of training data. The resultant trained neuro-fuzzy system can then be used to perform prediction on unseen data.

Following the flowchart in Figure 2.9, one is able to break up the various stages of designing a batch neuro-fuzzy system and examine issues stemming from each stage. Here, exist-

ing fuzzy partitioning techniques and fuzzy rules generation mechanisms shall be reviewed separately as follows:

Fuzzy partitioning

Fuzzy partitioning in a neuro-fuzzy system includes the determination of the numbers, the positions and the spreads of the fuzzy (linguistic) labels in each input and output linguistic variables. Eliciting membership functions from data is one of the fundamental issues associated with the applications of fuzzy logic as the success of an algorithm depends on the membership functions used [Medasani, Kim & Krishnapuram 98]. Nevertheless, the difficulties in achieving this goal lie in three aspects; namely, (1) there are no guidelines or rules that can be used to choose the appropriate membership generation technique; (2) a general lack of consensus on the definition and interpretation of membership functions; and (3) there are no measures available for evaluating the goodness or correctness of the membership functions generated using a particular method [Medasani, Kim & Krishnapuram 98]. As a result, there are several approaches to perform fuzzy partitioning in a neuro-fuzzy system as listed below:

Heuristics – uses a pre-determined number of membership functions defined evenly within each input-output domain range [Wang & Mendel 92][Kim & Kasabov 99]. This approach generates highly ordered membership functions within each input-output space, but to decide on the pre-determined number of membership functions remains a trial-and-error approach.

Clustering – partitions a dataset into subsets called clusters so that data in each subset are similar. This approach employs an unsupervised self-organization where seed clusters are determined a priori [Wang & Lee 02], and they are subsequently adapted to efficiently cover the regions where numerical data are present. As before, to decide on the pre-determined number of seeds remains a trial-and-error approach [Medasani, Kim & Krishnapuram 98].

In addition, new clusters cannot be added in the input-output spaces when the existing fuzzy partitioning is found to be insufficient in describing the set of training data. Although clustering algorithms such as Adaptive Resonance Theory [Carpenter et. al. 92][Lin & Lin 97] and Discrete Incremental Clustering [Tung & Quek 02][TUNG, Quek & Guan 08] attempt to rectify these problems, both approaches are based on a bottom-up approach where initial clusters grow to incorporate new training data. This may result in an over-expansion, and subsequently a loss of semantics to the final clusters formed. Nevertheless, the clustering approach remains a popular approach to fuzzy partitioning. For example, the linear vector quantization [Kohonen 82] has been used in established neuro-fuzzy systems such as [Lin & Lee 91][Quek & Zhou 96]; the fuzzy C-means [Bezdek 81] has been used in [Wang & Langari 95][Panella & Gallo 05]. In addition, a vast amount of literature have been reviewed on the topic of fuzzy clustering [Yang 93][Baraldi & Blonda 99a][Baraldi & Blonda 99b].

Projection – is a special case of fuzzy clustering, where clustering is performed for the combined input and output spaces respectively. Subsequently, membership functions in each input-output space are obtained by projecting the multi-dimensional input-output clusters into its respective uni-dimensional input-output spaces [Ruspini 82][Rong et. al. 06]. Unfortunately, the resultant projected membership functions are always highly over-lapping and opaque to the user, especially in the case of high input-output dimensionalities [Juang & Lin 98].

Summarizing, the following traits are desirable for a fuzzy partitioning technique in a neuro-fuzzy system:

1. There is no need to pre-define the number of input-output clusters, where the number of membership functions in each input-output space can differ and is self-determined based on the training data.

2. Membership functions in an input-output space can be added when necessary, such that old knowledge encoded in the system and new knowledge introduced from incoming data can co-exist to define the resultant system.
3. The fuzzy clusters created are human-interpretable and readable with clear semantic meanings, i.e. , the resultant membership functions in the input-output spaces should be well-ordered.

Fuzzy rules generation

One of the key advantages of using a neuro-fuzzy system as a modeling tool is to extract meaningful and human-interpretable linguistic fuzzy rules from the system, which can then be used to explain and understand the underlying dynamics of an application problem. Thus, eliciting interpretable fuzzy rules from numerical data is one of the most important aspects in designing a neuro-fuzzy system. As previously introduced, the general form of a fuzzy rule can be expressed as in (2.10), where two classes of neuro-fuzzy systems subsequently follow (see (2.11) and (2.15)).

A number of research works on the identification of fuzzy rules from numerical data have been reported [Quek & Zhou 96][Kim & Kasabov 99][Tung & Quek 02][Ang & Quek 05][Song et. al. 09], and a review can be found in [Mitra & Hayashi 00]. In general, the rulebase identification mechanism is carried out in three steps; namely, (1) an initial stage of self-organizing learning or clustering to determine the membership functions in the input-output spaces (i.e. forming the antecedent and consequent segments of *the* fuzzy rules); (2) competitive learning [Rumelhart & Zipser 85] follows to identify the set of fuzzy rulebase (i.e. identification of the antecedent and consequent segments of *a* fuzzy rule); and (3) a final stage of supervised learning to optimize the parameters of the membership functions in the input-output spaces (i.e. optimization of the fuzzy rules through a refinement of the respective antecedent and consequent segments). Step (1) is associated with fuzzy partitioning

which has been previously reviewed, and step (3) is associated with parameter learning; here, different approaches to step (2) are listed as follows:

Combinatorics of fuzzy rules – initializes the fuzzy rulebase to be the maximum combinatoric number of fuzzy rules with a given fuzzy partitioning of the input spaces, i.e., the number of fuzzy rules is given as $\prod_p J_p$ where J_p is the number of membership functions in an input space x_p . A serious drawback of this approach is the *curse of dimensionality* [Brown & Harris 94], where the size of the identified rulebase increases exponentially with the number of input dimensions. In addition, some rules created are never fired during the parameter learning stage of the neuro-fuzzy system because there are no training data that are residing in those regions. Following the antecedent identification, the consequent part of the fuzzy rules are identified through a hebbian approach [Lin & Lee 91][Quek & Zhou 96]; or, the rule layer of the system is fully connected to the consequent layer [Song et. al. 09]. While the consequent of a fuzzy rule is unique in the former approach, the fuzzy rules formed in the latter approach are contradictory in nature. This means very different, and possibly conflicting, consequences can be inferred using a single fuzzy rule. As a result, the resultant computational structure of the neuro-fuzzy system is opaque and the system functions as a black-box. This essentially defies one of the key reasons in performing modeling using a neuro-fuzzy system, which is to capitalize on the transparent nature of the system to understand and explain the dynamics of the application problem.

Simultaneous antecedent and consequent identification – identifies the antecedent and consequent segments of a fuzzy rule through a single pass of the training data block, where the best matched membership function in each respective input-output space is chosen. Some examples of neuro-fuzzy systems adopting this approach are [Kim & Kasabov 99][Tung & Quek 02]. Subsequently, the issue of possible conflicting/contradictory rules has to be addressed.

Figure 2.10: Flowchart of online learning in neuro-fuzzy systems.

Summarizing, the following traits are desirable for a fuzzy rules generation mechanism in a neuro-fuzzy system:

1. The set of fuzzy rules identified should be compact, where rules are created only when necessary.
2. The issue of contradicting rules in the system should be addressed, such that the resultant system is unique and conflict-free in order to provide a logical and intuitive description to the application problem.

2.3.3 Evolving neuro-fuzzy systems

When learning in an online mode, data arrives in a sequential manner and the neuro-fuzzy system performs learning on one data tuple at a time. Figure 2.10 shows a generic flowchart of designing a neuro-fuzzy system via online learning. As seen, for a current data tuple presented at a particular time instance, it goes through a cycle of structure and parameter learning. Structure learning for the data tuple involves a comparison with existing knowledge in the system, where either: (1) new clusters are created in the input-

output spaces of the system to represent the novel data tuple; or (2) existing clusters are modified to accommodate the data tuple. A new rule might be created if the current data tuple is found to be novel compared to the existing knowledge encrypted in the neuro-fuzzy system. Following the parameter learning on the current data tuple, the neuro-fuzzy system continues learning when a new data tuple is presented to the system; or, the system is employed to perform prediction on some unseen data.

Comparing the offline learning mode (see Figure 2.9) and online learning mode (see Figure 2.10) in a neuro-fuzzy system, there are two noticeable differences. First, the knowledge from the entire training data block is utilized for the design (i.e. structure and parameter learning) of a batch neuro-fuzzy system; while the knowledge from a single training data pair is utilized for the design of an evolving neuro-fuzzy system at each time instance. Second, a batch neuro-fuzzy system is ready as a modeling tool on unseen data after the whole design phase is over, and a new model has to be built with the updated training dataset when subsequent new training data are available; while an evolving neuro-fuzzy system can be used for modeling at anytime, where new clusters and rules can be incorporated to the existing structure when subsequent new training data are available.

As explained previously (see Section 2.3.1), online learning is generally more sensitive to the presence of noise because the effect of noise is not averaged over the entire training dataset [Baraldi & Blonda 99a]. On the other hand, most existing evolving neuro-fuzzy systems in the literature are Type-1 fuzzy models, where the fuzzy sets employed in the systems are of Type-1 fuzzy sets. Table 2.1 presents a summary of some existing evolving neuro-fuzzy systems in the literature. Since most of the existing evolving systems are Type-1 models as seen from the table, they may not perform satisfactorily under noisy learning environment. This is because Type-1 fuzzy sets are certain in the sense that the membership grades of the membership functions are crisp numbers, which might not be able to consider

Table 2.1: A summary of evolving neuro-fuzzy systems in the literature.

	TSK systems	(Publication Year)	Mamdani systems	(Publication Year)
Type-1	SONFIN	('98) [Juang & Lin 98]	FALCON-ART	('97) [Lin & Lin 97]
	NeuroFAST	('01) [Tzafestas & Zikidis 01]	EFuNN	('01) [Kasabov 01]
	GD-FNN	('01) [Wu, Er & Gao 01]	eFSM	('10) [Tung & Quek 10]
	DENFIS	('02) [Kasabov & Song 02]		
	eTS	('04) [Angelov & Filev 04]		
	SAFIS	('06) [Rong et. al. 06]		
Type-2	SEIT2FNN	('08) [Juang & Tsao 08a]	T2SONFS	('08) [Juang & Tsao 08b]
	ORGQACO	('09) [Juang & Hsu 09]		
	IT2FNN-SVM	('10) [Juang, Huang & Cheng 10]		

the effect of uncertainties caused by the presence of noise [Hagras 04][Sepúlveda et. al. 07]. On the other hand, Type-2 models are extensions of Type-1 models where the fuzzy sets used are Type-2 fuzzy sets. Since the membership grades of the membership functions are fuzzy, Type-2 fuzzy models possess the potential to address the modeling bottlenecks of its Type-1 counterparts [Mendel & John 02]. Consequently, recent efforts have been directed to extend evolving Type-1 neuro-fuzzy systems to Type-2 models as observed from Table 2.1.

Although evolving Type-2 neuro-fuzzy systems are gaining attention only in recent times, most works focus on TSK models. Despite being listed as an evolving Type-2 Mamdani neuro-fuzzy system, T2SONFS [Juang & Tsao 08b] adopts a sequential learning mode, but it is not an entirely online system since the input values have to be normalized prior to the design of the system. As explained in [Ang & Quek 05], fuzzy modeling is generally divided into two areas: (1) linguistic fuzzy model that is focused on interpretability, mainly consisting of Mamdani models; and (2) precise fuzzy model that is focused on accuracy, mainly consisting of TSK models. Hence, it is noted that there are currently minimum parallel effort in the literature concerning the design of an evolving Type-2 Mamdani neuro-fuzzy system.

On the other hand, existing evolving Type-2 neuro-fuzzy systems follow the flowchart as illustrated in Figure 2.10 for the design of the systems. From the flowchart, it is observed that new clusters and new rules are continuously added into the neuro-fuzzy system as the system continues learning on new incoming training data. This means that the computational structure of the system can only expand over time as learning continues. Subsequently, two major problems may occur: (1) an ever-expanding knowledge structure; and (2) a lack of interpretability of the resultant system. Since life-long learning of new data is possible for a neuro-fuzzy system learning in an online mode where new clusters and new rules are continuously incorporated to the existing structure, steps are needed to keep the size of the

system in check. This is especially important because an overly complex system requires much computational time and memory space, which eventually defeats one of the key purposes of adopting online learning – to avoid storage of the entire training data so as to lessen the time and memory needed during learning. In addition, an ever-expanding knowledge structure also leads to degradation of the interpretability in the system. As defined in [Guillaume 01], interpretability of a neuro-fuzzy system involves two conditions; namely, (1) the fuzzy partition must be readable such that membership functions in the input-output spaces can be interpreted as linguistic labels with clear semantic meanings; and (2) the set of fuzzy rules should be compact with good generalization abilities. As more clusters are created in the input-output spaces of an online neuro-fuzzy system, highly over-lapping or highly similar clusters might be present. This not only causes the fuzzy partition to be unreadable, it also adds to the computational complexity of the system.

Summarizing, the following traits are desirable for a learning mechanism in an evolving neuro-fuzzy system:

1. The system should be robust when learning in the presence of noise, since online learning is generally more sensitive to noise present in the learning environment.
2. The resultant structure of the system should be compact and readable, where the issues of an ever-expanding structure and the lack of interpretability due to highly over-lapping clusters should be addressed.

2.4 Summary

This chapter reviews the topics of neural networks, fuzzy systems, and neuro-fuzzy systems. Basic components in areas of neural networks and fuzzy systems are briefly introduced; while neuro-fuzzy systems, which are hybrid modeling tools synergized between neural net-

works and fuzzy systems, are reviewed in greater details. More importantly, the interest behind designing neuro-fuzzy systems from data is explained. Following that, two learning paradigms of learning from data are introduced. The two classes of neuro-fuzzy systems adopting the respective learning paradigms – namely, batch neuro-fuzzy systems and evolving neuro-fuzzy systems, which are significant to this work, are then discussed in length. Desirable traits in the learning mechanisms for the batch and evolving neuro-fuzzy systems are also highlighted.

The prevailing issues in batch neuro-fuzzy systems stem from the techniques used in the fuzzy partitioning and fuzzy rulebase identification of existing systems. Firstly, some existing fuzzy partitioning techniques require knowledge regarding the number of input-output clusters prior to the design of the systems. By heuristically initializing a neuro-fuzzy system with a fixed number of input-output clusters, the system is unable to learn and incorporate new knowledge from incoming training data. In addition, the fixed number of input-output clusters might not be sufficient in modeling the training data, leading to a subsequent loss of accuracy. Secondly, some existing neuro-fuzzy systems are unable to extract meaningful fuzzy rules from the systems to explain the dynamics of the application problem. This is because the fuzzy rulebase encoding the knowledge in the neuro-fuzzy system is inconsistent, where a single fuzzy rule can result in two different, and possibly conflicting, outcomes. Hence, a learning mechanism that is free of these drawbacks is desirable for deriving the knowledge base of a batch neuro-fuzzy system.

On the other hand, the prevailing issues in evolving neuro-fuzzy systems are the results of online learning in a noisy and time-varying environment. Most existing evolving neuro-fuzzy systems fail to address a byproduct of online learning, which is the vulnerability to noise. Although recent works have tried to consider this issue by incorporating Type-2 fuzzy systems when modeling noisy applications, they suffer from a general lack of interpretability of

the systems. This is because the knowledge bases of these systems are continuously expanding as life-long learning continues on new incoming data, such that no mechanism is in place to keep the resultant systems compact. In addition, highly over-lapping clusters might be created in the input-output spaces during continuous learning, aggravating the problem of lack of interpretability. Hence, a learning mechanism that is robust when performing online learning in a noisy and time-varying environment is desirable for the design of an evolving neuro-fuzzy system.

Chapter 3

Self-adaptive Fuzzy Inference Network

Neuro-fuzzy systems are hybrid systems that capitalize on the functionalities of fuzzy systems and neural networks [Nauck, Klawonn & Kruse 97]. The black-box nature of a neural network can be resolved by injecting the interpretability of a fuzzy system into the connectionist structure; while marrying the learning powers of a neural network into a fuzzy system enables the system to automatically refine its parameters. Despite these improved advantages over its individual predecessors, there is a main concern when designing a neuro-fuzzy system – to derive the knowledge base of the system. This can be further split into two issues; namely, the fuzzy partitioning of the input and output spaces, and the rule generation scheme of the system [Lin 94]. Fuzzy partitioning for a neuro-fuzzy system determines the numbers, the positions and the spreads of the linguistic labels in each input-output dimension; while the rule generation scheme determines the set of fuzzy rulebase/knowledge base governing the neuro-fuzzy system.

One approach to design a neuro-fuzzy system is to perform batch learning on numerical

data. This means that the training dataset is available prior to the design of the neuro-fuzzy system, where learning is performed through a repetitive cycling of the training data. Subsequently, both the fuzzy partitioning and the fuzzy rulebase can be automatically extracted from the numerical training data. Due to this automated and self-reliant design, batch neuro-fuzzy systems are also referred to as *self-organizing* neuro-fuzzy systems. Nevertheless, existing self-organizing neuro-fuzzy systems suffer from one or more of the following major drawbacks: (1) an inconsistent rulebase; (2) the need for prior knowledge such as the number of clusters to be computed; (3) heuristically designed knowledge acquisition methodologies; and (4) the stability-plasticity trade-off of the system.

Firstly, a consistent and unique rulebase allows a logical and intuitive interpretation of the knowledge represented within the computational structure of the neuro-fuzzy system [Casillas et. al. 03]. On the other hand, an inconsistent rulebase occurs when there exists two rules in the rulebase of the system such that the precedent conditions are similar but the resultant consequences differ [Jin, Von Seelen & Sendhoff 99]. Subsequently, totally conflicting outcomes may appear when the system encounters an input that simultaneously triggers the activations of the two rules. This inconsistent encryption of knowledge not only causes confusion for the human users, it also results in a meaningless and obscure description to the computational structure of the neuro-fuzzy system. This is functionally incompatible with the original motivation of developing a neuro-fuzzy system, where a transparent learning model is desired.

A second important consideration in the design of a neuro-fuzzy system is the choice of fuzzy partitioning technique adopted. Although numerical methods such as fuzzy Kohonen partitioning [Bezdek, Tsao & Pal 92], fuzzy C-means [Bezdek 81], linear vector quantization [Kohonen 82], among others [Wang & Mendel 92][Du 10] have been commonly used in fulfilling this task, these numerical methods still require prior knowledge regarding the

number of clusters necessary in each input-output dimension. Such information is often arbitrarily determined at the beginning and the numbers of clusters remain fixed throughout the entire learning process. Consequently, the system might not be able to achieve a desirable performance for the application problem due to either an over- or under-generalization restricted by the rigid amount of resources made available in the system. In addition, the pre-defined number of clusters in each input-output dimension also deprives the neuro-fuzzy system from subsequently incorporating new clusters. This is an intricate situation known as the stability-plasticity trade-off [Grossberg 82] where a regulated balance needs to be maintained for the co-existence of past knowledge and any future knowledge such that a current and up-to-date system is achieved for the modeling of the application environment.

Hence, this chapter presents the *Self-adaptive Fuzzy Inference Network* (SaFIN), a novel self-organizing neuro-fuzzy system, which addresses the above-stated deficiencies. A fully data-driven approach is employed for the automated formulation of fuzzy rulebase, which maintains a consistent knowledge encryption in SaFIN. In addition, SaFIN employs a new clustering technique known as the *Categorical Learning Induced Partitioning* (CLIP) inspired from the behavioral category learning process demonstrated by humans. CLIP is a single-pass fuzzy partitioning technique that performs clustering of the input-output dimensions based on knowledge extracted from each training tuple. By considering the significance of the incoming tuple with respect to any existing clusters in the input-output dimensions, the SaFIN model is able to automatically learn the presence of a new cluster. As a result, it frees the human user from pre-determining the number of clusters present; while a balance is coordinated between new knowledge from the current incoming training tuple and existing knowledge in the system such that old and new knowledge is able to co-define the structure of the model. Unlike previous heuristic clustering approaches proposed in the neuro-fuzzy paradigm [Kasabov 01][Kasabov & Song 02][Tung & Quek 02][Angelov & Filev 04][Rong et. al. 06][Tung & Quek 10], the SaFIN model employs a top-down approach in

its clustering mechanism which draws inspiration from the human-based category learning process [Mandler & Bauer 88]–[Mandler 08].

The chapter is organized as follows. The computational structure and the reasoning process of the SaFIN model are described in Section 3.1. The proposed learning mechanisms of SaFIN are introduced in Section 3.2. Section 3.3 evaluates the learning and self-organizing abilities of SaFIN through a series of standard benchmark experimental simulations. Lastly, Section 3.4 provides a summary of the key contributions in this chapter.

3.1 SaFIN: Self-adaptive fuzzy inference network

This section describes the computational structure and the reasoning process of the proposed *Self-adaptive Fuzzy Inference Network* (SaFIN).

3.1.1 Architecture of SaFIN

The proposed SaFIN model is a five layers neuro-fuzzy system as shown in Figure 3.1. Layer 1 consists of the input (variable) nodes; layer 2 is the antecedent nodes; layer 3 is the rule nodes; layer 4 consists of the consequent nodes; and layer 5 is the output (variable) nodes. In the SaFIN model, the input vector is denoted as $x = [x_1, \dots, x_p, \dots, x_P]^T$. The corresponding desired output vector is denoted as $d = [d_1, \dots, d_q, \dots, d_Q]^T$; while the computed output is denoted as $y = [y_1, \dots, y_q, \dots, y_Q]^T$. The notations used in Figure 3.1

Figure 3.1: Architecture of the Self-adaptive Fuzzy Inference Network (SaFIN).

are defined as follows:

- P number of input dimensions ;
- Q number of output dimensions ;
- I_p p -th input node ;
- O_q q -th output node ;
- J_p number of fuzzy clusters in I_p ;
- L_q number of fuzzy clusters in O_q ;
- A_{j_p} j -th antecedent fuzzy cluster in I_p ;
- C_{l_q} l -th consequent fuzzy cluster in O_q ;
- K number of fuzzy rules ;
- R_k k -th fuzzy rule .

Layer 3 of SaFIN encrypts the rulebase of the system where each rule node encodes an IF-THEN Mamdani-type fuzzy rule [Mamdani 77] given as in (3.1):

$$R_k : \text{IF } x_1 \text{ is } A_{j_1}^{(k)} \text{ and } \dots \text{ and } x_P \text{ is } A_{j_P}^{(k)} \quad \text{THEN } y_1 \text{ is } C_{l_1}^{(k)} \text{ and } \dots \text{ and } y_Q \text{ is } C_{l_Q}^{(k)} \quad (3.1)$$

where $A_{j_p}^{(k)}$ (resp. $C_{l_q}^{(k)}$) is the j -th antecedent (resp. l -th consequent) node associated with the p -th input (resp. q -th output) variable that is connected to the rule node R_k . The tunable parameters of SaFIN are the centres and widths of the fuzzy labels embedded in the antecedent and consequent nodes, where each node defines a gaussian membership function described as in (2.9). For convenience, it is re-listed in (3.2):

$$\mu(c, \sigma; x) = e^{-((x-c)^2/\sigma^2)} \quad (3.2)$$

such that c and σ are the centre and width of the function respectively (see Figure 2.3). Adaptation of the parameters is performed using the neural network-based gradient descent approach [Haykin 98]. Please refer to Appendix A for the computations of the back-propagation.

3.1.2 Reasoning process of SaFIN

As seen from Figure 3.1, the reasoning process of the SaFIN model is represented by solid arrows where the input vector x is presented to the system at layer 1. The proposed system then performs inference based on the input vector by propagating the information through layers 2 to 4. Consequently, the system produces a computed output vector y at layer 5. The details on the reasoning process of SaFIN are discussed here.

The generic operations for the proposed SaFIN model are defined as follows: the activation

functions of each layer $M \in \{1 \dots 5\}$ are denoted as $a^{(M)}$, and the corresponding output for an arbitrary node is denoted as o .

Layer 1: The function of the input nodes is to directly pass on the input vector to the next layer. Hence, the neural operation of I_p can be described as in (3.3):

$$o_p = a^{(1)}(x_p) = x_p . \quad (3.3)$$

Layer 2: The fuzzy labels A_{j_p} define the antecedent segments of the Mamdani-type rules described as in (3.1) where each label is defined as a gaussian function as described in (3.2). The function of layer 2 of SaFIN is to perform antecedent matching of the input training value with the respective input labels such that the degree of similarity between x_p and the membership function embedded in A_{j_p} is computed. Hence, the neural computation of A_{j_p} can be described as in (3.4):

$$o_{j_p} = a^{(2)}(o_p) = \mu_{j_p}(c_{j_p}, \sigma_{j_p}; x_p) \quad (3.4)$$

where $\mu_{j_p}(c_{j_p}, \sigma_{j_p}; x)$ refers to the gaussian membership function embedded in A_{j_p} .

Layer 3: The set of Mamdani-type rules that is induced from the training data is defined in the rule layer of the SaFIN model. Each rule node R_k computes the overall degree of similarity between the input training vector and the antecedent part of the k -th fuzzy rule. Hence, the firing rate of R_k is computed as in (3.5), where

$$o_k = a^{(3)}(o_{j_1}^{(k)}, \dots, o_{j_P}^{(k)}) = \min_{p \in \{1 \dots P\}} o_{j_p}^{(k)} . \quad (3.5)$$

Layer 4: This layer of SaFIN consists of the fuzzy labels C_{l_q} that define the consequent segments of the Mamdani-type fuzzy rules in the system. The function of layer 4 of the

system is to perform consequent derivation for the fuzzy rules based on the information from the input vector x . Since C_{l_q} may serve as output to more than one fuzzy rule, the cumulative neural computation for C_{l_q} can be described as in (3.6):

$$o_{l_q} = a^{(4)} \left(o_1^{(l_q)}, \dots, o_{K_{l_q}}^{(l_q)} \right) = \max_{k \in \{1 \dots K_{l_q}\}} o_k^{(l_q)} \quad (3.6)$$

where K_{l_q} is the total number of fuzzy rules in SaFIN that shares the same consequent node C_{l_q} and $o_k^{(l_q)}$ is the output of the k -th fuzzy rule that shares C_{l_q} .

Layer 5: The function of the output nodes is to perform defuzzification to obtain a crisp output value. This is achieved using the centre of averaging method [Lin & Lee 96] such that the neural operation of O_q can be described as in (3.7):

$$\begin{aligned} y_q = o_q &= a^{(5)} (o_{1_q}, \dots, o_{L_q}) \\ &= \frac{\sum_{l_q \in \{1_q \dots L_q\}} o_{l_q} c_{l_q} \sigma_{l_q}}{\sum_{l_q \in \{1_q \dots L_q\}} o_{l_q} \sigma_{l_q}} \end{aligned} \quad (3.7)$$

where c_{l_q} and σ_{l_q} are the centre and width of the gaussian function embedded in C_{l_q} respectively.

3.2 Learning mechanisms in SaFIN

There are two main components to the learning process of the proposed SaFIN model; namely, the fuzzy partitioning of the input-output dimensions and the rule generation procedure. Initially, there are neither fuzzy partitioning in the input-output spaces nor fuzzy rules in the system, i.e. , there are no nodes in hidden layers 2–4. A localized learning of the fuzzy labels is carried out such that the number of clusters in each input-output dimension can differ depending on the knowledge extracted from the numerical data. The

numbers, the positions and the spreads of the fuzzy labels are self-determined from the training dataset. New clusters are incorporated into the system when the knowledge extracted from the incoming training tuple is novel as compared to existing clusters in the system. Refinements are then made to the existing clusters such that old knowledge in the system and new knowledge from the incoming training tuple can co-exist to provide a more accurate representation of the training data. This tailored approach addresses the stability-plasticity of SaFIN. Hence, these advantages over existing techniques serve as the main motivations to the development of the Categorical Learning Induced Partitioning (CLIP) technique employed in the SaFIN model. As observed by Mandler et.al. in [Mandler, Bauer & McDonough 91], global conceptual categories such as *animals* and *vehicles* are the first level of distinctions made by young infants during category learning. Subsequently, high-contrast basic-level distinctions within each domain (for example, *dogs* versus *fish*, or *cars* versus *airplanes*) are being observed in older infants. This is then followed by moderate-contrast (*dogs* versus *rabbits*, or *cars* versus *motorcycles*) and low-contrast (*dogs* versus *horses*, or *cars* versus *trucks*) basic-level distinctions demonstrated in older children while performing category learning. By employing a similar divisive top-down approach, the CLIP technique draws inspiration from the behavioral category learning process exhibited by humans. (This is an efficient method of rapid knowledge acquisition that helps to expand the initial conceptual system formulated during infancy [Mandler, Bauer & McDonough 91].) Table 3.1 summarizes the desirable characteristics in CLIP as compared to existing clustering techniques commonly adopted in neuro-fuzzy systems; namely, linear vector quantization (LVQ) [Kohonen 82]; fuzzy C-means (FCM) [Bezdek 81]; fuzzy Kohonen partitioning (FKP) [Bezdek, Tsao & Pal 92]; adaptive resonance theory (ART) [Carpenter et. al. 92]; and discrete incremental clustering (DIC) [Tung & Quek 02]. Details on the proposed self-organizing CLIP clustering technique will be presented in Section 3.2.1.

The second key component in the design process of SaFIN is the formulation of the rulebase.

Table 3.1: A comparison between clustering techniques.

Technique	No prior knowledge	Clusters added incrementally	Interpretable labels
LVQ	×	×	×
FCM	×	×	×
FKP	×	×	×
ART	✓	✓	×
DIC	✓	✓	×
CLIP	✓	✓	✓

By employing a self-automated rule generation mechanism, no initial rulebase needs to be pre-specified by the human expert. Firstly, a fuzzy rule is formulated to capture the knowledge from each of the incoming training tuples. Following that, each of the fuzzy rules in the rulebase is assigned a weightage depicting its significance in the modeling of the application environment. Conflicting rules with low influences are deemed as outliers and are subsequently deleted from the system. This approach ensures that the proposed SaFIN model maintains a consistent rulebase that is able to provide an aptly description to the application problem. Details on the proposed self-automated rulebase formation mechanism will be presented in Section 3.2.2.

3.2.1 Self-organizing clustering in SaFIN

Learning in the proposed SaFIN model begins with extracting and utilizing knowledge from each incoming training tuple to establish an initial fuzzy partitioning. Figure 3.2 shows a flowchart of the self-organizing clustering technique, CLIP, adopted in SaFIN. With the arrival of each training tuple $[x, d]$, fuzzy partitioning is performed independently for each input-output dimension. If there is no existing fuzzy partitioning in an input dimension, then a new cluster spanning the entire domain is formed. This conforms to the initial global conceptual categories conceived by humans. For example, young infants first see a dog (a

Figure 3.2: Flowchart of the self-organizing clustering technique CLIP adopted in SaFIN.

particular instance of the global concept *animal*) and associate the entire domain of *animal* as a dog. It is only much later in the process of category learning that the basic concept *dog* is established [Mandler 08]. By translating this initial knowledge of the global concept, the formation of the first fuzzy cluster A_{1_p} in an input dimension p can be described using (3.8):

$$\begin{aligned} c_{1_p} &= x_p \\ \sigma_{1_p} &= R \left(\sqrt{-\frac{(\min_p - x_p)^2}{\log \alpha}}, \sqrt{-\frac{(\max_p - x_p)^2}{\log \alpha}} \right) \end{aligned} \quad (3.8)$$

where c_{1_p} and σ_{1_p} are the centre and width of the gaussian membership function embedded in A_{1_p} . A newly created membership function is centered upon the presented value; while $R(\sigma_1, \sigma_2) = 1/2 (\sigma_1 + \sigma_2)$ defines a *regulator function* that ensures a fuzzy cluster has distinct semantic meaning by maintaining a reasonable amount of buffer on either sides of its centre. Here, the boundary for the domain is given as $[\min_p, \max_p]$. The *minimum membership threshold* α is defined such that the membership value of any point in the domain should be at least α before regulation. The same initialization procedure is repeated for each output dimension.

Figure 3.3 shows the formation of an initial cluster in an input-output dimension. For the special case where the presented value coincides with the midpoint of the domain, the corresponding fuzzy partitioning is depicted in Figures 3.3(a) and (b). Since the centre of the newly created cluster is equi-distance from the lower and upper bounds, the membership function obtained before and after regulation is similar where the boundary points have the minimum membership value α . On the other hand, for an initial cluster created in an input-output dimension where the centre is skewed from the midpoint of the domain, the spreads of the cluster on either sides of the centre depend on the distance of the centre to the respective boundary points. This is illustrated in Figure 3.3(c). As seen from the figure,

Figure 3.3: Initialization of the self-organizing clustering technique performed in an input-output dimension of SaFIN.

the spread on the side of the cluster nearer to a boundary point can be significantly smaller compared to the opposing side which is further away from the other boundary point. In the extreme case where the first presented value coincides with one of the boundary points, there will be no spread on one side of the cluster while the opposing side has a very wide spread. Such clusters deter a clear interpretation of the knowledge encoded in the proposed SaFIN model. Hence, regulation is performed to ensure that each cluster created in an input-output dimension maintains its relevance and a distinct semantic meaning. Figure 3.3(d) shows the resultant cluster after performing regulation on the initially created cluster depicted in Figure 3.3(c). As clearly shown, an equal and reasonable amount of spread is maintained on both sides of the cluster centre to ensure that the cluster has clear semantic meaning.

On the other hand, if there is existing fuzzy partitioning in an input-output dimension, then SaFIN proceeds with computing the similarities between the presented value and existing clusters. As seen from (3.4), the similarity match between an input value x_p and an existing cluster in the p -th input dimension is given as $SM(x_p, A_{j_p}) = \mu_{j_p}(c_{j_p}, \sigma_{j_p}; x_p)$. Subsequently, SaFIN identifies the best matched fuzzy cluster in an input dimension via the

computed similarity values, i.e., the best matched fuzzy cluster is denoted as $A_{j_p^*}$ where $j_p^* = \arg \max_{j_p} SM(x_p, A_{j_p})$. If the similarity match between the identified best matched cluster and the presented value in an input dimension exceeds a *contrasting threshold* β , the best matched cluster is deemed as being able to give a satisfactory description of the presented value. However, if the similarity match falls below β , then a new cluster is created in the input dimension based on the presented value. Adjustments and refinements are subsequently made to the existing clusters in that dimension of SaFIN to incorporate the newly created cluster. This process effectively addresses the stability-plasticity of SaFIN by allowing the old knowledge to co-exist with the new information derived from the training data to define the rulebase. Specifically, the existing concepts (fuzzy sets) in the input dimension are refined and adapted to enhance their representations of the training data encountered. As stated in [Mandler 08], “every time a new category is learned within the global domains, it is necessarily learned as a subdivision of this larger division”. By comparing a new exemplar with existing concepts (fuzzy sets) in an input dimension via the computed similarity values, a conceptually distinct category will be formed by SaFIN when a prominent distinction is observed. This divisive top-down category learning mechanism observed in humans is emulated for the creation of a new fuzzy cluster in the proposed SaFIN model. The formation of a new fuzzy cluster $A_{J_p(t)+1}$ in the p -th input dimension can be described using (3.9):

$$\begin{aligned}
 c_{J_p(t)+1} &= x_p \\
 \sigma_{J_p(t)+1} &= \begin{cases} \sigma^R & \text{if } j_p^L = \text{NULL} \\ \sigma^L & \text{if } j_p^R = \text{NULL} \\ R(\sigma^R, \sigma^L) & \text{otherwise} \end{cases} \quad (3.9)
 \end{aligned}$$

where

$$\sigma^R = R \left(\sqrt{-\frac{(c_{j_p^R} - x_p)^2}{\log \alpha}}, \sigma_{j_p^R}(t) \right) ; \quad \sigma^L = R \left(\sqrt{-\frac{(c_{j_p^L} - x_p)^2}{\log \alpha}}, \sigma_{j_p^L}(t) \right) .$$

The immediate left and right neighbours of the newly created cluster are denoted as $A_{j_p^L}$ and $A_{j_p^R}$ respectively, where

$$j_p^L = \begin{cases} \text{NULL} & \text{if } c_{j_p} \geq x_p \text{ for } 1 \leq j_p \leq J_p(t) \\ \arg \min_{c_{j_p} < x_p} |c_{j_p} - x_p| & \text{otherwise} \end{cases}$$

$$j_p^R = \begin{cases} \text{NULL} & \text{if } c_{j_p} \leq x_p \text{ for } 1 \leq j_p \leq J_p(t) \\ \arg \min_{c_{j_p} > x_p} |c_{j_p} - x_p| & \text{otherwise} \end{cases} .$$

For simplicity, only refinements are made to immediate left and right neighbours of the newly created cluster. Refinements made can be classified into three cases:

1. If the newly created cluster has no left neighbour (i.e. $j_p^L = \text{NULL}$), then only the right neighbour is updated:

$$\sigma_{j_p^R}(t+1) = \sigma_{J_p(t)+1} .$$

2. If the newly created cluster has no right neighbour (i.e. $j_p^R = \text{NULL}$), then only the left neighbour is updated:

$$\sigma_{j_p^L}(t+1) = \sigma_{J_p(t)+1} .$$

3. If the newly created cluster has both left and right neighbours, then they are updated:

$$\sigma_{j_p^L}(t+1) = \sigma_{j_p^R}(t+1) = \sigma_{J_p(t)+1} .$$

Figure 3.4: Clustering steps performed in SaFIN after initialization.

The same fuzzy clustering process is performed for each output dimension.

Figure 3.4 illustrates the fuzzy clustering process in an input-output dimension of the proposed SaFIN model. After an initial cluster has been formed as in Figure 3.3(d), SaFIN continues to perform clustering when a novel data point is encountered, i.e., the computed similarity values between the presented point and the existing clusters fall below β . This is illustrated as Figure 3.4(a), where the novel data point has only right neighbour. A new cluster is then created using the information derived from this novel data point as seen in Figure 3.4(b). Since the distance of the presented value from the lower bound is significantly smaller compared to that from the upper bound, the spreads on either sides of the cluster centre are unbalanced. As explained before, this will decrease the interpretability of the knowledge encoded in the proposed SaFIN model. Hence, regulation is performed to

preserve a distinct semantic meaning of the newly created cluster as seen from Figure 3.4(c). The existing cluster is simultaneously refined to incorporate the newly created cluster in this figure. Following that, Figure 3.4(d) shows the arrival of a novel data point to the input-output dimension, where the presented value has immediate left and right neighbours. A new cluster is created such that it is centered upon the presented value in Figure 3.4(e). The spreads of the cluster on either sides of the centre depend on the distance of the centre to the respective centres of its immediate neighbours. The centres of the immediate neighbours have the minimum membership value α . Finally, to preserve the semantics of the newly created cluster and to incorporate the new cluster into the set of existing fuzzy clusters, regulation and refinements are performed in Figure 3.4(f).

3.2.2 Self-automated rule generation in SaFIN

The proposed SaFIN model proceeds with rule generation after achieving fuzzy partitioning for the input-output spaces. Figure 3.5 shows the two stages of the proposed self-automated rule generation mechanism; namely, (1) rule creation; and (2) consistency check. Rule creation formulates an initial rulebase using information derived from the training dataset. With the arrival of each training tuple $[x, d]$, the best matched fuzzy cluster is found for each input-output dimension (see computations in Section 3.2.1). The best matched fuzzy cluster in the p -th input dimension is denoted as $A_{j_p}^*$; while the best matched fuzzy cluster in the q -th output dimension is denoted as $C_{l_q}^*$. Subsequently, a fuzzy rule R^* is formulated where $\{A_{j_p}^*\}_{p=1}^P$ and $\{C_{l_q}^*\}_{q=1}^Q$ are the antecedent and consequent segments of R^* respectively. If R^* is novel (i.e. there is no existing rule in the knowledge base of the system such that the antecedent and consequent segments are similar to R^*), then it is inserted into the rulebase such that the new rule $R_{K(t)+1} = R^*$. By performing novelty check, SaFIN is ensured a unique rulebase. The new rule $R_{K(t)+1}$ is then assigned an initial weightage of $Wt_{K(t)+1} = 1$. In the SaFIN model, weightage of a fuzzy rule depicts its

Figure 3.5: Flowchart of the self-automated rule generation mechanism adopted in SaFIN.

significance in modeling the application environment. Hence, the higher the weightage of a fuzzy rule, the greater is its potential in modeling the application environment. On the other hand, R^* is not inserted into the rulebase if it is not novel. That is, there exists a fuzzy rule R_{k^*} in SaFIN such that R_{k^*} has the same antecedent and consequent segments as R^* . Subsequently, the weightage of R_{k^*} is enhanced as $Wt_{k^*}(t+1) = Wt_{k^*}(t) + 1$.

Following the formulation of an initial rulebase, the proposed SaFIN model performs consistency check on the rulebase. The design process of SaFIN is completed if the rulebase formulated is verified to be consistent. On the other hand, inconsistent rules with lower weightages are deleted for an inconsistent rulebase, i.e., if R_{k_1} and R_{k_2} are two rules in SaFIN with similar precedent conditions but differing consequences such that $Wt_{k_1} < Wt_{k_2}$, then $\{R_k\}_{k=1}^{K(t+1)} = \{R_k\}_{k=1}^{K(t)} \setminus R_{k_1}$. This approach of retaining the fuzzy rule with the strongest weightage not only ensures that the resultant rulebase is consistent, it also ensures that the proposed system provides a most aptly description of the application environment. Finally, some of the fuzzy labels might be “orphaned” when all fuzzy rules associated with them have been deleted. For an input dimension p , an “orphaned” fuzzy label is identified by $A_{j_p^o} \notin R_k$ for all k . Subsequently, the “orphaned” label $A_{j_p^o}$ is removed to ensure that the resultant computational structure of SaFIN is compact, i.e., $\{A_{j_p}\}_{j_p=1_p}^{J_p(t+1)} = \{A_{j_p}\}_{j_p=1_p}^{J_p(t)} \setminus A_{j_p^o}$. The same deletion process is performed for each output dimension.

3.3 Experimental results and analysis

This section illustrates the learning and self-organizing abilities of the proposed SaFIN model by employing it in three benchmarking datasets; namely, (1) identification of a non-linear system [Juang & Lin 98]; (2) the Nakanishi dataset [Nakanishi, Turksen & Sugeno 93][Sugeno & Yasukawa 93]; and (3) the UCI dataset [UCI].

Figure 3.6: Input spaces fuzzy partitioning for the identification of a nonlinear system in: (A) SaFIN; (B) SONFIN *before* alignment; and (C) SONFIN *after* alignment.

3.3.1 Example 1 – Identification of a nonlinear system

This experiment generalizes the dynamics of a nonlinear system, where the dataset is generated by a difference equation as described in (3.10):

$$y(t+1) = \frac{y(t)}{1 + y^2(t)} + u^3(t) \quad (3.10)$$

such that the present output of the system, $y(t+1)$, depends nonlinearly on its past output, $y(t)$, and an input, $u(t) = \sin(2\pi t/100)$. Following the description in [Juang &

Lin 98], 50000 training and 200 testing data pairs are generated with initial conditions $(u(0), y(0)) = (0, 0)$. Subsequently, the SaFIN model is applied to this identification problem with $\alpha = 0.2$ and $\beta = 0.6$. Four fuzzy labels are identified by the CLIP algorithm for each of the input-output spaces respectively. Figure 3.6 illustrates the identified fuzzy clusters in the input spaces $u(t)$ and $y(t)$. The respective fuzzy partitioning of the input spaces for the original problem derived by the SONFIN model [Juang & Lin 98] have also been included as a comparison. As clearly seen from Figure 3.6(A), the fuzzy clusters identified in SaFIN are highly ordered and have clear distinct semantic meanings with respect to the attached fuzzy labels (i.e. VL–Very Low, L–Low, H–High, VH–Very High). Comparatively, Figure 3.6(B) shows the initial identified fuzzy clusters in SONFIN. As seen, the fuzzy clusters are highly over-lapping, making it difficult to induce any clear semantic meanings to the derived clusters. To tackle this problem, SONFIN performs an additional step to compute the similarity of a newly formed cluster with existing clusters in the input spaces, and subsequently align the new cluster. This improved result is shown in Figure 3.6(C). Although the number of fuzzy clusters identified in the input spaces have reduced (from an initial 10 fuzzy clusters to 5 fuzzy clusters in $u(t)$; and an initial 10 fuzzy clusters to 7 fuzzy clusters in $y(t)$), the resultant fuzzy clusters still have a significant amount of over-lap as seen in the first two clusters of the input space $y(t)$. From this illustration, a key strength of a Mamdani neuro-fuzzy system is well-demonstrated in SaFIN, where intuitive semantic labels can be directly identified from the raw numerical data for the input-output spaces. This makes the derived membership functions in the input-output spaces human-interpretable and readable. On the other hand, TSK models such as SONFIN generally report large over-lapping regions in the identified fuzzy clusters where one can hardly induce any underlying data structure from the observed fuzzy sets [Tung & Quek 10].

Following the fuzzy partitioning of the input-output spaces, SaFIN uses the derived semantic

Figure 3.7: (a) Identified fuzzy rulebase; and (b) Modeling results on testing dataset for the identification of a nonlinear system using SaFIN.

fuzzy labels to identify eight fuzzy rules via the self-automated rule generation mechanism. Figure 3.7(a) lists the identified fuzzy rulebase in SaFIN. One can easily verify that the derived rulebase is consistent, where no two rules have the same antecedent segments but different consequent segments. Figure 3.7(b) shows the modeling results of the trained SaFIN model on the testing dataset. As seen, there is a perfect match between the computed outputs of the network and the desired outputs of the system, with SaFIN achieving a root mean squared error (RMSE) of 0.011 on the testing dataset.

On the other hand, the proposed learning mechanisms of SaFIN are influenced by two pre-selected variables α and β . To provide a clearer understanding of the influences of these two parameters on the modeling performances of SaFIN, different values of them are tested. The test results concerning the number of identified fuzzy labels, fuzzy rules and the RMSE on the testing dataset are listed in Table 3.2. From the tables, it is observed that the RMSE value remains consistent for certain range of the parameters. For a fixed value of $\alpha = 0.2$, the numbers of identified fuzzy labels in the two inputs-one output spaces increase as the contrasting threshold β increases. As a result, a higher value of β results in a decrease in the RMSE value, but at the expense of a larger rulebase. The same results hold for a constant value of $\beta = 0.6$ with a decreasing minimum membership threshold α . In conclusion, the larger the interval $[\alpha, \beta]$, the greater the numbers of identified fuzzy

Table 3.2: Sensitivity test for different values of α and β in SaFIN.

β	# Fuzzy Labels			# Rules	RMSE
0.50	Inputs =	3, 3;	Output =	3	0.017
0.55	Inputs =	4, 4;	Output =	4	0.011
0.60	Inputs =	4, 4;	Output =	4	0.011
0.65	Inputs =	6, 6;	Output =	6	0.007
0.70	Inputs =	6, 7;	Output =	6	0.007
$\alpha = 0.2$					
α	# Fuzzy Labels			# Rules	RMSE
0.10	Inputs =	6, 7;	Output =	6	0.006
0.15	Inputs =	6, 6;	Output =	6	0.007
0.20	Inputs =	4, 4;	Output =	4	0.011
0.25	Inputs =	4, 4;	Output =	4	0.012
0.30	Inputs =	3, 4;	Output =	3	0.023
$\beta = 0.6$					

labels and fuzzy rules, and thus the lower the RMSE values. In general, the variables are chosen as follows: $\alpha \in [0.2, 0.4]$ and $\beta \in [0.5, 0.7]$; and/or with an average interval size of $[0.2, 0.5]$.

3.3.2 Example 2 – Nakanishi dataset

The self-organizing abilities of the proposed SaFIN model are evaluated using three modeling experiments in the Nakanishi dataset [Nakanishi, Turksen & Sugeno 93][Sugeno & Yasukawa 93]; namely, modeling of (a) a nonlinear system; (b) the human operation of a chemical plant; and (c) the daily pricing of a stock in a stock market. Following the descriptions from the papers, each of the three datasets is split into three groups A, B, and C where A and B form the training dataset and C is the testing data. The benchmarks for comparisons are the accuracies on the testing data (calculated as the mean squared error MSE) and the correspondence between the computed outputs with the testing data (calculated as the Pearson correlation coefficient R). The experimental results of SaFIN are subsequently benchmarked against the following models; namely, Mamdani models – Hebb-R-R [Liu, Quek & Ng 07]; POPFNN [Quek & Zhou 96]; RSPOP [Ang & Quek 05]; EFuNN [Kasabov 01], Reasoning models [Nakanishi, Turksen & Sugeno 93] – Sugeno P&P-G, Sugeno P, Sugeno P-G, Mamdani, and Turksen IVCRI, and TSK models – ANFIS [Jang 93]; DENFIS [Kasabov & Song 02]; FITSK [Quah & Quek 06].

A nonlinear system

The objective of this experiment is to identify and model the underlying principles of a nonlinear system. In the original dataset, there were four input (x_1 – x_4) and one output (y) variables. Adopting the approach suggested in [Nakanishi, Turksen & Sugeno 93], only input variables x_1 and x_2 are subsequently used in the modeling. Respectively, a total of 3, 4 and 3 fuzzy clusters are identified by the proposed CLIP technique for the input-output

Figure 3.8: Semantic fuzzy clusters in the first input dimensions; and modeling results on testing datasets for the Nakanishi dataset using SaFIN.

dimensions. Figure 3.8A. illustrates the identified fuzzy clusters in the first input variable x_1 . As before, the fuzzy clusters identified are highly ordered, where clear semantic meanings can be attached to the fuzzy clusters. This result demonstrates the excellent self-organizing clustering abilities of the proposed SaFIN model. Also shown in Figure 3.8A. is the modeling results of the trained SaFIN model on the unseen testing dataset. The computed output of the SaFIN model follows the trend of the desired output, indicating the good generalization ability of SaFIN. A total of ten Mamdani-type fuzzy rules are identified by SaFIN in this experiment. They are listed in Table 3.3. As seen, the derived rulebase is consistent as no two rules have similar antecedent conditions and different consequences. In addition, the knowledge embedded in the set of fuzzy rulebase is highly systematic and sound. This can be seen by examining the first four rules. Keeping x_1 as High and varying x_2 from Very High to Very Low, the output y correspondingly varies from Low to Medium. This implies that y is inversely proportional to x_2 when x_1 is fixed. This relationship is repeatedly observed when x_1 is fixed at Medium or Low. Rearranging the rules, similar observations can be deduced between x_1 and y when x_2 is kept fixed. This result demonstrates the effectiveness of the self-automated rule generation mechanism proposed in SaFIN. Consolidated experimental results on the benchmarking measures are given in Table 3.4.

Human operation of a chemical plant

The proposed SaFIN model is employed to model the human operation of a chemical plant in this experiment. Although there were five input (x_1 – x_5) and one output (y) variables in the original dataset, only selected input variables (x_1 and x_3) are used in this experiment [Nakanishi, Turksen & Sugeno 93]. Three fuzzy clusters are identified in each of the input-output dimensions. Figure 3.8B. illustrates the semantic fuzzy clusters in x_1 identified by SaFIN, together with the modeling results of SaFIN on the testing dataset. There is an almost perfect match between the computed output of SaFIN and the desired output of the chemical plant. Subsequently, by attaching semantic labels to the fuzzy clusters in each

Table 3.3: Mamdani-type fuzzy rules identified in SaFIN for the Nakanishi dataset.

Rule	Nonlinear System			Chemical Plant			Stock Prediction			
	x_1	x_2	y	x_1	x_3	y	x_4	x_5	x_8	y
1	H	VH	L	H	L	L	PH	PH	PH	NH
2	H	H	L	M	L	L	PH	PH	PL	NH
3	H	L	L	M	M	M	PM	PH	PH	NH
4	H	VL	M	L	M	H	PM	PH	PL	NL
5	M	VH	L	L	H	H	PM	PL	Z	NM
6	M	L	M				PL	PH	PH	NL
7	M	VL	M				PL	PL	Z	PH
8	L	VH	M				NL	PH	PH	NL
9	L	L	M				NL	PH	PL	NL
10	L	VL	H				NL	PH	Z	NL

L–Low; M–Medium; H–High; Z–Zero; N–Negative; P–Positive; V–Very

input-output dimension, Mamdani-type fuzzy rules can be extracted from the SaFIN model. The five extracted fuzzy rules are listed in Table 3.3. Consistency of the fuzzy rulebase can be easily verified. Consolidated experimental results on the benchmarking measures are given in Table 3.4.

Daily pricing of a stock in a stock market

Using various statistics concerning a stock collected from a stock market, the proposed SaFIN model is employed to perform stock price prediction. Selected input variables (x_4 , x_5 and x_8) are used although there were initially ten input variables (x_1 – x_{10}) [Nakanishi, Turksen & Sugeno 93]. Correspondingly, the number of fuzzy clusters identified for the three input and one output variables are 6, 4, 5 and 6. Figure 3.8C. illustrates the fuzzy clusters in x_4 identified by SaFIN. Clear and distinct semantic labels are attached to the fuzzy clusters. A comparison between the computed output of the SaFIN model and the desired output of the testing dataset has also been included in Figure 3.8C., where the

proposed system is able to model the trend in the stock price movement satisfactorily. A total of twenty-one fuzzy rules are extracted from the SaFIN model; while Table 3.3 lists the first ten identified rules for this experiment. Consolidated experimental results on the benchmarking measures are given in Table 3.4.

Discussion

Table 3.4 shows the consolidated experimental results for the Nakanishi dataset between the proposed SaFIN model and the benchmarking models. SaFIN outperforms all the benchmarking models in the first two tasks by ranking first in terms of MSE and R. In the task of modeling a nonlinear system, SaFIN achieves a MSE value of 0.057, a remarkable 69.2% reduction compared to the second place Hebb-R-R. In addition, there is a slight improvement of 6.70% in the R value achieved from 0.911 to 0.972. For the second task of modeling a chemical plant, the proposed SaFIN model delivers an outstanding performance by achieving a reduction of 44.1% in terms of calculated MSE and a slight improvement of 0.001 in terms of correlation R compared to the second position Hebb-R-R model. Although SaFIN loses out to Hebb-R-R in the final task of stock prediction, it should be noted that Hebb-R-R uses eight (out of ten) input variables in this task [Liu, Quek & Ng 07]. Comparatively, SaFIN uses much lesser information by utilizing only three input variables, a significant reduction of 167%. This explains the slight compromise in the MSE and R values achieved by SaFIN. On the other hand, RSPOP uses five input variables [Ang & Quek 05]. Although it uses more information, the performances of SaFIN and RSPOP are comparable. On average, the proposed SaFIN model achieves a ranking of 1.5 for all the three tasks. This makes SaFIN the best performer in this set of experiment.

Table 3.4: Consolidated experimental results for the Nakanishi dataset using SaFIN.

Model	Nonlinear System		Chemical Plant		Stock Prediction		Average Rank
	MSE	R	MSE	R	MSE	R	
	(Rank)	(Rank)	(Rank)	(Rank)	(Rank)	(Rank)	
Hebb-R-R	0.185 (2)	0.911 (2)	2.423×10^4 (2)	0.998 (2)	15.14 (1)	0.947 (1)	1.7
POPFNN	0.270 (3)	0.877 (3)	5.630×10^5 (8)	0.946 (9)	76.22 (10)	0.733 (10)	7.2
RSPOP	0.383 (7)	0.856 (4)	2.124×10^5 (5)	0.983 (7)	24.86 (3)	0.922 (2)	4.7
Sugeno P&P-G	0.345 (6)	0.828 (7)	2.897×10^5 (7)	0.973 (8)	94.58 (12)	0.706 (11)	8.5
Sugeno P	0.776 (12)	0.558 (12)	6.372×10^5 (9)	0.933 (12)	35.47 (5)	0.883 (4)	9
Sugeno P-G	0.467 (9)	0.845 (6)	1.931×10^6 (12)	0.990 (6)	168.9 (13)	0.700 (12)	9.7
Mamdani	0.862 (13)	0.490 (13)	6.580×10^5 (10)	0.937 (11)	40.84 (7)	0.865 (7)	10.2
Turksen IVCRI	0.706 (11)	0.609 (11)	2.581×10^5 (6)	0.993 (4)	93.02 (11)	0.661 (13)	9.3
ANFIS	0.286 (4)	0.853 (5)	2.968×10^6 (13)	0.780 (13)	38.06 (6)	0.875 (6)	7.8
EFuNN	0.566 (10)	0.720 (10)	7.247×10^5 (11)	0.946 (9)	72.54 (9)	0.756 (9)	9.7
DENFIS	0.411 (8)	0.805 (9)	5.240×10^4 (4)	0.995 (3)	69.82 (8)	0.810 (8)	6.7
FITSK	0.336 (5)	0.828 (7)	3.862×10^4 (3)	0.993 (4)	33.78 (4)	0.883 (4)	4.5
SaFIN	0.057 (1)	0.972 (1)	1.354×10^4 (1)	0.999 (1)	23.49 (2)	0.918 (3)	1.5

3.3.3 Example 3 – UCI dataset

The classification and regression abilities of the proposed SaFIN model are evaluated using three real-life datasets from the UCI dataset; namely; (a) the servo data; (b) the wine data; and (c) the iris data. Both the wine and the iris data are classification problems; while the servo data is a regression problem. A detailed description of the datasets is found in [UCI]. In the tasks of servo regression and wine classification, the first 60% of the data are used as training set, while the remaining 40% are used for testing; while a three-fold cross-validation, with 34% of the data as training and 66% of the data as testing sets, is performed in the iris classification. The benchmarking measures are the accuracies on the testing data (calculated as the root mean squared error RMSE for the regression problem, and classification accuracies CA% for the classification tasks) and the complexity of the networks (indicated by the size of the rulebase and the training time T_s). The proposed SaFIN model is subsequently benchmarked against the following models; namely, k -NN [Cover & Hart 67]; MLP; HyFIS [Kim & Kasabov 99]; DENFIS [Kasabov & Song 02]; EFuNN [Kasabov 01]; FMM [Simpson 92]; and GFMM [Gabrys & Bargiela 00]. All the models are running on the same computer platform; while results for FMM and GFMM are extracted from [Gabrys & Bargiela 00].

Table 3.5 shows the consolidated experimental results for the UCI dataset. The testing results are listed for the servo and wine tasks; while the average performances are given for the iris classification. The performances of the proposed SaFIN model and the benchmarking models are subsequently ranked according to their testing accuracies and training time¹. As clearly seen, the SaFIN model is consistently one of the top performers when ranked against the testing accuracies for all three tasks. This illustrates the excellent modeling and generalization abilities of SaFIN. In addition, the SaFIN model uses much lesser rules com-

¹The models are not ranked for the size of their rulebase because the information is either not applicable or unavailable for most models. In addition, both FMM and GFMM do not participate in the servo regression task (and subsequently, the ranking) because they are classification models.

Table 3.5: Consolidated experimental results for the UCI dataset using SaFIN.

Model	Servo			Wine			Iris			Average Rank
	# Rules	RMSE (Rank)	T (Rank)	# Rules	CA (Rank)	T (Rank)	# Rules	CA (Rank)	T (Rank)	
<i>k</i> -NN	N.A.	1.022 (6)	0.03 (1)	N.A.	81.69 (6)	0.03 (1)	N.A.	94.65 ± 1.60 (4)	0.02 (1)	3.2
MLP	N.A.	0.712 (2)	0.78 (2)	N.A.	82.54 (5)	4.52 (4)	N.A.	93.98 ± 3.53 (5)	0.54 (5)	3.8
HyFIS	84	0.742 (4)	2.26 (5)	96	94.37 (4)	12.09 (6)	13.3	95.36 ± 3.17 (3)	0.47 (3)	4.2
DENFIS	8	0.734 (3)	5.90 (6)	15	98.59 (3)	5.70 (5)	12	97.01 ± 0.95 (1)	0.70 (6)	4
EFuNN	75	0.804 (5)	0.90 (3)	100	100 (1)	1.00 (2)	28	92.63 ± 2.98 (6)	0.27 (2)	3.2
SaFIN	12	0.575 (1)	1.09 (4)	93	100 (1)	3.98 (3)	13.7	96.34 ± 0.55 (2)	0.50 (4)	2.5
FMM*				-	100 (-)	-	48	97.33	-	-
GFMM*				-	100 (-)	-	29	97.33	-	-

* For the iris classification, 50% of the data is used for training and testing respectively.

Figure 3.9: Distribution of iris dataset.

pared to most of the benchmarking models in the tasks, second only to DENFIS. Despite that, it should also be noted that the training time of DENFIS is among the slowest for all the three experiments. On the other hand, SaFIN has a moderate training time when ranked against the benchmarking models, claiming the third or fourth positions out of six benchmarking models. Despite being among the fastest training models, both k -NN and EFuNN do not deliver comparable testing performances. On average, the proposed SaFIN model achieves a ranking of 2.5, making it the best performer in this set of experiment. On a second note, the testing performances of SaFIN are comparable with the classification models, FMM and GFMM, in both the wine and iris classification tasks, while it utilizes significantly lesser rules. SaFIN is also consistent in its modeling capabilities by achieving the smallest variance in the three-fold cross-validation for the iris classification task. This result demonstrates that the proposed SaFIN model is able to deliver superior modeling performances, while maintaining a good balance in the complexity of the network.

As a further investigation to test the classification potential of the proposed SaFIN model, a recalling test is performed such that the whole dataset of the iris classification task is used to train SaFIN, and the trained model is subsequently tested on the dataset. Figure 3.9 illustrates the data distribution where there are three classes in the iris classification; namely, Setosa, Virginica and Versicolor. Each instance of the data consists of four physical attributes; namely, sepal length, sepal width, petal length, and petal width. As clearly seen,

Figure 3.10: Fuzzy clusters identified by SaFIN (with corresponding numerical data) for the iris classification task.

the Setosa class is linearly separable from the other classes; while the Virginica and Versicolor classes are nonlinearly separable. The fuzzy clusters identified by the proposed SaFIN model, with the corresponding semantic fuzzy labels, are shown in Figure 3.10. Also included in the figure is the numerical distribution of the physical attributes for the three classes. As seen, there are significant amount of over-lap between the distributions of the three classes for the sepal measurements, with SaFIN identifying two and four fuzzy labels for the length and width respectively. On the other hand, the petal measurements show three classes of data, corresponding to the three classes of flowers. Subsequently, SaFIN is able to identify three fuzzy clusters in the petal measurements, each corresponding to the three classes of flowers. This illustrates the excellent self-organizing clustering abilities of the proposed SaFIN model. Finally, SaFIN achieves a remarkable classification accuracy of 97.33% for this recalling test. This indicates that the proposed model possesses great classification potential for nonlinearly separable classification tasks.

3.4 Summary

This chapter presents a novel self-organizing neuro-fuzzy system framework named *Self-adaptive Fuzzy Inference Network* (SaFIN). A key strength of SaFIN is the proposed knowledge acquisition methodology that draws inspiration from the behavioral category learning process exhibited by humans. This translates to a new single-pass fuzzy partitioning technique known as *Categorical Learning Induced Partitioning* (CLIP) that is able to rapidly partition the input-output spaces. Knowledge is extracted from each arriving training tuple to perform fuzzy partitioning of the input-output spaces. This approach not only allows a tailored partitioning of the input-output dimensions; new clusters in each input-output dimension can also be automatically identified by SaFIN when a prominent distinction is observed. This frees the human users from pre-defining the number of clusters necessary in the input-output dimensions. In addition, refinements are made to existing clusters in an input-output space when a new cluster is added. This effectively addresses the stability-plasticity trade-off of the model by maintaining a balance between the co-existence of past knowledge (in the system) and future knowledge (from the newly added cluster) in the computational structure. Finally, SaFIN handles the problem of a conflicting rulebase by retaining only the most significant rule in the set of inconsistent rules. This approach ensures that the proposed SaFIN model maintains a consistent rulebase that is able to give an aptly description of the application environment. SaFIN is subsequently employed in three benchmarking experiments to demonstrate its superiority as a self-organizing neuro-fuzzy system; and excellent performances have been achieved.

Chapter 4

Transportation Case Study: Highway Traffic Modeling Using SaFIN

This chapter investigates the potential of using the SaFIN model, proposed in Chapter 3, for the modeling of a real-world application involving the prediction of highway traffic flow density. The effectiveness of the model is evaluated in terms of the mean squared error (MSE) and the Pearson correlation coefficient (R) between the computed output of SaFIN and the actual traffic flow density of the highway. These measures are significant because they indicate the modeling accuracy and the generalization ability of the proposed SaFIN model for the application problem, i.e., determining its suitability to detect changes in the demands on the highway usage throughout the experimental duration, which is important in traffic planning to provide a good level of service to commuters.

This chapter is organized as follows. Section 4.1 discusses on the increasing needs to perform good traffic flow prediction. Section 4.2 summarizes fundamental traffic flow characteristics,

and some associated modeling techniques in the literature. The deficiencies of existing modeling techniques are also highlighted. This is followed by a description of the highway traffic flow data that is considered in this case study in Section 4.3. Section 4.4 presents the experimental results achieved. Lastly, Section 4.5 provides some concluding remarks to this transportation case study.

4.1 Motivation

A pressing force for changes and adjustments in a transportation system is an increase in demands on the system caused by an increase in the number of motor vehicles. This is a direct consequence of growing affluence, where vehicle ownership tends to rise with increasing income [Simmonds & Hack 00]. Subsequently, the performance of a transportation system is evaluated by its ability to move the increase influx of people, goods and equipments from one place to another place in an effective and reliable manner.

Traffic congestion on a highway occurs when the traffic demand exceeds the operational capacity of the highway. It is one of the most serious challenges threatening the effectiveness of a transportation system, since traffic congestion can lead to other associated environmental and economical issues, such as reduced capacity, reduced safety, increased gas emissions, increased fuel consumption, and increased travel times leading to productivity losses [Darbha, Rajagopal & Tyagi 08]. In general, there are 6 levels of services to evaluate the traffic flow conditions along a highway section [Transportation Research Board 00], and they are defined as follows [Garber & Hoel 02]:

Level of Service A - Free flow operations in which vehicles are completely unimpeded in their ability to maneuver, where the effects of incidents or point breakdowns are easily absorbed.

Level of Service B - Traffic is moving under reasonable free flow conditions and the ability to

maneuver within the traffic stream is only slightly restricted, where the effects of incidents or point breakdowns are easily absorbed.

Level of Service C - Speeds are at or near free flow speed, but the freedom to maneuver is noticeably restricted such that more care is required when performing lane changes. When minor incidents occur, local deterioration in service will be substantial and queues may be expected to form behind any significant blockage.

Level of Service D - Speeds begin to decline slightly such that freedom to maneuver is more noticeably limited. Because there is so little space to absorb disruptions, queuing can be expected in the events of minor incidents.

Level of Service E - Operations are volatile because there are virtually no useable gaps. Maneuvers such as lane changes or merging of traffic from entrance ramps will result in a disturbance of the main traffic stream. Minor incidents will also result in immediate and extensive queuing.

Level of Service F - Operation is under breakdown condition. The number of vehicles that can pass through a point is less than the number of vehicles arriving upstream of the point. Breakdown occurs when the ratio of forecasted demand to capacity exceeds 1.0.

As can be seen, the prediction and forecasting of highway traffic flow density is important because it helps to determine the demands on the highway usage, which is a useful indicator to the level of service that might be experienced by commuters. A high level of service is expected when the forecasted traffic flow is small; while the expected level of service drops when the forecasted traffic flow is huge. In such cases, pre-emptive measures can then be adopted to improve the level of service and prevent the onset of congestion (for example, raising the tolls on a road during peak hours to limit the number of vehicles, or increasing the gasoline tax to reduce the number of vehicles on the roads). In addition, growth in the

volume of traffic is anticipated to be an on-going phenomenon [Darbha, Rajagopal & Tyagi 08], thus emphasizing the urgent needs to have an efficient system for a good and reliable prediction of traffic flow density on the roads.

4.2 Traffic flow analysis

This section reviews some basic characteristics of traffic flow analysis, where a brief review on intelligent transportation systems is presented. Basic elements used in the description of traffic flow will be defined in Section 4.2.1. This is followed by an introduction to two classical models to mathematically formalize the traffic flow dynamics in Section 4.2.2. Section 4.2.3 presents alternative approaches to model traffic flow, focusing on the study and development of intelligent transportation systems.

4.2.1 Basic elements in traffic flow analysis

There are four basic elements when describing the traffic flow conditions [Garber & Hoel 02]; namely, flow, density, speed, and the headway between two vehicles in a traffic stream. They shall be defined here.

Definition 8 Flow (Q) is the equivalent hourly rate at which vehicles pass through a point on a highway during a period less than an hour. It is defined as in (4.1):

$$Q = \frac{3600 n}{T} \quad (4.1)$$

where n is the number of vehicles passing through a point on the highway in T seconds. The unit for flow is thus given as veh/hr.

Definition 9 Density (K) is the number of vehicles travelling over a unit length of a highway at an instant in time. Assuming the unit length of a highway to be one kilometre, the unit for density is then given as veh/km.

Definition 10 Speed (U) is the distance travelled by a vehicle during a unit in time, and the unit for speed is given as km/hr. There are two types of mean speeds to describe the general traffic stream on a highway, and they are defined as follows:

- Time mean speed ($\bar{U}_t$) is the arithmetic mean of the speeds of vehicles passing through a point on a highway during an interval of time. It is defined as in (4.2):

$$\bar{U}_t = \frac{1}{n} \sum_{i=1}^n U_i \quad (4.2)$$

where n is the number of vehicles passing through a point on the highway, and U_i is the speed of the i -th vehicle.

- Space mean speed ($\bar{U}_s$) is the harmonic mean of the speeds of vehicles passing through a point on a highway during an interval of time. It is defined as in (4.3):

$$\bar{U}_s = \frac{nD}{\sum_{i=1}^n T_i} \quad (4.3)$$

where n is the number of vehicles passing through a point on the highway, D is the length of a section on the highway, and T_i is the time taken for the i -th vehicle to travel this section of the highway.

Definition 11 There are two dimensions when describing the headway between two vehicles, and they are defined as follows:

- Time headway is the time difference between the moment the front of a first vehicle

arrives at a point on the highway and the moment the front of the next vehicle arrives at that same point. The unit is thus given as hr.

- Space headway is the distance between the front of a first vehicle and the front of the next vehicle, and the unit is given as km.

In general, there are two approaches when describing the traffic flow conditions; namely, the *macroscopic* approach and the *microscopic* approach. The first three elements (i.e. flow, density, and space mean speed of the traffic stream) defined above are associated with the macroscopic approach, while the last two elements (i.e. speed of a vehicle, and the headway between two vehicles) are related to the microscopic approach. Intuitively, the macroscopic elements provide an overview of the traffic flow condition on a highway by inspecting the main traffic stream; while the microscopic elements consider distinct members in the traffic by examining vehicle spacings and speeds of individual vehicles. Since both the spacings and the speeds of individual vehicles are directly related to the overall speed of the main traffic stream [Quek, Pasquier & Lim 09], the macroscopic traffic flow elements shall form the main focus in this work. A general equation describing the relationship between the three macroscopic elements is given in (4.4):

$$Q = K \bar{U}_s . \quad (4.4)$$

4.2.2 Classical mathematical models

As explained previously, the macroscopic elements offer an overview of the general highway traffic flow stream. This section shall review two classical mathematical models that relate the three basic elements in the macroscopic approach to traffic flow analysis. They are the *Greenshields model* and the *Greenberg model*.

Greenshields model - Greenshields presented one of the earliest attempts at understanding the mechanics of traffic flow by hypothesizing a linear relationship between the space mean speed and the density of the traffic flow [Greenshields et. al. 35]. It is expressed as in (4.5):

$$\bar{U}_s = U_f - \frac{U_f}{K_j} K \quad (4.5)$$

where U_f is known as the *mean free speed*, which is the maximum absolute speed achieved as the flow tends to zero; while K_j is known as the *jam density*, which is the maximum density permissible. Substituting (4.4) into (4.5), an alternative form of the model describing the relationship between the flow and the density of the traffic is given in (4.6):

$$Q = \left(U_f - \frac{U_f}{K_j} K \right) K . \quad (4.6)$$

Greenberg model - Greenberg adopted the analogy of fluid flow to develop macroscopic relationships in traffic flow [Greenberg 59], where the relationship between the space mean speed and the density of the traffic is given as in (4.7):

$$\bar{U}_s = U_o \ln \frac{K_j}{K} \quad (4.7)$$

such that U_o is the speed at maximum flow. As before, an equivalent model describing the relationship between the flow and the density of the traffic can be obtained by substituting (4.4) into (4.7), and it is given as in (4.8):

$$Q = U_o K \ln \frac{K_j}{K} . \quad (4.8)$$

As seen from (4.5)–(4.8), one is able to compute both the flow and the space mean speed of the traffic using information from the density. This, subsequently, makes the density of a

Figure 4.1: Fundamental principles of traffic flow analysis.

traffic stream the most important descriptive element to evaluate the service quality of the traffic.

4.2.3 Intelligent transportation systems

The mathematical models developed in Section 4.2.2 are applicable only if they satisfy some boundary criteria defined in the *fundamental principles of traffic flow analysis* [Garber & Hoel 02]. Figure 4.1 illustrates three graphs describing the relationships between the macroscopic elements, that define the fundamental principles for traffic flow analysis, where two boundary conditions have been marked as B_1 and B_2 respectively. At B_1 , the space mean speed of the traffic flow approaches mean free speed U_f when the density approaches zero; and at B_2 , the space mean speed approaches zero when the density of the traffic flow approaches the jam density K_j . Since the Greenshields model satisfies both B_1 and B_2 , it is applicable under both light and dense traffic flow conditions; on the other hand, the Greenberg model satisfies only the boundary criterion at B_2 , indicating that it is only applicable under dense traffic conditions. In general, the effectiveness of mathematical

models for traffic flow analysis is limited to regions where boundary conditions are fulfilled. Subsequently, such restrictions have motivated alternative approaches to model traffic flow conditions such that the entire duration of interest can be accounted for effectively.

An intelligent transportation system aims to enhance the traffic infrastructure and the operational capacity of the roads via the acquisition and processing of information through the use of recent advances in electronics, telecommunications, computing and information processing technologies [Darbha, Rajagopal & Tyagi 08]. With respect to this vision, alternative methods such as probabilistic approaches [Hoogendoorn & Bovy 01][Schadschneider 02][Mahnke, Kaupužs & Lubashevsky 05], neural networks [Dougherty & Kirby 94][Ledoux 97][Dia 01], and fuzzy models [Ngo & Li 94][Ghatee & Hashemi 09] have been employed to model highway traffic flow. Despite some promising results, most statistical approaches developed are based on some theoretical assumptions and principles, which can limit their practicality on real-world applications [Stathopoulos & Karlaftis 01]. On the other hand, hybrid methods such as neuro-fuzzy systems have demonstrated greater flexibility when dealing with large-scale real-world problems [Dimitriou, Tsekeris & Stathopoulos 08]. Subsequently, recent works have demonstrated much interest in using neuro-fuzzy systems for the modeling of various aspects of a transportation system. For example, Hu et. al. proposes an autonomous traffic surveillance system such that immediate responses can be activated in the events of incidents [Hu et. al. 04]. In [Cho et. al. 09], a highway vehicles tracking system using images is presented to process both day and night traffic analysis. On the other hand, a novel biometric profiling of individual driver has been reported in [Wahab et. al. 09], such that existing vehicle security can be enhanced. Chen and Tai proposed a road detection experimentation based on a new colour space [Chen & Tai 10], which can potentially help drivers to avoid hazards on the roads.

In this case study, the potential of using the SaFIN model for the detection of changes in

Figure 4.2: Location of site 29 along the Pan Island Expressway in Singapore.

the demands on the highway usage throughout the entire duration of interest is investigated. This is particularly useful in determining the possible level of service that might be experienced by commuters, such that suitable adjustments can be adapted to an existing transportation system to enhance the service quality of the highway.

4.3 Data collection and experimental setup

The set of highway traffic flow data employed in this case study is described hereafter. Data was collected from a five lanes section along the Pan Island Expressway (PIE), Singapore, in both the east-bound and west-bound directions towards Changi and Jurong respectively. In this study, only the east-bound direction is considered, where the data collection is performed at site 29 located at exit 15 on the expressway. Data samples were collected from inductive loop detectors installed beneath the road surface. Since 1996, the Land Transport Authority has pre-installed such detectors along major roads in Singapore to facilitate the collection of traffic flow data.

Figure 4.3: Traffic flow densities of the three straight lanes along PIE at site 29.

Figure 4.2 shows a picture of the location where the data was collected. There are a total of five lanes; namely, three straight lanes for the main traffic (lanes 1–3), and two exit lanes (lanes 4–5). Only data from the three straight lanes are used in this case study. They are correspondingly denoted as L1, L2 and L3. Each data sample has four attributes; namely, the time t at which the traffic flow data was measured, and the traffic flow densities for the three straight lanes during time t . Subsequently, the SaFIN model is employed to model the traffic flow trend. After which, the trained model is used to predict traffic flow density of a lane (L1, L2 or L3) at a time $t + \tau$ for $\tau = 5, 15, 30, 45$ and 60 minutes.

Figure 4.3 shows the traffic flow data for lanes L1–L3 spanning over a period of six days from 5-th to 10-th September 1996. The data is divided into three cross-validation groups, hereby denoted as CV1, CV2 and CV3 respectively. The training data for each cross-validation group is extracted accordingly from the period labeled in Figure 4.3.

Figure 4.4: Fuzzy clusters identified by SaFIN for the time domain, and for lanes L1–L3 (with distributions of raw data) for the training set of CV1 when $\tau = 5$ minutes.

4.4 Experimental results and analysis

This section presents the modeling performances of the SaFIN model for the highway traffic flow data described in Section 4.3. The effectiveness of SaFIN is evaluated in terms of the mean squared error MSE and the Pearson correlation coefficient R achieved between the computed output of the model and the actual traffic flow density of the highway. These measures help to determine the suitability of employing SaFIN for the detection of changes in the demands on the highway usage throughout the entire experimental duration (i.e. the duration of interest). The performance of the SaFIN model is subsequently benchmarked against the following models; namely, Hebb-R-R [Liu, Quek & Ng 07]; RSPOP [Ang & Quek 05]; MLP (with a configuration of 4 input nodes, 10 hidden nodes and 1 output node); GenSoFNN [Tung & Quek 02], EFuNN [Kasabov 01]; DENFIS [Kasabov & Song 02]; and eFSM [Tung & Quek 10].

Figure 4.4 illustrates the identified fuzzy clusters in the time domain and those for lanes L1–L3 for the training set of CV1 when $\tau = 5$ minutes. The distributions of the raw numerical data are also shown for each of the three lanes in the figure. The identified fuzzy clusters are well-ordered where clear distinct semantic fuzzy meanings can subsequently be attached. As seen, the fuzzy clusters identified in SaFIN for lanes L1–L3 coincide with the peaks of the distributions as marked by the dotted circles. Although there are very low distributions of data closer to the upper bounds of the three lanes, it is observed that the distributions immediately before the peaks are closest/at the zero mark. Hence, SaFIN identifies a cluster near the upper bound of each lane to cater for this peak. This figure demonstrates the tailored approach adopted in SaFIN for the fuzzy partitioning, where the numbers and positions of fuzzy clusters identified correspond to the distributions of the original numerical data. Subsequently, this allows traffic officers and policy makers to have an intuitive grasp of the knowledge embedded in the raw numerical data.

Using the semantic fuzzy labels in Figure 4.4, twenty-three fuzzy rules are identified by SaFIN as listed in Table 4.1. From the table, four rules correspond to the morning hours, fourteen rules correspond to the afternoon period, and five rules describe the night time traffic. During the early morning and late night hours, the amount of traffic flowing through lanes L1–L3 range from Low to Medium. This means that, there is only light to moderate amount of traffic on the highway during these periods. Comparing this observation to the training set of CV1 in Figure 4.3, this is indeed the case. Subsequently, the predicted traffic demand on the highway during these periods is moderately low. This corresponds to the usual work-life cycle of most people, where majority of the highway users are resting during the early morning and late night periods. On the other hand, most of the identified rules describe traffic flow for the afternoon period, i.e., during the day when most people travel for work and errands. Within this period, the traffic flow in lanes L1–L3 can range from Low to High. This means that variations in the highway usage can be observed throughout

Table 4.1: Mamdani-type fuzzy rules identified in SaFIN for CV1 when $\tau = 5$ minutes.

Rule	Time t	L1(t)	L2(t)	L3(t)	L1($t + 5$)
1	Morning	L	L	L	L
2	Morning	L	L	M	L
3	Morning	L	M	M	L
4	Morning	M	M	M	M
5	Afternoon	L	L	L	M
6	Afternoon	L	M	L	M
7	Afternoon	L	M	M	M
8	Afternoon	L	M	H	M
9	Afternoon	M	L	L	M
10	Afternoon	M	L	M	M
11	Afternoon	M	M	L	M
12	Afternoon	M	M	M	M
*13	Afternoon	M	M	H	M
*14	Afternoon	M	H	M	M
*15	Afternoon	M	H	H	M
*16	Afternoon	H	M	M	M
*17	Afternoon	H	H	H	M
*18	Afternoon	H	M	H	H
19	Night	L	L	L	L
20	Night	L	M	L	L
21	Night	L	M	M	L
22	Night	M	M	L	L
23	Night	M	M	M	L

L–Low; M–Medium; H–High

Figure 4.5: Recall and prediction of traffic density on lane L1 when $\tau = 5$ minutes.

the day period, where half of the fourteen rules describe the peaks hours of the day when people rush to/off work (i.e. rules *13–*18), with the remaining describing the day hours when people are already at work. Subsequently, the predicted traffic demand on the highway during the day period is moderately high (as compared to that during the early morning and late night hours). Hence, the set of identified fuzzy rulebase in SaFIN provides a logical, sound, and intuitive description to the highway traffic flow, where the demands on the highway usage for different hours of the day can be effectively accounted for.

The recall and prediction performances of SaFIN for lane L1 when $\tau = 5$ minutes are shown

in Figure 4.5. The results are shown for the three cross-validation classes CV1–CV3. As seen, the recall performances on the training sets for CV1–CV3 (i.e. Figures 4.5(a),(c),(e)) are excellent, with more than 0.9 achieved for the R values. To illustrate that SaFIN has learnt from the training sets to generalize the underlying knowledge to the entire duration of interest, the trained model is subsequently employed on the testing datasets. The prediction performances of SaFIN on the unseen data for CV1–CV3 are shown in Figures 4.5(b),(d),(f) respectively. Compared to the recalling performances, the prediction performances of SaFIN have deteriorated slightly with larger reported MSE values, and smaller reported R values. Despite that, SaFIN is able to perform excellent prediction on the testing datasets, where the peaks and the valleys of the traffic flow densities (i.e. the increase and decrease in demands on the highway usage) can be clearly identified. This result demonstrates the superior modeling and generalization abilities of SaFIN such that it is able to provide a good prediction of the highway traffic flow density, where the changes in demands on the highway usage are effectively accounted for.

The prediction performances of SaFIN are benchmarked against MLP and existing neuro-fuzzy systems, and the consolidated highway traffic flow prediction results are shown in Figure 4.6. The average R value and the average MSE value from the three cross-validation groups CV1–CV3 for each prediction horizon are plotted with respect to the lanes L1–L3. As seen, SaFIN is among the top performers in this highway traffic flow prediction application such that it is able to consistently achieve one of the highest average R and the lowest average MSE under different time horizons. This is particularly prominent when $\tau = 60$ minutes where SaFIN is ranked either the first or second positions for all three lanes L1–L3; while most of the benchmarking models have greater errors due to a longer time lag in the prediction horizon. Through this benchmarking with other existing models, SaFIN has demonstrated a consistently excellent performance such that it is able to perform good prediction within the various time periods of interest.

Figure 4.6: Traffic flow prediction results.

Table 4.2: Average performances for the traffic flow prediction.

Model	Average R ($\pm$ Std. Dev.)	Average MSE ($\pm$ Std. Dev.)
Hebb-R-R	0.864 ($\pm$ 0.046)	0.114 ($\pm$ 0.042)
RSPOP	0.834 ($\pm$ 0.041)	0.146 ($\pm$ 0.038)
MLP (4-10-1)	0.847 ($\pm$ 0.065)	0.130 ($\pm$ 0.055)
GenSoFNN	0.813 ($\pm$ 0.028)	0.164 ($\pm$ 0.037)
EFuNN	0.798 ($\pm$ 0.050)	0.189 ($\pm$ 0.041)
DENFIS	0.831 ($\pm$ 0.051)	0.153 ($\pm$ 0.054)
eFSM	0.840 ($\pm$ 0.043)	0.154 ($\pm$ 0.040)
SaFIN	0.862 ($\pm$ 0.043)	0.118 ($\pm$ 0.037)

Table 4.2 shows the average performances of all the models for this highway traffic flow density modeling task. As clearly shown, SaFIN demonstrates superior modeling potential, second only to Hebb-R-R, in terms of the average benchmarking measures achieved. Despite employing a time-consuming and computationally intensive iterative post-training phase to recursively identify a reduced set of fuzzy rules with the aim of a good accuracy, Hebb-R-R performs only marginally better than the proposed SaFIN model. Comparatively, the performance of SaFIN is much more consistent and stable as shown by the small standard deviations about the average benchmarking indexes. Although GenSoFNN achieves a lower standard deviation in the R value, it should be noted that the average performance of GenSoFNN is among the poorest under both the benchmarking measures. This result demonstrates the excellent modeling potential of the proposed SaFIN model; while maintaining a highly consistent and stable performance under varying time horizons (i.e. different time periods of interest).

4.5 Summary

This chapter demonstrates the suitability and effectiveness of employing the SaFIN model proposed in Chapter 3 for the modeling of highway traffic flow density. As the amount of traffic flow on the highway is directly related to the service of quality experienced by commuters, it is important to have an efficient system to perform a good and reliable traffic flow prediction. Subsequently, adjustments can be made to existing traffic planning policies when the forecasted demand on the highway exceeds the operational capacity of the highway (i.e. when traffic congestion occurs) such that a comfortable level of service can be provided.

Experimental results based on real-world traffic flow data collected from the Pan Island Expressway in Singapore are presented. Results showed that SaFIN is able to provide intuitive insights to the raw numerical data by providing semantic fuzzy labels and fuzzy rules to explain the traffic flow density on the highway throughout the day, i.e. , the changes in the demands on the highway usage throughout the duration of interest can be effectively accounted for in SaFIN. The modeling and generalization abilities of SaFIN are then compared against MLP and some existing neuro-fuzzy systems. Results showed that SaFIN is able to achieve superior performances with varying prediction horizons, while most of the benchmarking models are not able to handle the larger time lag in the prediction. In addition, SaFIN has also achieved a more consistent and stable performance for the highway traffic flow prediction when compared to the other models.

The potential of highway traffic flow prediction using the SaFIN model discussed in this chapter is encouraging as it demonstrates the feasible application of SaFIN in the area of traffic planning.

Chapter 5

Evolving Type-2 Neural Fuzzy Inference System

To design a neuro-fuzzy system, one crucial component is to derive the knowledge base of the system. This can be further investigated under two issues; namely, the fuzzy partitioning of the input and output spaces, and the rule generation scheme adopted. In Chapter 3, batch learning from numerical data is utilized to design a neuro-fuzzy system. Despite demonstrating good generalization to an application problem from a finite set of data, batch neuro-fuzzy systems might not perform as well when the dynamics of an application problem is changing over time. This is because a trained batch system from a previous time will not be able to perform good prediction on unseen data after a long period of time lag, where the dynamics of the unseen data has changed from that period of the trained system. On the other hand, time-varying application problems are frequently seen in real-life examples, and they can range from financial trading instruments [Tung & Quek 05b] [Tung & Quek 11], to assistive biomedical instruments [Vidaurre et. al. 06] [Shenoy et. al. 06], to physical phenomena [Horenko 10]. Hence, it is pragmatic to explore alternative neuro-fuzzy systems

with incremental evolutionary structures and parameters to model such non-stationaries.

A new class of evolving neuro-fuzzy systems with online learning abilities has thus been developed as seen from Table 5.1. Evolving neuro-fuzzy systems adopt a data-centric incremental learning mode where data is sequentially arriving such that each incoming data tuple goes through a cycle of structure and parameter learning. The data tuple is not stored in the system after learning and the neuro-fuzzy system performs learning on a single tuple at each time instance. This essentially allows an evolving neuro-fuzzy system to incorporate new knowledge that emerges after the system has identified its computational structure; hence, provides a life-long learning cycle for the detection of temporal shifts in the data patterns. Through this incremental processing of data, evolving neuro-fuzzy systems address the stability-plasticity trade-off [Grossberg 82] of a learning system. This means, while a progressive learning of new knowledge modifies an initial trained system (i.e. a process known as *plasticity*, that allows the continuous learning of new knowledge through changes to the structure and parameters of the initial trained system), it is able to avoid catastrophically erasing existing knowledge that is relevant (i.e. a process known as *stability*, that allows the system to recall previously learned knowledge encoded in the structure and parameters of the initial trained system). As stated by Abraham and Robins [Abraham & Robins 05], a dynamic learning system can retain memory and minimize memory losses through “a regulated balance between stability and plasticity to solve the trade-off between the stability required to retain information and the plasticity required for new learning”. Nevertheless, existing evolving neuro-fuzzy systems suffer from one or more of the following major drawbacks: (1) susceptibility to noisy data; (2) an ever-expanding knowledge structure; and (3) a lack of interpretability of the system.

Firstly, most existing evolving neuro-fuzzy systems adopt the structure of a Type-1 fuzzy model as seen from Table 5.1. As a result, they might not perform as desired under a noisy

Table 5.1: A summary of evolving neuro-fuzzy systems in the literature (reproduced from Table 2.1).

	TSK systems	(Publication Year)	Mamdani systems	(Publication Year)
Type-1	SONFIN	('98) [Juang & Lin 98]	FALCON-ART	('97) [Lin & Lin 97]
	NeuroFAST	('01) [Tzafestas & Zikidis 01]	EFuNN	('01) [Kasabov 01]
	GD-FNN	('01) [Wu, Er & Gao 01]	eFSM	('10) [Tung & Quek 10]
	DENFIS	('02) [Kasabov & Song 02]		
	eTS	('04) [Angelov & Filev 04]		
	SAFIS	('06) [Rong et. al. 06]		
Type-2	SEIT2FNN	('08) [Juang & Tsao 08a]	T2SONFS	('08) [Juang & Tsao 08b]
	ORGQACO	('09) [Juang & Hsu 09]		
	IT2FNN-SVM	('10) [Juang, Huang & Cheng 10]		

learning environment [Hagras 04][Sepúlveda et. al. 07]. On the other hand, it is known that online learning generally results in a learning system that is vulnerable to noise [Baraldi & Blonda 99a]. Hence, an evolving neuro-fuzzy system with greater noise tolerance is desired. Recent efforts have been directed to extend evolving Type-1 neuro-fuzzy systems to Type-2 models, since Type-2 models possess the potential to address the modeling bottlenecks of its Type-1 counterparts [Mendel & John 02]. This trend can be seen from Table 5.1. From the table, most works on evolving Type-2 neuro-fuzzy systems are TSK models. In contrast, there is minimum parallel effort for evolving Mamdani models. Although T2SONFS [Juang & Tsao 08b] has been listed as an evolving Type-2 Mamdani neuro-fuzzy system in the table, it should be noted that it is not entirely an online system since the input values have to be normalized prior to the design of the system. In [Ang & Quek 05], the differences between the two models have been explained as such: Mamdani models, also known as linguistic fuzzy models, focus on the interpretability of the systems; while TSK models, also known as precise fuzzy models, focus on the accuracy of the systems. Consequently, there is currently a need to investigate the design of an evolving Type-2 Mamdani neuro-fuzzy system.

Secondly, existing models of evolving Type-2 neuro-fuzzy systems perform continuous learning on incoming data tuples, where new rules and new clusters are added to the existing structures of the systems. On the other hand, there is no mechanism in the systems to address the increasing structural complexities as the systems perform life-long learning. As a result, the systems suffer from an ever-expanding knowledge structure and degradation of interpretability of the systems. Keeping the resultant structures of the systems compact is an important issue because an overly complex system creates a burden on the computational time and memory space required, which is counterbalance to the savings in time and memory by performing learning in an online mode. In addition, highly over-lapping clusters might be present as new clusters are created for incoming data tuples. This, subsequently, causes the fuzzy partition to be unreadable, and degrades the interpretability of

the neuro-fuzzy systems [Guillaume 01].

This chapter presents the *evolving Type-2 neural Fuzzy Inference System* (eT2FIS), a novel evolving Type-2 Mamdani neuro-fuzzy system, which addresses the above-mentioned deficiencies faced by existing systems. An incremental sequential learning scheme is employed for the structure and parameter learning of the eT2FIS model. Through its carefully crafted learning mechanism, eT2FIS is able to transform low-level raw numerical data to high-level human interpretable fuzzy rules. The set of rulebase governing the computational structure is of Type-2 IF-THEN Mamdani-type fuzzy rules [Mamdani 77], which is more promising in handling uncertainties present in noisy learning environments compared to Type-1 models. There are three main operations in the learning of eT2FIS; namely, (1) generation of new fuzzy rules; (2) deletion of obsolete rules; and (3) merger of highly over-lapping fuzzy labels in the input-output spaces. A new rule is created only when a newly arrived data is novel compared to the present knowledge encrypted in the system; and an obsolete rule is deleted when it is no longer relevant to the current learning environment. This addition-removal approach maintains an up-to-date fuzzy rulebase in eT2FIS for the modeling of current dynamics of the environment, where a regulated balance is maintained such that old knowledge (in the system) and new knowledge (from an incoming training tuple) is able to co-define the structure of the model. This addresses the stability-plasticity trade-off of the system; while a balance is maintained between the size of the system and the generalization capabilities of the system. Highly over-lapping fuzzy labels in the input-output spaces are merged to maintain a compact rulebase in eT2FIS, thus improving on the interpretability of the resultant system.

The chapter is organized as follows. The computational structure, reasoning process and neural computations of the eT2FIS model are described in Section 5.1. The proposed learning mechanisms of eT2FIS are introduced in Section 5.2. Section 5.3 evaluates the learning

Figure 5.1: Architecture of the evolving Type-2 neural Fuzzy Inference System (eT2FIS).

and adaptation abilities of eT2FIS through a series of standard benchmark experimental simulations. Lastly, Section 5.4 provides a summary of the key contributions in this chapter.

5.1 eT2FIS: Evolving Type-2 neural fuzzy inference system

This section describes the computational structure, the reasoning process and the neural computations of the proposed *evolving Type-2 neural Fuzzy Inference System* (eT2FIS).

5.1.1 Architecture of eT2FIS

The proposed eT2FIS model is a five layers neuro-fuzzy system as shown in Figure 5.1. Layer 1 consists of the input variable nodes; layer 2 consists of the antecedent nodes; layer

3 is the rule nodes; layer 4 is the consequent nodes; and layer 5 consists of the output variable nodes. In its initial form, there are no fuzzy partitioning in the input-output spaces or fuzzy rules in the system, i.e., there are no nodes in hidden layers 2–4. Learning for eT2FIS is performed incrementally where $x(t) = [x_1(t), \dots, x_p(t), \dots, x_P(t)]^T$ is the input vector at time step t . The corresponding desired output vector is denoted as $d(t) = [d_1(t), \dots, d_q(t), \dots, d_Q(t)]^T$; while the computed output vector is denoted as $y(t) = [y_1(t), \dots, y_q(t), \dots, y_Q(t)]^T$. The notations used in Figure 5.1 are defined as follows:

P	number of input dimensions ;
Q	number of output dimensions ;
I_p	p -th input node ;
O_q	q -th output node ;
$J_p(t)$	number of fuzzy clusters in I_p at time t ;
$L_q(t)$	number of fuzzy clusters in O_q at time t ;
$\tilde{A}_{j_p}$	j -th antecedent fuzzy cluster in I_p ;
$\tilde{C}_{l_q}$	l -th consequent fuzzy cluster in O_q ;
$K(t)$	number of fuzzy rules at time t ;
R_k	k -th fuzzy rule .

Layer 3 of eT2FIS encrypts the rulebase of the system where each rule node encodes a Type-2 IF-THEN Mamdani-type fuzzy rule given as in (5.1):

$$R_k : \text{IF } x_1 \text{ is } \tilde{A}_{j_1}^{(k)} \text{ and } \dots \text{ and } x_P \text{ is } \tilde{A}_{j_P}^{(k)} \quad \text{THEN } y_1 \text{ is } \tilde{C}_{l_1}^{(k)} \text{ and } \dots \text{ and } y_Q \text{ is } \tilde{C}_{l_Q}^{(k)} \quad (5.1)$$

where $\tilde{A}_{j_p}^{(k)}$ (resp. $\tilde{C}_{l_q}^{(k)}$) is the j -th antecedent (resp. l -th consequent) node associated with the p -th input (resp. q -th output) variable that is connected to the rule node R_k . The fuzzy label in each antecedent/consequent node is an interval Type-2 gaussian membership

function (see Figure 2.5), whose FOU is defined as in (2.13). It is re-listed as (5.2) for convenience:

$$\mu(x) = [\underline{\mu}(x), \bar{\mu}(x)] \quad (5.2)$$

where $\underline{\mu}(x)$ and $\bar{\mu}(x)$ are the lower and upper membership functions respectively. They are defined accordingly as in (2.14) and re-listed as (5.3):

$$\underline{\mu}(x) = \begin{cases} \mu^R(c^R, \sigma; x) & \text{if } x \leq 1/2 (c^L + c^R) \\ \mu^L(c^L, \sigma; x) & \text{otherwise} \end{cases} ; \quad \bar{\mu}(x) = \begin{cases} \mu^L(c^L, \sigma; x) & \text{if } x \leq c^L \\ 1 & \text{if } c^L < x \leq c^R \\ \mu^R(c^R, \sigma; x) & \text{if } x > c^R \end{cases} \quad (5.3)$$

where $\mu^L(c^L, \sigma; x)$ and $\mu^R(c^R, \sigma; x)$ are the left and right formation gaussian functions respectively. The gaussian function is defined as in (2.9): $\mu(c, \sigma; x) = e^{-((x-c)^2/\sigma^2)}$ such that c is the centre and σ is the width of the function. Adaptation of the centres is performed using neural network-based gradient descent approach [Haykin 98] [Mendel 01] (see Appendix B); while the widths vary incrementally as eT2FIS performs learning (see Section 5.2.2).

5.1.2 Reasoning process of eT2FIS

As seen from Figure 5.1, the reasoning process of the eT2FIS model is represented by solid arrows where the input vector $x(t)$ is presented to the system at layer 1. The proposed system then performs inference based on the input vector by propagating the information through layers 2 to 4. Consequently, the system produces a computed output vector $y(t)$ at layer 5. The details on the reasoning process of eT2FIS are discussed here.

The generic operations for the reasoning process of eT2FIS are defined as follows: the forward activation functions of each layer $M \in \{1 \dots 5\}$ are denoted as $fa^{(M)}$, and the

corresponding forward output for an arbitrary node is denoted as fo .

Layer 1: The function of the input nodes is to directly pass on the input vector to the next layer. Hence, the neural operations of I_p can be described as in (5.4):

$$fo_p = fa^{(1)}(x_p(t)) = x_p(t) . \quad (5.4)$$

Layer 2: The fuzzy labels $\tilde{A}_{j_p}$ define the antecedent segments of the Type-2 Mamdani-type fuzzy rules described as in (5.1), where each fuzzy label is defined as an interval Type-2 gaussian function described as in (5.2). The function of layer 2 of eT2FIS is to perform similarity matching of the input value with the respective input labels. Hence, the neural computations of $\tilde{A}_{j_p}$ can be described as in (5.5):

$$fo_{j_p} = fa^{(2)}(fo_p) = [\underline{f}_{j_p}, \bar{f}_{j_p}] \quad (5.5)$$

where the bounds of the interval Type-1 output set are computed as in (5.6):

$$\underline{f}_{j_p} = \underline{\mu}_{j_p}(x_p) ; \quad \bar{f}_{j_p} = \bar{\mu}_{j_p}(x_p) . \quad (5.6)$$

such that $\mu_{j_p}(x) = [\underline{\mu}_{j_p}(x), \bar{\mu}_{j_p}(x)]$ refers to the Type-2 membership function embedded in $\tilde{A}_{j_p}$. For simplicity, the time index t has been dropped for the remaining parts of the reasoning process.

Layer 3: The set of Mamdani-type rules that is induced incrementally from the numerical data is defined in the rule layer of the eT2FIS model. Each rule node R_k computes the overall degree of similarity between the input vector and the antecedent part of the k -th

fuzzy rule. Hence, the firing rate of R_k is computed as in (5.7), where

$$fo_k = fa^{(3)} \left(fo_{j_1}^{(k)}, \dots, fo_{j_p}^{(k)} \right) = \left[\underline{f}_k, \bar{f}_k \right] . \quad (5.7)$$

The bounds of the interval Type-1 output set are computed as in (5.8):

$$\underline{f}_k = \min_{p \in \{1 \dots P\}} \underline{f}_{j_p}^{(k)} ; \quad \bar{f}_k = \min_{p \in \{1 \dots P\}} \bar{f}_{j_p}^{(k)} \quad (5.8)$$

where $fo_{j_p}^{(k)} = \left[\underline{f}_{j_p}^{(k)}, \bar{f}_{j_p}^{(k)} \right]$ is the computed output of $\tilde{A}_{j_p}^{(k)}$ described as in (5.5).

Layer 4: This layer of eT2FIS consists of the fuzzy labels $\tilde{C}_{l_q}$ that define the consequent segments of the Mamdani-type fuzzy rules in the system. The function of layer 4 of the system is to perform consequent derivation for the fuzzy rules based on information from the current input vector. Since $\tilde{C}_{l_q}$ may serve as output to more than one fuzzy rule, the cumulative neural computations of $\tilde{C}_{l_q}$ can be described as in (5.9):

$$fol_q = fa^{(4)} \left(fo_1^{(l_q)}, \dots, fo_{K_{l_q}}^{(l_q)} \right) = \left[\underline{f}_{l_q}, \bar{f}_{l_q} \right] \quad (5.9)$$

where K_{l_q} is the total number of fuzzy rules in eT2FIS that shares the same consequent node $\tilde{C}_{l_q}$. The bounds of the interval Type-1 output set are computed accordingly as in (5.10):

$$\underline{f}_{l_q} = \max_{k \in \{1 \dots K_{l_q}\}} \underline{f}_k^{(l_q)} ; \quad \bar{f}_{l_q} = \max_{k \in \{1 \dots K_{l_q}\}} \bar{f}_k^{(l_q)} \quad (5.10)$$

such that $fo_k^{(l_q)} = \left[\underline{f}_k^{(l_q)}, \bar{f}_k^{(l_q)} \right]$ is the output of the k -th fuzzy rule that shares $\tilde{C}_{l_q}$.

Layer 5: The output nodes perform a two-steps computation to obtain a crisp output value; namely, type-reduction and defuzzification. A modified height type-reduction [Driankov, Hellendoom & Reinfrank 96] is adopted in the eT2FIS model, and it is computed using Karnik-Mendel (KM) iterative algorithm [Karnik, Mendel & Liang 99]. Hence, the neural

computations of O_q can be described as in (5.11):

$$fo_q = fa^{(5)}(fo_{1_q}, \dots, fo_{L_q(t)}) = y_q(t) \quad (5.11)$$

where $y_q = 1/2 (\underline{Y}_q + \overline{Y}_q)$ is the defuzzied value of the type reduced set $Y_q = [\underline{Y}_q, \overline{Y}_q]$. The type reduced set is defined as: $\int \dots \int_{\theta_{l_q}} 1 / \left(\sum_{l_q} z_{l_q} \theta_{l_q} / \sum_{l_q} \theta_{l_q} \right)$ such that z_{l_q} is the midpoint of the output domain $\tilde{C}_{l_q}$, i.e., $z_{l_q} = 1/2 (c_{l_q}^L + c_{l_q}^R)$, and $\theta_{l_q} \in fo_{l_q} = [\underline{f}_{l_q}, \overline{f}_{l_q}]$. Applying KM algorithm, $\{z_{l_q}\}$ is sorted in ascending order. This can be achieved using an $L_q(t) \times L_q(t)$ permutation matrix. The corresponding $\{\theta_{l_q}\}$ is subsequently sorted. Please refer to Section 2.2.2 for a detailed implementation of the KM algorithm for determining the type reduced set Y_q .

5.1.3 Neural computations of eT2FIS

The neural computations defined in the proposed eT2FIS model are bi-directional, i.e., in the forward and backward sense. The neural computations of eT2FIS are discussed here.

Forward Operation: As seen from Figure 5.1, the forward operation of eT2FIS coincides with its reasoning path. In particular, the forward operation is defined as the neural computations of layers 1 to 3 of the system.

Backward Operation: The backward operation of eT2FIS is represented by dotted arrows in Figure 5.1, where the desired output vector $d(t)$ is presented to the system in layer 5. The proposed system then passes the information towards the rule layer. The backward operation is a mirrored computation of the forward operation as discussed below.

The generic operations for the backward operation of eT2FIS are defined as follows: the backward activation functions of each layer $M \in \{3 \dots 5\}$ are denoted as $ba^{(M)}$, and the corresponding backward output for an arbitrary node is denoted as bo .

Layer 5: The output nodes directly pass on the output vector to the next layer such that the neural operations of O_q can be described as in (5.12):

$$bo_q = ba^{(5)}(d_q(t)) = d_q(t) . \quad (5.12)$$

Layer 4: The consequent nodes perform similarity matching of the desired output value with the respective output labels such that the neural computations of $\tilde{C}_{l_q}$ can be described as in (5.13):

$$bo_{l_q} = ba^{(4)}(bo_q) = [\underline{b}_{l_q}, \bar{b}_{l_q}] \quad (5.13)$$

where $\underline{b}_{l_q} = \underline{\mu}_{l_q}(d_q)$ and $\bar{b}_{l_q} = \bar{\mu}_{l_q}(d_q)$.

Layer 3: A rule node computes the overall degree of similarity between the desired output vector and the consequent part of the k -th fuzzy rule such that the neural computations of R_k are described as in (5.14):

$$bo_k = ba^{(3)}(bo_{l_1}^{(k)}, \dots, bo_{l_Q}^{(k)}) = [\underline{b}_k, \bar{b}_k] \quad (5.14)$$

where $\underline{b}_k = \min_{q \in \{1 \dots Q\}} \underline{b}_{l_q}^{(k)}$ and $\bar{b}_k = \min_{q \in \{1 \dots Q\}} \bar{b}_{l_q}^{(k)}$.

The forward-and-backward neural computations are defined to compute the activation levels of the fuzzy rules in eT2FIS when a data pair $[x(t), d(t)]$ is presented. The objective is two-folds; namely, to (1) establish the *certainty factors* of fuzzy rules; and (2) determine the creation of a new rule (see Section 5.2.2). Certainty factor of a fuzzy rule reflects the modeling potential of the rule under the current learning environment, and it is defined as in (5.15):

$$\zeta_k(t) = \max[age_k(t), act_k(t)] ; \quad \zeta_k(0) = 1 \quad (5.15)$$

where the certainty factor ζ of a newly created rule k is initialized to unity. This means that

a newly created rule in eT2FIS is assumed the greatest modeling potential since $0 < \zeta \leq 1$. Subsequently, the computation of ζ consists of two parts; namely, the forgetting and the enhancement components. They are computed respectively as in (5.16):

$$age_k(t) = \eta_k \zeta_k(t-1) ; \quad act_k(t) = \min[F(x), B(d)] \quad (5.16)$$

where η_k is a decaying constant; while $F(x) = 1/2 (\underline{f}_k + \bar{f}_k)$ and $B(d) = 1/2 (\underline{b}_k + \bar{b}_k)$ are the forward and backward activations of R_k respectively. The *age* component mimics the biological phenomenon of *long-term depression* (LTD) [Maren & Baudry 95]. Intuitively, information from $[x(t), d(t)]$ can be retained by a one-to-one retention of a fixed synaptic configuration encoding the data pair. However, a more realistic approach is to consider passive decomposition of the configuration because the relevance of a particular data pair is temporal under a dynamically changing environment. On the other hand, the *act* component mimics the biological phenomenon of *long-term potentiation* (LTP) [Maren & Baudry 95]. That means, information storage in the network of a human brain often outlasts its initial synaptic configurations because long-term memory of knowledge can be enhanced through repeated rehearsals of related information. Hence, the modeling potential of a forgotten fuzzy rule can be restored by a repeated rehearsal of related information that first elicited its formation.

Through an incremental update of the certainty factors, eT2FIS is ensured a current and up-to-date rulebase that is able to model the current underlying dynamics of the environment. If a fuzzy rule is able to generalize the recent encountered set of numerical data, the dominating factor in the computation of its certainty factor is the enhancement component. This, subsequently, maintains a high value for the computed certainty factor, thus enhancing the presence of the rule in the fuzzy rulebase. On the other hand, the forgetting mechanism kicks in if a fuzzy rule fails to give a satisfactory generalization of the recent set

of encountered numerical data. Subsequently, the modeling potential of the rule decreases gradually over time until the rule becomes irrelevant to the learning environment or it gets restored through a rehearsal episode.

5.2 Learning mechanisms in eT2FIS

There are two main components to the learning process of the proposed eT2FIS model; namely, structure learning and parameter learning. Three key operations in the structure learning are: (1) generation of new fuzzy rules, (2) deletion of obsolete rules, and (3) merger of highly over-lapping fuzzy labels in the input-output spaces; while parameter learning is performed using backpropagation. Figure 5.2 shows a flowchart of the online sequential learning in eT2FIS. An incremental learning from numerical data is employed for the structure and parameter learning of eT2FIS where the system learns and evolves with the arrival of each new data tuple. Initialization is performed if there are no existing fuzzy rules, i.e. , there are no hidden nodes in the system. On the other hand, structure and parameter learning are activated by the data tuple if there are existing rules in the system. Computational structure of the eT2FIS model is then established, and the system can be employed to model an application or further training is performed. Details on the structure learning mechanism of eT2FIS are presented in this section; and the parameter learning mechanism is presented in Appendix B.

Compared to the generic online learning adopted in neuro-fuzzy systems (see Figure 2.10), the merger operation of highly over-lapping fuzzy labels and the deletion operation of obsolete rules employed in eT2FIS (see Figure 5.2) help to address the issues of an ever-expanding knowledge structure and the lack of interpretability of the system.

Figure 5.2: Flowchart of the incremental learning process employed in eT2FIS.

5.2.1 Initialization

Prior to the commencement of learning, there are no nodes in the hidden layers 2–4 of the proposed eT2FIS model. Learning of the system begins with extracting and utilizing knowledge from the first incoming data tuple $[x(0), d(0)]$ to establish an initial structure of the eT2FIS model. Subsequently, the formation of a first fuzzy cluster $\tilde{A}_{1_p}$ in an input dimension p can be described using its FOU as in (5.17):

$$\mu_{1_p}(x) = \left[\underline{\mu}_{1_p}(x), \bar{\mu}_{1_p}(x) \right] \quad (5.17)$$

where the centres and widths of the left and right formation gaussian functions can be described accordingly as in (5.18):

$$\begin{aligned} c_{1_p}^L &= x_p - \Delta; \quad c_{1_p}^R = x_p + \Delta \\ \sigma_{1_p} &= \sigma_0 \end{aligned} \quad (5.18)$$

such that Δ defines a small perturbation and the width is initialized as σ_0 . The same fuzzy clustering process is performed for each output dimension q to formulate a first fuzzy cluster $\tilde{C}_{1_q}$. In addition, a new fuzzy rule R_1 is formulated to encode the knowledge from $[x(0), d(0)]$ where $\left\{ \tilde{A}_{1_p} \right\}_{p=1}^P$ and $\left\{ \tilde{C}_{1_q} \right\}_{q=1}^Q$ form the antecedent and consequent segments of R_1 respectively.

5.2.2 Rule generation

The proposed eT2FIS model proceeds with the generation of a new fuzzy rule when a data tuple $[x(t), d(t)]$ is presented to a non-empty system at time t . The activation levels of the fuzzy rules are computed using (5.16). If an existing fuzzy rule is able to represent the

presented data, the system proceeds with the update of certainty factors for the fuzzy rules as in (5.15). A fuzzy rule is deemed as being able to represent the presented data if the computed activation level of the rule exceeds a *rule generation threshold* Λ .

On the other hand, a new fuzzy rule is created if existing rules in eT2FIS fail to provide a representation of the presented data, i.e., $act_k \leq \Lambda$ for all k . eT2FIS proceeds to identify the best matched fuzzy clusters in the input dimensions via the computed similarity values. Deriving from (5.5), the similarity value between an input value x_p and an existing cluster in the p -th input dimension is given as $SV(x_p, \tilde{A}_{j_p}) = 1/2 (f_{j_p} + \bar{f}_{j_p})$. Subsequently, the best matched fuzzy cluster in an input dimension p is denoted as $\tilde{A}_{j_p^*}$ where $j_p^* = \arg \max_{j_p} SV(x_p, \tilde{A}_{j_p})$. A new fuzzy rule $R_{K(t+1)}$, where $K(t+1) = K(t) + 1$, is formulated to encode the knowledge from $[x(t), d(t)]$ such that each identified best matched fuzzy cluster can be categorized into three operations as follows:

- 1 *No action is required for the best matched fuzzy cluster and it is defined as part of the antecedent segment of $R_{K(t+1)}$.* This scenario occurs when the similarity value between the presented value and the best matched fuzzy cluster is greater than Λ . That means, the best matched fuzzy cluster is able to represent the presented value.
- 2 *Modify the best matched fuzzy cluster and the modified cluster is defined as part of the antecedent segment of $R_{K(t+1)}$.* This scenario occurs when the computed similarity value is less than Λ , but an updated similarity value between the presented value and the modified best matched fuzzy cluster is greater than Λ . That means, the modified best matched fuzzy cluster is able to represent the presented value although it is initially unable to prior to an update. There are two types of modifications: (a) **Increase the spread of the fuzzy cluster:** The spread between the centres of the left and right formation gaussian functions of the best matched fuzzy cluster is denoted as $s_{j_p^*}$, i.e.,

$s_{j_p^*}(t) = c_{j_p^*}^R(t) - c_{j_p^*}^L(t)$. The spread $s_{j_p^*}$ is then modified as described in (5.19):

$$s_{j_p^*}(t+1) = s_{j_p^*}(t) + \eta \cdot \left(1 - SV\left(x_p, \tilde{A}_{j_p^*}\right)\right) \cdot s_{j_p^*}(t) \quad (5.19)$$

where η is a modification rate. Subsequently, the centres of the left and right functions are updated accordingly as in (5.20):

$$\begin{aligned} c_{j_p^*}^L(t+1) &= c_{j_p^*}^R(t) - s_{j_p^*}(t+1) \quad \text{if } x_p < c_{j_p^*}^L(t) \\ c_{j_p^*}^R(t+1) &= c_{j_p^*}^L(t) + s_{j_p^*}(t+1) \quad \text{if } x_p \geq c_{j_p^*}^R(t) ; \end{aligned} \quad (5.20)$$

(b) **Increase the width of the fuzzy cluster:** The width of the best matched fuzzy cluster is modified as described in (5.21):

$$\sigma_{j_p^*}(t+1) = \sigma_{j_p^*}(t) + \eta \cdot \left(1 - SV\left(x_p, \tilde{A}_{j_p^*}\right)\right) \cdot \sigma_{j_p^*}(t) \quad (5.21)$$

where η is a modification rate. Modification is performed on the best matched cluster by *either* increasing the spread or the width of the membership function under two conditions; namely, updated similarity value between the presented value and the modified best matched fuzzy cluster is greater than Λ , and the updated similarity value results in the largest increase in the computed similarity value. Else, modification to the best matched cluster is performed by increasing *both* its spread and width.

3 A new fuzzy cluster is created and it is defined as part of the antecedent segment of $R_{K(t+1)}$. This scenario occurs when both the computed similarity value and the updated similarity value (as discussed in 2) are less than Λ . That means, the presented value is novel as compared to existing clusters in the system. The formation of a new fuzzy cluster $\tilde{A}_{J_p(t+1)}$, where $J_p(t+1) = J_p(t) + 1$, in the p -th input dimension can be described using

its FOU as in (5.22):

$$\mu_{J_p(t+1)}(x) = \left[\underline{\mu}_{J_p(t+1)}(x), \bar{\mu}_{J_p(t+1)}(x) \right] \quad (5.22)$$

where the centres and widths of the left and right formation gaussian functions can be described accordingly as in (5.23):

$$\begin{aligned} c_{J_p(t+1)}^L &= x_p - \Delta ; \quad c_{J_p(t+1)}^R = x_p + \Delta \\ \sigma_{J_p(t+1)} &= \begin{cases} \sigma^R & \text{if } j_p^L = \text{NULL} \\ \sigma^L & \text{if } j_p^R = \text{NULL} \\ R(\sigma^R, \sigma^L) & \text{otherwise} \end{cases} \end{aligned} \quad (5.23)$$

such that

$$\sigma^R = R \left(\sqrt{-\frac{(c_{j_p^R} - x_p)^2}{\log \alpha}}, \sigma_{j_p^R}(t) \right) ; \quad \sigma^L = R \left(\sqrt{-\frac{(c_{j_p^L} - x_p)^2}{\log \alpha}}, \sigma_{j_p^L}(t) \right) .$$

The width of the newly created fuzzy cluster is determined based on the notion of behavioral category learning process demonstrated by humans (see Chapter 3). $R(\sigma_1, \sigma_2) = 1/2 (\sigma_1 + \sigma_2)$ defines a regulator function that ensures a newly created fuzzy cluster has distinct semantic meaning; while α is a minimum membership threshold. Following Chapter 3, the minimum membership threshold α is fixed as 0.2. The immediate left and right neighbours of the newly created cluster are denoted as $\tilde{A}_{j_p^L}$ and $\tilde{A}_{j_p^R}$ respectively,

where

$$j_p^L = \begin{cases} \text{NULL} & \text{if } c_{j_p} \geq x_p \text{ for } 1 \leq j_p \leq J_p(t) \\ \arg \min_{c_{j_p} < x_p} |c_{j_p} - x_p| & \text{otherwise} \end{cases}$$

$$j_p^R = \begin{cases} \text{NULL} & \text{if } c_{j_p} \leq x_p \text{ for } 1 \leq j_p \leq J_p(t) \\ \arg \min_{c_{j_p} > x_p} |c_{j_p} - x_p| & \text{otherwise} \end{cases}$$

such that $c_{j_p} = 1/2 (c_{j_p}^L + c_{j_p}^R)$. Refinements are subsequently made to the immediate left and right neighbours of the newly created cluster as follows:

1. If the newly created cluster has no left neighbour (i.e. $j_p^L = \text{NULL}$), then only the right neighbour is updated:

$$\sigma_{j_p^R}(t+1) = \sigma_{J_p(t+1)} .$$

2. If the newly created cluster has no right neighbour (i.e. $j_p^R = \text{NULL}$), then only the left neighbour is updated:

$$\sigma_{j_p^L}(t+1) = \sigma_{J_p(t+1)} .$$

3. If the newly created cluster has both left and right neighbours, then they are updated:

$$\sigma_{j_p^L}(t+1) = \sigma_{j_p^R}(t+1) = \sigma_{J_p(t+1)} .$$

Figure 5.3 illustrates the fuzzy partitioning process in an input-output dimension of the proposed eT2FIS model. Shown in Figure 5.3(a) is the existing clusters in the input-output space. Subsequently, a novel data point is encountered in Figure 5.3(b) where the computed similarity values between the presented point and the existing clusters fall below Λ . In addition, it should be checked that the computed similarity

Figure 5.3: Fuzzy partitioning steps performed in eT2FIS.

values between the presented point and modified versions of the existing clusters also fall below Λ . A new cluster is then created using information derived from this novel data point as seen in Figure 5.3(c). The spreads of the cluster on either side of the centre depend on the distance of the presented point to the respective centres of its immediate neighbours. The centres of the immediate neighbours have the minimum membership value α . Regulation is performed to preserve a distinct semantic meaning of the newly created cluster as shown in Figure 5.3(d). Existing clusters are simultaneously refined to incorporate the newly created cluster.

The same identification process is performed for each output dimension to determine the consequent segment of $R_{K(t+1)}$. Following that, certainty factors of the fuzzy rules are computed based on the modified structure of the eT2FIS model.

Figure 5.4: A. Computation of similarity match between two fuzzy clusters; and B. Merger process of two highly over-lapping fuzzy labels in eT2FIS.

5.2.3 Merger of fuzzy labels

The second operation in the structure learning of eT2FIS is the merger of highly over-lapping fuzzy labels in the input-output spaces. The objective is two-folds; namely, to improve the semantic interpretations of the fuzzy clusters, and to reduce the computational complexity of the system with a more compact rulebase. The system proceeds to identify the most similar fuzzy labels in the input dimensions via similarity matches. The similarity match between two existing clusters in the input dimension p is denoted as $SM(\tilde{A}_{j_p^1}, \tilde{A}_{j_p^2})$ and its computation is described as in (5.24):

$$SM(\tilde{A}_{j_p^1}, \tilde{A}_{j_p^2}) = \min\left(A_{j_p^1 \cap j_p^2} / A_{j_p^1}, A_{j_p^1 \cap j_p^2} / A_{j_p^2}\right) \quad (5.24)$$

where $A_{j_p^m}$ is the area enclosed by the isosceles triangle with centre $c_{j_p^m} = 1/2 \left(c_{j_p^m}^L + c_{j_p^m}^R \right)$ and base $4\sigma_{j_p^m}\sqrt{\ln 2}$, $m = 1, 2$; and $A_{j_p^1 \cap j_p^2}$ is the triangular area enclosed by the intersection of the two isosceles triangles. Figure 5.4A. illustrates the graphical meanings of the definitions given in (5.24). Subsequently, the most similar fuzzy labels in an input dimension p are denoted as $\tilde{A}_{j_p'}$ and $\tilde{A}_{j_p''}$ where the similarity match between them is the greatest among all existing clusters in the input dimension p .

The most similar fuzzy clusters $\tilde{A}_{j_p'}$ and $\tilde{A}_{j_p''}$ are merged if the similarity match between them exceeds a *merger threshold* Ω , i.e., $SM \left(\tilde{A}_{j_p'}, \tilde{A}_{j_p''} \right) \geq \Omega$. Figure 5.4B. illustrates the merger process of two highly over-lapping fuzzy labels. The merged fuzzy label is denoted as $\tilde{A}_{j_p' \cup j_p''}$ and it can be described using its FOU as in (5.25):

$$\mu_{j_p' \cup j_p''}(x) = \left[\underline{\mu}_{j_p' \cup j_p''}(x), \bar{\mu}_{j_p' \cup j_p''}(x) \right] \quad (5.25)$$

where the centres and widths of the left and right formation gaussian functions can be described accordingly as in (5.26):

$$\begin{aligned} c_{j_p' \cup j_p''}^L &= 1/2 \left(c_{j_p'}^L + c_{j_p''}^L \right) ; \quad c_{j_p' \cup j_p''}^R = 1/2 \left(c_{j_p'}^R + c_{j_p''}^R \right) \\ \sigma_{j_p' \cup j_p''} &= 1/2 \left(\sigma' + \sigma'' \right) \end{aligned} \quad (5.26)$$

such that

$$\sigma' = \frac{c_{j_p' \cup j_p''}^L - c_{j_p'}^L + \sigma_{j_p'} \sqrt{\ln 2}}{\sqrt{\ln 2}} ; \quad \sigma'' = \frac{c_{j_p' \cup j_p''}^R - c_{j_p''}^R + \sigma_{j_p''} \sqrt{\ln 2}}{\sqrt{\ln 2}} .$$

Without loss of generality, it is assumed that $c_{j_p'} < c_{j_p''}$ where $c_{j_p} = 1/2 \left(c_{j_p}^L + c_{j_p}^R \right)$.

The same merger process is repeated for each output dimension to merge highly over-lapping fuzzy labels in the output spaces. Subsequently, the proposed eT2FIS model performs

consistency check on the rulebase. An inconsistent rulebase occurs when there are two rules in the rulebase of the system such that the antecedent conditions are similar but the resultant consequences differ. If the rulebase is inconsistent, then inconsistent rules with lower certainty factors are deleted. This approach not only ensures that the resultant rulebase is consistent, it also ensures that eT2FIS provides the most aptly description of the current learning environment.

5.2.4 Deletion of obsolete rules

The last operation in the structure learning of eT2FIS is the deletion of obsolete rules. This is necessary to remove irrelevant rules that do not participant in the modeling of the current learning environment, hence ensuring an up-to-date rulebase that is able to model the current underlying dynamics of the environment. This approach also spares the proposed system from an explosively increasing rulebase by maintaining a compact rulebase as it performs a life-long incremental learning of the environment.

As explained previously, the certainty factor of a fuzzy rule is high if it is able to generalize the recent encountered set of numerical data. The inverse is also true where the certainty factor of a fuzzy rule is low if it fails to give a satisfactory generalization of the recent set of encountered numerical data. Consequently, a fuzzy rule in eT2FIS is regarded as an obsolete rule if its certainty factor falls below a *deletion threshold* Γ , i.e. , delete R_k if $\zeta_k(t) < \Gamma$ such that $K(t+1) = K(t) - 1$. Finally, some of the fuzzy labels might be “orphaned” when all fuzzy rules associated with them have been deleted. The orphaned fuzzy labels are then removed to ensure that the resultant computational structure of eT2FIS is compact.

5.3 Experimental results and analysis

This section illustrates the learning and adaptation abilities of the proposed eT2FIS model by employing it in two benchmarking datasets; namely, (1) identification of a nonlinear system [Juang & Lin 98]; and (2) prediction of a chaotic system [Mackey & Glass 77].

5.3.1 Example 1 – Identification a nonlinear system

There are three parts to this application problem [Juang & Lin 98]; namely, (a) the online identification of a non-time-varying system; (b) the online identification of a time-varying system; and (c) the identification of a nonlinear noisy system.

Online identification of a non-time-varying system

This experiment generalizes the dynamics of a nonlinear system, where the dataset is generated by a difference equation as described in (3.10). The problem formulation has previously been explained in Section 3.3.1. However, unlike Section 3.3.1 where learning is performed in a batch manner from numerical data, the eT2FIS model is applied to identify the nonlinear system in an online mode, i.e. , no prior offline knowledge is used to train the model, and the model performs structure and parameter learning upon the arrival of each incoming data tuple with a single pass.

The thresholds for the three key operations in the structure learning of eT2FIS are set as follows: rule generation threshold, $\Lambda = 0.4$; merger threshold, $\Omega = 1$; and deletion threshold, $\Gamma = 0$. In addition, the spreads between the left and right formation functions of the fuzzy labels in the model are initialized to zero, since there is no noise added to this problem. This means that the proposed model is being used as an evolving Type-

Figure 5.5: (a)–(c): Fuzzy partitioning achieved for the online identification of a non-time-varying nonlinear system; and (d) Identified fuzzy rulebase in eT2FIS.

1 Mamdani neuro-fuzzy system. Subsequently, 6, 4 and 3 fuzzy labels are identified by eT2FIS for this two inputs-one output application problem respectively. Figures 5.5(a) and (b) illustrate the identified fuzzy clusters in the input spaces $u(t)$ and $y(t)$ respectively; and Figure 5.5(c) illustrates the identified fuzzy clusters in the output space $y(t+1)$. As seen, intuitive semantic labels in the input-output spaces can be directly identified from the raw numerical data using the eT2FIS model. Subsequently, the model identifies a total of 14 fuzzy rules at the end of the experiment using the derived semantic fuzzy labels and they are listed in Figure 5.5(d). One can easily verify that the derived rulebase is consistent.

Since the identification process of the nonlinear system is performed in an online manner from the numerical data, the evolving nature of the proposed eT2FIS model can be seen from the changes in the size of the identified rulebase as the experiment is conducted. This is shown in Figure 5.6(a). As seen, the number of fuzzy rules identified climbs steadily as eT2FIS performs learning on the nonlinear system. New rules are identified when the data encountered are novel. This continues until the learning stabilises, and the number of

Figure 5.6: (a) Number of fuzzy rules identified; and (b) Modeling results for the online identification of a non-time-varying nonlinear system using eT2FIS.

identified fuzzy rules then remains constant, such that the final number of rules identified is 14. Subsequently, the learned eT2FIS model is used to perform prediction on the unseen testing data. The result of the generalization ability of the model is seen in Figure 5.6(b). There is an almost perfect match between the computed outputs of the model and the desired outputs of the system, with eT2FIS achieving a root mean squared error (RMSE) of 0.053 on the testing data.

This modeling problem focuses on a non-time-varying system, thus the deletion operation is not necessary and it is switched off by setting $\Gamma = 0$. This can be seen from Figure 5.6(a), where the number of identified fuzzy rules increases over the course of the experiment (compare this observation with that obtained in the second part of the application problem where eT2FIS is used to perform the online identification of a time-varying system, see Figure 5.8). To study the effects of the *rule generation threshold*, Λ , and the *merger threshold*, Ω , on the modeling ability of the proposed eT2FIS model, different values are tested. The test results concerning the number of identified fuzzy labels, fuzzy rules and the RMSE on the testing data are listed in Tables 5.2 and 5.3. Looking at Table 5.2, it is observed that eT2FIS performs consistently within certain ranges of Λ . When Λ is set to a low value, the modeling ability of eT2FIS suffer because fewer rules are created. On the other extreme, when Λ is set to a high value, eT2FIS performs better with a lower reported RMSE value, but at the expense of a larger rulebase. In general, a trade-off between interpretability

Table 5.2: Sensitivity test for different values of Λ in eT2FIS.

Λ	# Fuzzy Labels			# Rules	RMSE
0.30	Inputs =	3, 4;	Output =	3	0.091
0.35	Inputs =	4, 4;	Output =	3	0.068
0.40	Inputs =	6, 4;	Output =	3	0.053
0.45	Inputs =	5, 5;	Output =	4	0.048
0.50	Inputs =	8, 5;	Output =	4	0.033

$$\Omega = 1 ; \quad \Gamma = 0$$

and accuracy is needed [Ang & Quek 05], and this is achieved in eT2FIS as follows: better performance is expected when Λ is high, with a decrease in the interpretability of the model due to a larger rulebase; while a better interpretability is achieved with a low Λ , resulting in a decrease in the modeling ability. Meanwhile, the number of identified fuzzy labels also increase with an increasing Λ value. When $\Lambda = 0.5$, as much as eight fuzzy clusters are identified for the input space $u(t)$. To achieve a more compact structure, merging of highly similar/over-lapping fuzzy clusters can be performed on the model. The test results using different values of Ω are listed in Table 5.3. As seen, the performance of eT2FIS is consistent when very similar fuzzy clusters are merged. The result is a smaller rulebase, and a lesser number of fuzzy clusters identified, i.e., a more compact computational structure is achieved. However, when Ω is over-relaxed, not very similar clusters might be merged, leading to a significant jump in the RMSE value. Ideally, merging should only be performed for highly over-lapping fuzzy clusters, and this can be achieved by setting a larger merger threshold.

The performance of the proposed eT2FIS model is subsequently compared against the following evolving neuro-fuzzy systems; namely, SONFIN [Juang & Lin 98]; SAFIS [Rong et. al. 06]; eTS [Angelov & Filev 04]; FALCON-ART [Lin & Lin 97]; EFuNN [Kasabov 01]; and eFSM [Tung & Quek 10]. The benchmarking measure is the RMSE achieved on the testing

Table 5.3: Sensitivity test for different values of Ω in eT2FIS.

Ω	# Fuzzy Labels			# Rules	RMSE	
0.9	Inputs =	8, 5;	Output =	4	21	0.033
0.8	Inputs =	7, 5;	Output =	4	20	0.032
0.7	Inputs =	6, 5;	Output =	4	17	0.034
0.6	Inputs =	4, 5;	Output =	4	12	0.055
0.5	Inputs =	4, 3;	Output =	2	8	0.114
$\Lambda = 0.5 ; \quad \Gamma = 0$						

Table 5.4: Results of online identification of a non-time-varying system using eT2FIS.

Model	Type	# Rules	RMSE
SONFIN	TSK	10	0.013
SAFIS	TSK	8	0.012
eTS	TSK	19	0.008
FALCON-ART	Mamdani	289	0.138
EFuNN	Mamdani	18	0.058
eFSM	Mamdani	14	0.073
eT2FIS	Mamdani	14	0.053

data. Table 5.4 shows the benchmarking results of the models. From the table, it can be seen that TSK-type systems generally perform better with smaller RMSE values, using a similar number of rules. However, this is not unexpected since TSK-type systems are generally more expressive and accurate compared to Mamdani-type models, focusing on achieving good modeling performances at the expense of a opaque system structure [Tung & Quek 10]. On the other hand, Mamdani systems focus on the interpretability of systems, and this is well-demonstrated in eT2FIS as seen from Figure 5.5. Comparing within the Mamdani-type systems, the proposed eT2FIS model is able to deliver an outstanding performance, both in terms of the lowest RMSE value achieved and the lowest number of rules used. This result illustrates the excellent generalization ability of the proposed model, with an equally well-balanced interpretability maintained in the model.

Online identification of a time-varying system

This experiment is performed to demonstrate the evolving ability of the proposed eT2FIS model, with the dataset generated as in (5.27):

$$y(t+1) = \frac{y(t)}{1+y^2(t)} + u^3(t) + f(t) \quad (5.27)$$

where a disturbance $f(t)$ is introduced into the system in (3.10), such that $f(t)$ is described as in (5.28):

$$f(t) = \begin{cases} 0 & \text{for } 1 \leq t \leq 1000 \text{ and } t \geq 2001 \\ 1 & \text{for } 1001 \leq t \leq 2000 \end{cases} . \quad (5.28)$$

The disturbance is introduced at $t = 1001$, and subsequently removed at $t = 2001$. Since this is a modeling problem concerning the identification of a time-varying system, the eT2FIS model is evaluated in an online mode. The thresholds for the three key operations in the structure learning of eT2FIS are set as follows: $\Lambda = 0.4$; $\Omega = 1$; $\Gamma = 0$.

Figure 5.7 illustrates the online modeling performance of eT2FIS, zooming in on the modeling of eT2FIS when the disturbance $f(t)$ is added in part (a), and removed in part (b). As seen, there are very good matches between the computed output of eT2FIS and the desired output of the system, both after $f(t)$ is introduced at $t = 1001$ and removed at $t = 2001$. This indicates that the proposed model is able to track changes in the application environment well, adapting to the changes in an efficient manner. Overall, eT2FIS achieves a RMSE of 0.180 over the duration $t \in [1, 3000]$. Subsequently, the change in the number of fuzzy rules identified by eT2FIS throughout the course of online identification of this time-varying system is illustrated in Figure 5.8(a). From the figure, it can be seen that there is an increment in the number of fuzzy rules each time the dynamics of the application problem changes, i.e., when disturbance is added or removed. Since the deletion threshold Γ has been set to zero, the deletion mechanism is switched off. This is illustrated in Figure 5.8(a),

Figure 5.7: Modeling performance of eT2FIS when disturbance is (a) added; and (b) removed.

Figure 5.8: Number of fuzzy rules identified by eT2FIS when: (a) no deletion is activated; and (b) deletion is activated.

where the number of rules can only increase correspondingly to changes in the underlying dynamics of the application. At the end of the course of this experiment, a total of twelve rules have been identified by the eT2FIS model. Considering the short modeling span of 3000 time steps, the size of the fuzzy rulebase is still interpretable. However, the model might be burdened by a very large rulebase if a longer modeling time span is performed for time-varying application problems, affecting the interpretability of the model. A regulation on the size of the identified fuzzy rulebase is thus desirable in such a situation.

To study the effect of the *deletion threshold*, Γ , on the modeling ability of the eT2FIS model, different values of Γ are tested, with the rest of the pre-selected variables unchanged. Deletion of obsolete rules is a feature in the proposed model to tackle the prevailing issue of an ever-expanding structure in evolving neuro-fuzzy systems. As before, the test results concerning the number of identified fuzzy labels, fuzzy rules and the RMSE values achieved

Table 5.5: Sensitivity test for different values of Γ in eT2FIS.

Γ	# Fuzzy Labels			# Rules	RMSE	
0.0	Inputs =	6, 3;	Output =	5	12	0.180
0.1	Inputs =	7, 3;	Output =	5	11	0.172
0.2	Inputs =	6, 3;	Output =	4	11	0.175
0.3	Inputs =	5, 3;	Output =	4	8	0.174
0.4	Inputs =	10, 3;	Output =	4	13	0.202

$$\Lambda = 0.4 ; \quad \Omega = 1$$

are listed in Table 5.5. It is observed that eT2FIS performs consistently when the values of Γ are kept small. The identified numbers of fuzzy labels and fuzzy rules become more compact when deletion is performed, thus improving on the interpretability of the model. However, there is a great jump in the RMSE value when Γ is set too large. This is because, for this particular time-varying problem, the rate of change of the underlying dynamics is relatively slow. When the rate of updating the fuzzy rules through a deletion of obsolete rules is too rapid, the model could end up forgetting the knowledge too soon after it has been learned. In general, the deletion threshold is set proportional to the rate of change of the learning environment's dynamics, i.e., a small Γ is sufficient for a slow changing application problem, while a larger Γ is needed for a rapid changing application problem. In addition, deletion operation in eT2FIS can be switched off by setting $\Gamma = 0$ when the application problem is non-time-varying as illustrated in part 1 of this section. Subsequently, the change in the number of rules identified by eT2FIS throughout the course of online identification of the time-varying system for $\Gamma = 0.3$ is illustrated in Figure 5.8(b). As seen, there is a good regulation on the number of fuzzy rules identified by eT2FIS throughout the course of the experiment. An increase in the number of fuzzy rules is observed whenever $f(t)$ is added or removed. Obsolete rules that do not participant in the modeling of the current environment are then deleted. This can be seen from the decrease in the number of rules, following each increase. At the end of this experiment, a more compact fuzzy rulebase (consisting of 8

Table 5.6: Performances of eT2FIS, eT1FIS-V1 and eT1FIS-V2 for the identification of a nonlinear noisy system.

Noise level		[−0.1, 0.1]		[−0.5, 0.5]	
Model	# Epochs	# Rules	Ave. RMSE $\pm$ SD	# Rules	Ave. RMSE $\pm$ SD
eT2FIS	200	10	0.034 $\pm$ 0.002	11	0.138 $\pm$ 0.004
eT1FIS-V1	500	12	0.048 $\pm$ 0.003	16	0.153 $\pm$ 0.007
eT1FIS-V2	500	10	0.063 $\pm$ 0.002	11	0.170 $\pm$ 0.004

rules) is identified, while the modeling performance is maintained (as seen from Table 5.5, where RMSE = 0.174).

Identification of a nonlinear noisy system

This experiment is performed to illustrate the noise resistance ability of eT2FIS, where the same nonlinear system in (3.10) is considered. A total of 200 input-output data pairs are generated. Following the experimental description in [Juang & Tsao 08b], it is assumed that each measured $y(t)$ contains noise, and the noisy value is denoted as $y_n(t)$. The added noise is artificially generated white noise with uniform distribution. Simulations with noise being generated in the ranges $[-0.1, 0.1]$ and $[-0.5, 0.5]$ are conducted. Noise is added to the original 200 clean input-output data pairs. For training, the inputs are $y_n(t)$ and $u(t)$; while the desired output is $y_n(t+1)$. After training, another set of noise is added to the original 200 clean data pairs, and noisy values $y_n(t)$ are fed as inputs to test the noise resistance ability of eT2FIS. Subsequently, the RMSE between the computed output of eT2FIS and the original desired clean output $y(t+1)$ of the system is calculated. There are 20 Monte Carlo realizations for this experiment, where the mean and standard deviation values are averaged. The learning is performed in an offline manner where the training data is learned by cycling through it a number of epochs.

In the first part of the benchmarking, eT2FIS is compared against two variations of its

Type-1 counterparts. For the first variation, an evolving Type-1 Mamdani model is built using the same learning mechanisms, where the spreads between the left and right formation functions of the fuzzy labels in the model are initialized to zero; and the model is referred to as eT1FIS-V1. For the second variation, an eT2FIS model is built using the proposed learning mechanisms, where the testing model is then converted to a Type-1 model by replacing all the Type-2 fuzzy sets by Type-1 fuzzy sets (i.e. centre $c = 1/2 (c^L + c^R)$; width σ remains unchanged); and the model is referred to as eT1FIS-V2. For noise level $[-0.1, 0.1]$, the thresholds in eT2FIS are set as follows: $\Lambda = 0.35$; $\Omega = 0.55$; $\Gamma = 0$; and the initial spreads between the left and right functions of the fuzzy labels are 0.02; while considering a greater noise level $[-0.5, 0.5]$, Λ is reduced to 0.3 and the initial spreads between the left and right functions of the fuzzy labels are increased to 0.024. Table 5.6 shows the performances of the three models. The Type-2 model outperforms both versions of its Type-1 counterparts under the two different noise levels, where lower RMSE values are reported. This is on top of the lesser number of training epochs required by eT2FIS. With the other pre-selected variables unchanged (except the spreads of the fuzzy labels), eT1FIS-V1 generally identifies more rules. Although this helps in reducing the RMSE achieved, the model is not as stable compared to eT2FIS as seen from the larger standard deviations of eT1FIS-V1. On the other hand, eT1FIS-V2 inherits the fuzzy rules from eT2FIS, with the fuzzy labels replaced as Type-1 sets. Since the original model is built with the aim of creating a Type-2 model, this could explain the poorest performance of eT1FIS-V2 among all the models. On the flip side, the performance of eT1FIS-V2 is more stable compared to eT1FIS-V1 as seen from the similar standard deviations achieved by eT1FIS-V2 from eT2FIS. This result illustrates the better and more stable noise resistance ability of the proposed eT2FIS model compared to its Type-1 counterparts, indicating that it is more robust when learning under noisy application environments.

As an illustration, Figure 5.9(a)–(c) show the identified fuzzy clusters in the input-output

Figure 5.9: (a)–(c) Fuzzy partitioning achieved for the identification of a nonlinear noisy system with noise level $[-0.1, 0.1]$; and (d) Identified fuzzy rulebase in eT2FIS.

Figure 5.10: Modeling performance for eT2FIS when noise level is (a) $[-0.1, 0.1]$; and (b) $[-0.5, 0.5]$.

Table 5.7: Performances of eT2FIS and the benchmarking models for the identification of a nonlinear noisy system.

Noise level		[−0.1, 0.1]		[−0.5, 0.5]	
Model	# Epochs	# Rules	Ave. RMSE	# Rules	Ave. RMSE
SONFIN	500	6	0.041	6	0.170
T2SONFS	500	6	0.034	6	0.138
eT2FIS	500	10	0.033	11	0.140

spaces for a single run of the noise level $[-0.1, 0.1]$; and Figure 5.9(d) lists the corresponding fuzzy rulebase derived from the semantic fuzzy labels. As seen, the fuzzy clusters identified are Type-2 fuzzy sets and they are highly ordered such that semantic fuzzy labels can be attached to the fuzzy clusters. In addition, the spreads and widths of the identified fuzzy clusters differ, depending on the numerical data presented. This tailored approach for the numerical data helps to derive meaningful and intuitive fuzzy labels in the input-output spaces, hence enhancing the interpretability of the resultant eT2FIS model. One can also verify that the derived fuzzy rulebase is consistent. Subsequently, the modeling performance of the proposed eT2FIS model on a single run of the noise level $[-0.1, 0.1]$ is shown in Figure 5.10(a). The match between the computed outputs of eT2FIS and the desired outputs of the system is perfect, with eT2FIS achieving a RMSE value of 0.035. Also included in Figure 5.10 is the modeling performance of eT2FIS on a single run of the noise level $[-0.5, 0.5]$. As expected, there are more fluctuations in the computed outputs when the noise level increases, since there are greater uncertainties in noisier learning environments. This is reflected in the increase in the RMSE value achieved. Despite that, the proposed model performs good modeling as seen from Figure 5.10(b), where the computed outputs of eT2FIS are able to model the general trend of the application problem such that both the peaks and the valleys of the data are well captured.

In the second part of the benchmarking, the performance of the proposed eT2FIS model is compared against the following evolving neuro-fuzzy systems; namely, a Type-1 system

– SONFIN [Juang & Lin 98]; and a Type-2 Mamdani system – T2SONFS [Juang & Tsao 08b]. Table 5.7 shows the results of the comparison. As a fair comparison, 500 epochs of training are performed for all the benchmarking models. Being a Type-1 model, SONFIN performs the poorest among the three models for both of the different noise levels. On the other hand, both the Type-2 Mamdani models perform comparably in terms of the number of fuzzy rules identified, and the RMSE values achieved. In fact, it should be noted that the proposed model is able to achieve similar performances within 200 epochs of training (see Table 5.6). This indicates that eT2FIS is able to perform fast learning of the application environment, while achieving good modeling performances.

From this experiment, the proposed Type-2 eT2FIS model has demonstrated better performances compared to both its Type-1 counterparts, as well as generic Type-1 models, under noisy learning environments; while achieving comparable performance compared to corresponding Type-2 models.

5.3.2 Example 2 – Prediction of a chaotic time-series

The noise resistance ability of the proposed eT2FIS model is evaluated by employing it in a benchmark comparison involving the prediction of a chaotic time-series with noise. The original chaotic time-series is generated by a delay differential equation as described in (5.29):

$$\frac{\partial x(t)}{\partial t} = \frac{0.2 x(t - \tau)}{1 + x^{10}(t - \tau)} - 0.1 x(t) \quad (5.29)$$

which was first investigated by Mackey and Glass [Mackey & Glass 77]. Following the approach in [Jang 93], a fourth-order Runge-Kutta method was applied to compute the numerical approximation of the series with conditions as follows: $\tau = 30$ and initial condition

$x(0) = 1.2$. Four past values are used to predict the present value, where

$$\text{input vector} = [x(t-24), x(t-18), x(t-12), x(t-6)] ; \quad \text{output vector} = [x(t)] .$$

A total of 1000 data pairs are extracted from the interval $t \in [124, 1123]$. The first 500 pairs are then used as training set; while the remaining 500 pairs are used for testing. In the training part, Gaussian white noise with mean 0 and standard deviation 0.1 is added to the time-series x , and the noisy series is denoted as x_n ; while in the testing part, two testing sets are created by adding Gaussian white noise with mean 0 and standard deviations 0.1 and 0.3 to x respectively. Subsequently, the RMSE between the computed output of eT2FIS and the original desired clean output $x(t)$ of the time-series is calculated. There are 10 Monte Carlo realizations for this experiment. The performance of the proposed eT2FIS model is benchmarked against the following evolving neuro-fuzzy systems; namely, Type-1 models – SONFIN [Juang & Lin 98]; DENFIS [Kasabov & Song 02]; EFuNN [Kasabov 01]; and Type-2 models – SEIT2FNN [Juang & Tsao 08a]; IT2FNN-SVM [Juang, Huang & Cheng 10]. The thresholds in the eT2FIS model are set as follows: $\Lambda = 0.25$; $\Omega = 1$; $\Gamma = 0$; and the initial spreads between the left and right functions of the fuzzy labels are 0.08.

Figure 5.11 illustrates the input spaces fuzzy partitioning achieved by the eT2FIS model for a run of the noisy chaotic time-series. As seen, the identified fuzzy clusters are highly ordered such that clear semantic fuzzy labels can be attached to the fuzzy clusters. This subsequently allows a good interpretability of the learned model.

One run of the testing datasets for the two noise levels are illustrated in Figure 5.12. As seen, the modeling performance of eT2FIS is good when the noise level is low, achieving a RMSE of 0.035 on the testing data. As expected, the performance of the model deteriorates when it is tested on a higher noise level, where the RMSE is reported at a higher 0.144. More fluctuations in the computed output of eT2FIS are also observed. Nevertheless, the

Figure 5.11: Input spaces fuzzy partitioning for the noisy chaotic series in eT2FIS.

Figure 5.12: Modeling performance for eT2FIS when noise level is (a) STD = 0.1; and (b) STD = 0.3.

Table 5.8: Average performances of the identification of a chaotic time-series using eT2FIS.

Testing noise level		STD = 0.1	STD = 0.3
Model	# Rules	Ave. RMSE	Ave. RMSE
SONFIN	10	0.108	0.256
DENFIS	50	0.093	0.222
EFuNN	374	0.137	0.226
SEIT2FNN	5	0.097	0.212
IT2FNN-SVM	6	0.087	0.193
eT2FIS	16	0.105	0.195

performance is still satisfactory, where the peaks and valleys of the desired output have been well captured by the proposed model.

Table 5.8 shows the consolidated average performances of the eT2FIS model, together with the benchmarking models, for this prediction of chaotic time-series task. In general, the testing RMSE values increase as the noise levels in the testing sets increase. As seen, most of the models achieve comparable results, in terms of the identified number of fuzzy rules and the RMSE achieved on the testing data, when the noise level is low. This is with the exception of EFuNN, where the number of fuzzy rules identified runs into the hundreds, and the RMSE achieved is also the largest. Subsequently, the greater noise resistance abilities of the Type-2 models are demonstrated when the models are tested on a testing set with a higher noise level. The top performers for noise level 0.3 are the Type-2 models, with the proposed eT2FIS model achieving the second position. Nevertheless, it should be noted that both the other Type-2 models in this benchmark are TSK models, which are focused on the accuracies of the models. On the other hand, eT2FIS is a Mamdani model, which is focused on the interpretability of the model. This is shown by the well-ordered distinct fuzzy semantic labels identified in eT2FIS as illustrated in Figure 5.11. From this comparison, the eT2FIS model has demonstrated good modeling abilities (compared to existing neuro-fuzzy systems) despite the presence of noise, such that the interpretability of the model is not

compromised.

5.4 Summary

This chapter presents the proposed *evolving Type-2 neural fuzzy inference system* (eT2FIS) – a novel Type-2 Mamdani neuro-fuzzy system with online learning mechanism. With regards to existing evolving neuro-fuzzy systems, eT2FIS is one of the first few attempts in unifying the concepts of evolving neuro-fuzzy systems with Type-2 Mamdani models. Learning is incrementally performed in eT2FIS such that the system evolves and learns with the arrival of a new data pair. This allows the proposed system to progressively incorporate new knowledge; hence, provides a life-long learning for the underlying dynamics of an application environment. The eT2FIS model addresses the stability-plasticity trade-off of a learning system by drawing inspiration from the human biological memory retention mechanism. This is demonstrated through the learning of eT2FIS where positive associations between new knowledge from the environment and existing knowledge in the system enhances the related stored knowledge (i.e. through a biological phenomenon known as long-term potentiation); while negative associations between the new knowledge from the environment and existing knowledge in the system degrades the related stored knowledge (i.e. through a biological phenomenon known as long-term depression). Subsequently, a new rule is created only when the knowledge from the environment is novel with respect to the stored knowledge in the system; while an obsolete rule is deleted when it is no longer actively participating in the modeling of the current environment. This addition-removal mechanism provides a regulated balance between old knowledge in the system and new knowledge from incoming data, such that old and new knowledge can co-exist to define the structure of the system. This helps to curb the problem of an ever-expanding knowledge structure in existing evolving neuro-fuzzy systems. Another key feature in the proposed

eT2FIS model is the merger of highly over-lapping fuzzy labels in the input-output spaces. This ensures a compact and more interpretable structure of the eT2FIS model. Finally, eT2FIS is employed in two benchmarking applications; and the experimental results have shown that eT2FIS possesses a good noise resistance ability, while maintaining a compact and highly interpretable structure.

Chapter 6

Financial Case Study: Stock Price Tracking Using eT2FIS

This chapter investigates the potential of using the eT2FIS model, proposed in Chapter 5, for the use in a real-world modeling application involving the tracking of stock price movement. In the first part of the chapter, an artificially generated stock data is employed to evaluate the suitability of using eT2FIS in financial research. In the second part of the chapter, the change in the underlying dynamics of real-life stock prices over a period of five decades is studied. Subsequently, the effectiveness of the model is investigated in terms of the trading profits achieved when employing eT2FIS in an online trading of the stock. The trading profits, measured in terms of the portfolio end values, are considered in this case study because this measure is more meaningful to an investor.

This chapter is organized as follows. Section 6.1 discusses on the need to perform online tracking of stock prices, which is related to correlations observed in stock returns. Section 6.2 summarizes some associated existing models used in the study of correlations in stock returns, where the inadequacies of the modeling techniques are highlighted. A brief

discussion on the use of neuro-fuzzy systems in financial related topics is also included. This is followed by an evaluation of the suitability of eT2FIS in financial modeling in Section 6.3. Section 6.4 describes the real-life stock price data that is considered in this case study. Section 6.5 presents the experimental results achieved. Lastly, Section 6.6 provides some concluding remarks to this financial case study.

6.1 Motivation

Empirical research in finance has generally identified two classes of regularities in stock prices, i.e. , stock prices exhibit both positive and negative serial correlations in returns. Positive correlations of stock returns are termed *momentum trading*; while negative correlations of returns or price reversals are referred to as *disposition effects*. Acknowledging the presence of these trends, individual traders and institutes have adopted strategies to capitalize on these predictabilities in stock prices for trading [Cohen, Gompers & Vuolteenaho 02][Badrinath & Wahal 02][Griffin, Harris & Topaloglu 03], such that superior trading profits can be realized.

The phenomena of positive and negative correlations in stock returns can be explained by the general responses of stock prices to market news that contains cash-flow relevant information. Events or news that are a priori likely to contain cash-flow relevant information, such as earnings surprises and dividend initiations, are followed by a stock price drift in the same direction as the initial announcement return. This means that good news today is slowly incorporated into the stock prices over a period of time, hence leading to positive autocorrelations in the stock prices during these time horizons. This is also known as *under-reaction* of stock prices to the news [Cutler, Poterba & Summers 91][Chan, Jegadeesh & Lakonishok 97], where current good news has the power to predict positive returns in the future. On the other hand, stock prices tend to become over-priced in longer time horizons

Figure 6.1: Relations between correlations in stock returns and stock price movement: a simplistic illustration.

following a stream of good news, and subsequently earning a lower than expected return. This phenomenon is referred to as *over-reaction* of stock prices [De Bondt & Thaler 85][Cutler, Poterba & Summers 91][Fama & French 92], where consistent patterns of news in the same direction tend to over- or under-valuate the truth value of the stock. Subsequently, researchers from behavioral finance have tried to devise models to account for these empirical evidence [Barberis, Shleifer & Vishny 98][Daniel, Hirshleifer & Subrahmanyam 98][Hong & Stein 99]. (See Section 6.2.1 for a discussion of the models.)

Figure 6.1 illustrates the responses of stock prices to a stream of good news, such that the phenomena of both momentum trading and disposition effects could be possibly explained. At the start of the investment (i.e. time t_1), the investor feels bullish about the performance of the stock. Subsequently, he projects a positive increment in the movement of the stock, which is shown by the red dotted line from time t_1 – t_2 . At time t_2 , some good news regarding the performance of the stock is released (as marked by the green cross at time t_2). This boosts the confidence of the investor, such that his projection for the future performance of the stock follows a faster rate of gain compared to his initial projection. This is shown by the steeper gradient of his projection from time t_2 – t_3 . Upon receiving a second good news at time t_3 (as marked by the green cross), the investor's projection is further adjusted to demonstrate his increased confidence in the performance of the stock. This adjusted

sentiment is shown by the steepest segment of the red dotted line from time t_3 onwards. The red dotted line indicates the general effect of under-reaction and over-reaction of the stock prices, where the effect of single good news is slowly incorporated into the stock price, but an over-valuation of the stock prices occur following a stream of good news. As seen, the truth value of the stock (the blue solid line) has been on the rise throughout the whole observation period. However, the rate of growth of the stock prices is unable to keep up with the projected rate of growth of the investor. Subsequently, price reversal is observed at time t_4 . From this illustration, positive serial correlations of the stock returns (i.e. momentum trading) is observed from time t_1 – t_4 , while negative serial correlations (i.e. disposition effects) occurs thereafter. This figure offers a simplistic illustration of the relationship between correlations in stock returns and the stock price movement. It should be noted that the discussion offered does not include the cumulative trading effects of the whole population of traders. In addition, the volatility of the stock prices is also not included in the discussion.

Existing literature on the study of correlations in stock returns have reached some degree of consensus on the directions of price movements over a short and a long time horizon. Studies have suggested that returns tend to be positively correlated in short to medium term (between 3 to 12 months), and returns tend to be negatively correlated over longer time horizons [De Bondt & Thaler 85][Jegadeesh & Titman 93][Rouwenhorst 98]. However, the actual time periods of a very short, short and long term have remained much debatable. For instance, [Cutler, Poterba & Summers 91] has reported positive correlations for horizons longer than a month; but [Bernard & Thomas 90] and [Barberis, Shleifer & Vishny 98] have found price reversals for periods longer than 9 months and 3–5 years respectively. On the other hand, Grinblatt and Keloharju found that investors are momentum traders for very short horizons of 2–3 days, but their trading patterns reverse for longer time horizons [Grinblatt & Keloharju 01]. Recent findings have suggested that such differences in

views could possibly be results of varying volatilities of different stocks. Arena, Haggard and Yan have found that returns to momentum trading are higher among those stocks with higher idiosyncratic volatility [Arena, Haggard & Yan 08]. In addition, they also reported on a positive time-series relations between momentum returns and aggregate idiosyncratic volatility. On the other hand, [McLean 10] has reported on reversal observations in high idiosyncratic risk stocks. Subsequently, the ability to efficiently track the underlying volatile movement of individual stocks becomes crucial for a further understanding to the phenomena of momentum trading and price reversals¹.

6.2 Financial time-series analysis

This section reviews three popular models proposed to explain both positive and negative correlations observed in stock returns in Section 6.2.1; subsequently, a brief discussion on the applications of neuro-fuzzy systems in financial research is presented in Section 6.2.2.

6.2.1 Behavioral models

This section shall review three behavioral models that try to explain both the short term positive correlations of stock returns, and the long term negative correlations as introduced previously.

Barberis, Shleifer and Vishny's model (BSV model) [Barberis, Shleifer & Vishny 98] - The BSV model was motivated from two important phenomena documented in psychology; namely, *conservatism* and *representativeness heuristic* [Griffin & Tversky 92]. The conservatism bias causes investors to update their priors insufficiently when they first observe new

¹Tracking of stock prices allows one to identify the changes in the stock price movement in an incremental manner. This is unlike prediction of stock prices, where the modeling system is confined to learning within a fixed time period. Subsequently, the system is unable to give a good forecast of the stock when the underlying environment has changed from the time when the system was learning.

information about a stock. This, subsequently, leads to the initial under-reaction observed. At the same time, investors suffer from a representativeness bias, which leads to a delayed over-reaction. This means that a person will first evaluate the probability of an uncertain event by the degree of similarities between the event and his knowledge of related events, so as to assess the novelty of the uncertain event. An important manifestation of such an evaluation is that people think they see patterns in truly random sequences, thus offering an explanation to over-reaction.

Mathematically, the BSV model uses two matrices to represent the two states of the stock price movement, which are separately driven by the two underlying psychological states (i.e. conservatism and representativeness heuristic) respectively. Subsequently, a transition matrix governs the transition between the two states. This means that at anytime, an investor is either in the conservatism state (i.e. trading in an under-reaction manner) or in the representativeness state (i.e. trading in an over-reaction manner).

Daniel, Hirshleifer and Subrahmanyam's model (DHS model) [Daniel, Hirshleifer & Subrahmanyam 98] - Similar to the BSV model, the DHS model was inspired from two well-known psychological biases; namely, *over-confidence* and *biased self-attribution*. In financial trading, investors generally perform trading based on their personal abilities to process and analyze information. If an investor over-estimates his ability to identify the significance of existing information that others neglect, he will under-estimate his forecast errors. Thus, an over-confident investor is one who over-estimates the precision of his private information analysis, over-weighing the information publicly received by all. Subsequently, this leads to an over-reaction of the stock prices. On the other hand, when more public information arrives, this inefficient deviation of the stock price is partially corrected. On the long term, the stock prices will move close to the truth value of the stock, resulting in an under-reaction.

In summary, the DHS model hypothesized that traders generally under-react to public

information, but over-react to private information.

Hong and Stein's model (HS model) [Hong & Stein 99] - Unlike the BSV and the DHS models where each individual investor's trading pattern is investigated, the HS model explains the phenomena of positive and negative correlations by inspecting the interactions between two groups of investors. They are referred to as the news watchers and momentum traders. Several guidelines are imposed on the trading strategies of the two groups: (1) news watchers' trading strategy is based *only* on private information regarding a stock; (2) momentum traders' trading strategy is based *only* on past stock price changes, such that their forecasts of the stock prices must be uni-variate functions of historical prices; and (3) private information diffuses gradually across the group of news watchers.

Beginning from the news watchers, they perform trading using only private information (i.e. guideline (1)). When new information is introduced, stock prices will then adjust slowly, which helps to explain the under-reaction observed. This is a direct consequent of guideline (3) imposed on the group of news watchers. Subsequently, the group of momentum traders attempt to profit from this under-reaction caused by news watchers. This is possible because momentum traders based their forecasts solely on past stock price changes (i.e. guideline (2)). The initial reaction to this trading strategy is an acceleration of the stock prices in the direction of the fundamentals. However, with continuation of this strategy, the eventual stock prices will exceed the truth value of the stock, leading to an over-reaction of stock prices.

From the discussion of the three models above, correlations observed in stock returns are explained via trading patterns of individual traders and/or interactions between groups of traders. While the models attempt to explain the existence of such correlations through the perspective of the players in the market, a study of the movement in stock prices might offer some alternative insights to these empirical findings. In particular, the degree of volatility

of individual stocks might shed some light to the lengths of positive and negative sequences of the stocks [Arena, Haggard & Yan 08][McLean 10] (The lengths of the correlations have not been explicitly discussed or explained in the three models). Subsequently, the ability to efficiently track the price movement of stocks forms the focus of this case study.

6.2.2 Neuro-fuzzy systems in financial research

Neuro-fuzzy systems have been actively employed in various areas of financial research, especially in the forecasting and prediction of stocks and indexes. For instance, Ang and Quek proposed a forecasting of daily stock price differences rather than the daily stock prices, so as to prevent a loss of information from directly setting the forecasted stock prices as input to the trading system [Ang & Quek 06b]. On the other hand, a volatility-based straddle trading model has been proposed in [Tung & Quek 11], and the viability of the model has been tested on the Hong Kong Index options. In addition, other examples of neuro-fuzzy systems in financial research include the assessment of loan credits [Malhotra & Malhotra 02], warning systems for predicting bank failures [Tung, Quek & Cheng 04], among others.

In this case study, the suitability of using eT2FIS in financial research is first evaluated in Section 6.3. The online tracking of real-life volatile financial stock prices is then investigated in Sections 6.4 and 6.5. Since empirical studies have shown that financial volatility is time-varying and non-stationary in nature [Mele & Fornari 00], the potential of using the eT2FIS model to perform online tracking of the stock price movement is studied. This is because eT2FIS is an evolving neuro-fuzzy system, which has the ability to incrementally detect changes in the underlying dynamics of a learning environment. In addition, being a Type-2 Mamdani model, eT2FIS has the potential to handle the effects of noise, while providing valuable insights to the application problem.

Figure 6.2: A particular generation of the artificial stock data.

6.3 Experiment: Artificially generated stock price data

This section evaluates the suitability of using the eT2FIS model for financial research using an artificially generated stock data. The effectiveness of eT2FIS is evaluated in terms of the mean squared error (MSE) between the computed output of eT2FIS and the desired output of the test data. The performance of the eT2FIS model is compared against the following models; namely, the random walk model; feed-forward neural network FFNN (configured with 10 input neurons, 30 hidden neurons, and 1 output neuron); radial basis function network RBFN (with a default spread of 0.1); ANFIS [Jang 93]; EFuNN [Kasabov 01]; and DENFIS [Kasabov & Song 02].

The artificial stock price data z is generated as described in [Moody et. al. 98] using (6.1) and (6.2):

$$\begin{aligned} y(t) &= y(t-1) + \beta(t-1) + k\epsilon(t) \\ \beta(t) &= \alpha\beta(t-1) + v(t) \end{aligned} \tag{6.1}$$

$$z(t) = \exp\left(\frac{y(t)}{\max y - \min y}\right) \tag{6.2}$$

where $\alpha = 0.9$, $k = 3$, and initial conditions $y(0) = 0$ and $\beta(0) = 0$. Both ϵ and v

Table 6.1: Average performances for the forecasting of the price difference on the artificial stock price data.

Model	# Rules	Recall MSE ($\times 10^{-6}$)	Test MSE ($\times 10^{-6}$)
Random walk	-	7.81 ± 5.53	9.68 ± 7.17
FFNN	-	14994 ± 3525	31981 ± 28795
RBFN	-	7.81 ± 5.52	9.69 ± 7.17
ANFIS	37.9 ± 47.7	4.20 ± 4.07	13.06 ± 11.56
DENFIS	317.0 ± 90.1	6.64 ± 8.92	8.76 ± 5.03
EFuNN	2138.2 ± 139.5	2.48 ± 2.10	14.84 ± 10.15
eT2FIS	89.7 ± 60.1	7.18 ± 1.86	8.70 ± 4.39

are normal random deviates with zero mean and unit variance. Subsequently, 10000 data samples are generated, of which the first 3000 are used for training and the rest for testing. Figure 6.2 illustrates a particular generation of the stock price data z . For this suitability study, the experimental setup of [Ang & Quek 06b] is adopted, where the previous nine days daily price differences, $\Delta z(t-1)$ to $\Delta z(t-9)$, serve as inputs to predict the current price difference, $\Delta z(t)$. A comparison of the average performances over 20 generations of the stock data are then carried out between the benchmarking models.

Table 6.1 shows the average performances of the models for the forecasting of the price difference on the artificially generated stock data. Both the average performances on the recall and testing tasks are listed in the table, together with the average number of fuzzy rules identified. Although both ANFIS and EFuNN demonstrated a good recalling result, the testing performances of the models suffered, indicating an over-trained phenomenon. On the other hand, both the average recall and testing performances of neuro-fuzzy systems such as DENFIS and eT2FIS yielded better results when compared with the random walk approach and RBFN. In addition, the proposed eT2FIS model identified much lesser fuzzy rules, second only to ANFIS which is a precise fuzzy model. This result demonstrated that both DENFIS and eT2FIS are superior choices compared to the rest of the benchmarking

Table 6.2: Average performances for the forecasting of the price difference on the artificial stock price data with injected uniform noise.

Model	# Rules	Test MSE ($\times 10^{-6}$)
DENFIS	319.0 ± 90.6	10.2 ± 6.18
eT2FIS	61.0 ± 38.3	8.98 ± 4.79

models for the modeling of financial data.

In the second part of the suitability study, uniform noise in the range of $[-0.005, 0.005]$ is injected into the desired output of the training dataset. Both the DENFIS and the eT2FIS models are then trained on the noisy data, and the trained models are used to perform forecasting on the unseen test data. The average performances of the two models are subsequently listed in Table 6.2. While the average performance of the proposed eT2FIS model has dropped marginally (by 3.22 % from 8.70×10^{-6} to 8.98×10^{-6}), the average performance of DENFIS suffered a significant 16.4 % deterioration. Overall, eT2FIS achieves an 11.9 % reduction in the testing MSE and a savings of 80.9 % in the number of fuzzy rules used when compared to DENFIS for this noise injected data. This result demonstrated the noise resistance ability of the proposed model when modeling in an uncertain environment, indicating on its suitability in modeling real-life volatile financial data.

6.4 Data collection and experimental setup

The set of real-life stock price data employed in this case study is described hereafter. The daily closing stock prices of Alcoa Incorporation were collected from the Yahoo! Finance website over a period of five decades from 2-th January 1962 to 30-th July 2010. A total of 12,229 data points was collected. Figure 6.3 illustrates the stock price movement of the company, such that the values cover a range from 0.49 to 43.71. The daily price differences of the stock are also included in the figure. As seen from the change in fluctuations of the

Figure 6.3: Daily closing stock prices of Alcoa Incorporation from 2-nd January 1962 – 30-th July 2010 (together with the daily stock price differences).

price differences, the volatility of the stock prices increase gradually from the 1980s, and subsequently, the daily differences of the stock price changes drastically during the decade from 2000–2009.

In this case study, each data sample uses the previous two days' stock prices as inputs. The eT2FIS model is then employed to track the stock price movement in an online mode. Subsequently, the model is used to predict the stock price at a later time (i.e. for a single day ahead, and for a week ahead). Let $y(t)$ represents the stock price at day t . The performance of eT2FIS is evaluated by the online trading performance of the model at day $t + \lambda$ for $\lambda = 1$ and 5 days respectively.

A realistic yet simple computation of the profits and losses of a trading model from [Moody et. al. 98] is adopted in this case study. Two commonly used trading decision models based on the technical analysis approach are compared; namely, with price forecast and without price forecast. The action of a trading system is assumed to be either in a short or long position at any time instance t as represented by $F(t)$ where $F = -1$ or 1 respectively. A period of continuous -1 or 1 indicates a neutral position. The trading profit of the trading

system is subsequently modeled by the portfolio end values R using a multiplicative return based on [Moody et. al. 98] calculated using (6.3):

$$R(t) = [1 + F(t-1)r(t)][1 - \delta|F(t) - F(t-1)|] \quad (6.3)$$

where $r(t) = y(t)/y(t-1) - 1$ and δ is a fraction (of the transacted price value) that represents the transaction cost. The transaction cost is set at a default value of 0.1 %. Here, the buy and sell decisions F are computed based on a modification of the popular Moving Average Convergence Divergence (MACD) oscillator developed by Gerald Appel [Plummer & Ridley 03]. MACD uses the crossovers of a *Fast* signal as given in (6.4) and a *Slow* signal as given in (6.5) to indicate a buy or sell decision:

$$Fast(t) = EMA_{\tau_{short}}^y(t) - EMA_{\tau_{long}}^y(t) \quad (6.4)$$

$$Slow(t) = EMA_{\tau_{slow}}^{Fast}(t) \quad (6.5)$$

where the Exponential Moving Average (EMA) of a price series is given as in (6.6):

$$EMA_{\tau}^z(t) = Kz(t) + (1 - K)EMA_{\tau}^z(t-1) \quad (6.6)$$

such that $K = 2/(\tau + 1)$ and τ is the number of time instants of the moving average. For this case study, the values are set as follows: $\tau_{short} = 16$, $\tau_{long} = 26$, and $\tau_{slow} = 15$. Instead of using the crossover signals to generate the trading decision, a simpler moving average trading rule following [Ang & Quek 06b] is used as in (6.7):

$$F(t) = \text{sign}(Slow(t)) \quad (6.7)$$

Subsequently, the trading model without forecast is computed using (6.4)–(6.7).

Figure 6.4: Trading decision model with forecasting.

On the other hand, the computation of the current trading signal $F(t)$ using both the forecasted value $\hat{y}(t+1)$ and the current value $y(t)$ is given as in (6.8)–(6.11) [Ang & Quek 06b]:

$$Fast'(t+1) = EMA_{\tau_{short}}^{\hat{y}}(t+1) - EMA_{\tau_{long}}^{\hat{y}}(t+1) . \quad (6.8)$$

$$Slow'(t+1) = EMA_{\tau_{slow}}^{Fast'}(t+1) . \quad (6.9)$$

$$EMA_{\tau}^{\hat{z}}(t+1) = K\hat{z}(t+1) + (1-K)EMA_{\tau}^{\hat{z}}(t) . \quad (6.10)$$

$$F'(t) = \text{sign} (Slow'(t+1)) . \quad (6.11)$$

6.5 Experimental results and analysis

This section presents the online tracking and trading performances of the eT2FIS model for the stock price of Alcoa Incorporation described in Section 6.4. The effectiveness of eT2FIS is evaluated in terms of trading profits achieved, calculated using the portfolio end

Figure 6.5: Semantic fuzzy labels identified by eT2FIS in the input spaces for $\lambda = 1$ day.

values as introduced in the previous section. Figure 6.4 illustrates the online trading model using eT2FIS as the forecasting system. Instead of statistical measures, the trading profits of the trading model offer a more meaningful measure for investors. The performance of the proposed eT2FIS model is subsequently benchmarked against the DENFIS [Kasabov & Song 02] model.

Figure 6.5 illustrates the fuzzy clusters identified by the eT2FIS model in the input spaces $y(t-1)$ and $y(t)$ for the online tracking of stock prices when $\lambda = 1$ day. A total of 7, 6 and 4 fuzzy clusters are identified for the input-output spaces respectively. As seen from Figure 6.5, the identified fuzzy clusters are highly-ordered and distinct, thus providing a high level of interpretability to the proposed model. Subsequently, semantic fuzzy labels can be associated with the identified fuzzy clusters. Using the semantic fuzzy labels in Figure 6.5, eT2FIS identified seventeen fuzzy rules at the end of the tracking experiment. Table 6.3 lists the rules. As seen from the table, there are two groups of fuzzy rules describing two different phenomena. For rules 1 to 13, the previous two days' stock prices and the predicted stock price are in the same direction, i.e., if the previous two days exhibit positive (negative) growth, then the predicted price for the next day is also positive (negative). This phenomenon is commonly known as *trend-chasing*, which is related to under-reaction of stock prices and momentum trading [Hong & Stein 99]. On the other

Figure 6.6: Online tracking performances of eT2FIS for a day ahead forecast of the stock.

hand, rules 14 to 17 demonstrate the phenomenon of *trend-reversing*, which is related to over-reaction of stock prices and price reversals. This means that the previous two days' stock prices (i.e. inputs) are in the opposite direction as the predicted price movement (i.e. output). As hypothesized by [Barberis, Shleifer & Vishny 98], the market is more likely to be driven by the under-reaction, rather than the over-reaction, of stock prices. This is also supported by the eT2FIS model where majority of the identified rules describe the trend-chasing phenomenon. In addition, the identified rulebase in eT2FIS illustrates that at anytime, the market can be driven by a mixture of both under- and over-reaction. This is unlike the behavioral models introduced in Section 6.2.1, where only one driving force is possible at each instant of time.

Subsequently, Figure 6.6 illustrates the online tracking performances of the eT2FIS model for $\lambda = 1$ day. As seen, there is an excellent match between the one day forecast by the proposed model and the actual stock price movement. Despite the more volatile period from 2000–2009 (see tracking result in the blown-up of that period), the tracking performances of eT2FIS is still excellent, where the peaks and valleys of the stock price movement are all identified. Overall, the proposed model achieved a MSE value of 0.166 and an R value of

Table 6.3: Mamdani-type fuzzy rules identified in eT2FIS for $\lambda = 1$ day.

Description	Rule	$y(t-1)$	$y(t)$	$y(t+1)$
Trend-chasing:	1	VH	VH	VH
	2	VH	H	H
	3	H	VH	H
	4	H	H	VH
	5	M	H	H
	6	M	MH	H
	7	M	ML	L
	8	ML	L	L
	9	ML	VL	VL
	10	L	ML	L
	11	L	VL	VL
	12	VL	L	L
	13	VL	VL	L
Trend-reversing:	14	H	MH	L
	15	MH	MH	L
	16	ML	ML	H
	17	L	L	H

L–Low; M–Medium; H–High; V–Very

Figure 6.7: Stock price tracking performances.

Table 6.4: Average performances for the stock price tracking task.

Model	# Rules	Average R	Average MSE
DENFIS	14.0	0.999	0.286
eT2FIS	15.5	0.999	0.277

0.999 for the whole tracking duration for this one day ahead forecasting. This result is a highly encouraging indication to the suitability and potential of employing eT2FIS for the online trading task.

The online trading performances of eT2FIS are benchmarked against another evolving neuro-fuzzy system, DENFIS. As an illustration, only the performances after 2008 are shown. As seen from Figure 6.7, the online tracking performances of both DENFIS and eT2FIS are excellent, where the peaks and the valleys of the desired output are identified by the two benchmarking models. In addition, the average statistical performances, together with the average number of identified fuzzy rules, of the two models are also highly comparable as seen from Table 6.4. This result shows that the statistical measures might not be indicative of the trading performances of the two models.

The DENFIS and the proposed eT2FIS models are subsequently evaluated using the trading profits achieved during this period of interest. The portfolio end values of the two models are

Figure 6.8: Online trading performances.

Table 6.5: Performances for the online stock trading task.

Model	$R(t)$			
	$\lambda = 1$ day	(% Gains)	$\lambda = 5$ days	(% Gains)
Model without forecast	1.783		1.779	
Model with ideal forecast	1.933	(8.41)	2.077	(16.8)
DENFIS	1.920	(7.68)	1.916	(7.70)
eT2FIS	1.930	(8.24)	2.032	(14.2)

shown in Figure 6.8. Also included in the figure are the portfolio end values for the trading decision models without forecast and with ideal forecast. The trading signals for stock trading without forecast is computed using (6.4)–(6.7); and the trading signals for DENFIS and eT2FIS are computed using (6.8)–(6.11) where $\hat{y}(t+1)$ is the forecasted output from DENFIS and eT2FIS respectively; while the trading signals for stock trading with ideal forecast is computed using (6.8)–(6.11) where the forecasted value $\hat{y}(t+1)$ is the actual price value of the next time instance $y(t+1)$. All portfolio end values are computed using (6.3). Starting with a portfolio value of $R(0) = 1$, the trading model without forecast yielded the least profits with a portfolio end value of $R(t) = 1.783$ for $\lambda = 1$ day and $R(t) = 1.779$ for $\lambda = 5$ days. The performance for this online stock trading task is subsequently listed in Table 6.5. On the other hand, the trading model with ideal forecast earned the highest profits for both the forecasting horizons, with a portfolio end value of $R(t) = 1.933$ for $\lambda = 1$ day (i.e. a maximum gain of 8.41%) and $R(t) = 2.077$ for $\lambda = 5$ days (i.e. a maximum gain of 16.8 %). Both the earnings of the DENFIS and the eT2FIS models fall between these two extremes. This can be seen from Figure 6.8, where (b) illustrates a zoom in of the entire duration of interest as shown in (a). Although the profits for DENFIS and eT2FIS are very comparable when $\lambda = 1$ day (with DENFIS achieving a gain of 7.68 % and eT2FIS achieving a gain of 8.24 %), the profits of the proposed eT2FIS model is observed to be much higher compared to that of DENFIS when $\lambda = 5$ days. This can be seen from Figure 6.8B.(b). As seen from Table 6.5, the gain achieved by eT2FIS is almost double of that achieved by DENFIS when $\lambda = 5$ days. Overall, the portfolio end values of eT2FIS are $R(t) = 1.930$ (i.e. a gain of 8.24%) and $R(t) = 2.032$ (i.e. a gain of 14.2 %) for $\lambda = 1$ and 5 days respectively. This result shows that despite showing similar statistical performances in Table 6.4, the proposed model achieves a much more significant amount of profits when compared to that achieved by DENFIS.

6.6 Summary

This chapter demonstrates the suitability and effectiveness of employing the eT2FIS model proposed in Chapter 5 for the modeling of volatile financial stock data. Experimental results on the forecasting of price differences of an artificially generated stock data showed that neuro-fuzzy models such as eT2FIS and DENFIS yielded superior generalization performances when compared with the well-established random walk approach. Subsequently, experimental results based on real-life financial stock price data are presented. Since the tracking of stock price movement is directly related to the study of correlations in the stock prices, eT2FIS is employed to perform online tracking of the stock data. Results showed that eT2FIS is able to provide intuitive insights to the raw numerical data by providing semantic fuzzy labels and fuzzy rules to explain the stock price movement, i.e., the presence of a combination of two driving forces (namely, trend-chasing and trend-reversing) behind the movement of the stock at each time instance. The generalization ability of eT2FIS is then compared against DENFIS. Although statistical results have shown that both the models are comparable under varying tracking time horizons, the trading profits achieved using the proposed eT2FIS model significantly exceeds that achieved by DENFIS. In addition, it is more meaningful for investors when the trading profits achieved are employed as a benchmarking measure.

The potential of online tracking and trading for volatile financial stock prices using the eT2FIS model discussed in this chapter is encouraging as it demonstrates the feasible application of eT2FIS in the area of financial research.

Chapter 7

Conclusions and Future Works

This chapter provides some concluding discussions on the topics presented in the thesis. A summary of the contributions will be presented in Section 7.1. This is followed by a discussion on the directions for future works in Section 7.2.

7.1 Conclusions

To design a neuro-fuzzy system, a key area to consider is to derive the knowledge base of the system. This can be further viewed as two problems; namely, the fuzzy partitioning of the input and output spaces (such that the antecedent and consequent segments of the fuzzy rules are defined); and the rule generation procedure of the system (such that consistent and meaningful rules are created with respect to the application problem). With regards to this, some former neuro-fuzzy systems are manually crafted by human experts, where the tasks of fuzzy partitioning and rules generation are pre-determined by the human experts. Unfortunately, this is not always feasible because human knowledge is not always readily available, especially for complex application problems where complete knowledge from the

human experts is difficult to formulate and verbalize. As such, a data-driven approach is adopted to design a neuro-fuzzy system in this thesis.

In general, there are two approaches to design a neuro-fuzzy system from numerical data. They are performed via:

1. offline learning, where the block of training data is collected and made available before the design phase of the neuro-fuzzy system; and
2. online learning, where the numerical data arrives in a sequential manner such that learning in the neuro-fuzzy system is performed on a single data tuple at each time instant.

Subsequently, the works in this thesis are presented via two tracks:

1. batch neuro-fuzzy systems, where offline learning from data is employed; and
2. evolving neuro-fuzzy systems, where online learning from data is employed.

Figure 7.1 illustrates a summarized view of the research works achieved in this thesis.

In the first part of the thesis, the characteristics and deficiencies of existing batch neuro-fuzzy systems are described. In general, batch neuro-fuzzy systems are also known as self-organizing neuro-fuzzy systems because of their automated design from numerical data. Existing self-organizing neuro-fuzzy systems suffer from one or more of the following major drawbacks:

1. an inconsistent rulebase;
2. the need for prior knowledge such as the number of clusters to be computed;
3. heuristically designed knowledge acquisition methodologies; and

Figure 7.1: A summarized view of research contributions in this thesis.

4. the stability-plasticity trade-off of the system.

While problem (1) is related to the fuzzy rules generation mechanism adopted, problems (2)–(4) are directly related to the techniques employed for the fuzzy partitioning.

To address problems (2)–(4), a novel fuzzy partitioning technique named Categorical Learning Induced Partitioning (CLIP) is presented in Section 3.2.1. The CLIP technique is motivated from the behavioral category learning process exhibited in humans, which addresses problem (3). In addition, CLIP provides a tailored approach to fuzzy partitioning of the input-output spaces such that new clusters in each input-output space can be automatically identified when a prominent distinction is observed. This effectively addresses problem (2). Lastly, refinements are made to existing clusters in an input-output space when a new cluster is created such that a balance is maintained between the new knowledge from incoming training data and the stored knowledge in the system, hence addressing problem (4). Subsequently, the issue of an inconsistent rulebase (i.e. problem (1)) is addressed using the rulebase identification process presented in Section 3.2.2, where only the most significant rule in a set of inconsistent rules is retained such that the neuro-fuzzy system gives a most aptly description of the application problem. Following that, the proposed CLIP technique in Section 3.2.1 and the proposed rulebase identification process in Section 3.2.2 are employed in a novel self-organizing neuro-fuzzy system named Self-adaptive Fuzzy Inference Network (SaFIN) in Chapter 3. In Section 3.3, SaFIN is compared against existing models in three standard benchmarking experiments, and excellent performances have been achieved to demonstrate its potential as a self-organizing neuro-fuzzy system.

A real-world application of SaFIN is presented on a transportation case study in Chapter 4. The effectiveness of the model is evaluated in terms of the mean squared error and the Pearson correlation coefficient between the computed output of SaFIN and the actual traffic flow density of the highway, such that the suitability of employing SaFIN for the

modeling of highway traffic flow density is investigated. This is important because a good and reliable traffic flow prediction on the highway allows for timely adjustments to be made to existing traffic planning policies, which eventually affects the level of service experienced by commuters. Experimental results have shown that using SaFIN for the highway traffic modeling provides intuitive insights to the problem such that changes in the demands on the highway usage throughout the day can be effectively accounted for. In addition, SaFIN achieves superior modeling performance compared to existing models, giving a more consistent and stable performance even when the prediction horizons are extended to longer periods.

In the second part of the thesis, the characteristics and deficiencies of existing evolving neuro-fuzzy systems are described. Existing evolving neuro-fuzzy systems suffer from one or more of the following major drawbacks:

1. susceptibility to noisy data;
2. an ever-expanding knowledge structure; and
3. a lack of interpretability of the system.

These drawbacks are the consequences of online learning in an uncertain (hence leading to problem (1)) and time-varying (hence leading to problems (2)–(3)) environment.

To address the drawbacks, a novel evolving neuro-fuzzy system named evolving Type-2 neural Fuzzy Inference System (eT2FIS) is proposed in Chapter 5. The eT2FIS model synergizes the concepts of evolving neuro-fuzzy systems and Type-2 Mamdani models into a common framework, which effectively addresses problem (1) by using a second order model to handle the noise present. In addition, the eT2FIS model is also one of the first few evolving Type-2 Mamdani models in the literature. With regards to problem (2), an addition-removal process of rules is proposed in eT2FIS (see Section 5.2.2 for addition and

see Section 5.2.4 for removal), where a new rule is created only if the incoming training data is novel with respect to existing stored knowledge in the system, and an out-dated rule is pruned when it no longer participates in the modeling of the current learning environment. This means that only essential rules with respect to the current environment are kept in the system. Lastly, problem (3) is addressed by merging highly over-lapping fuzzy labels in the input-output spaces of eT2FIS as explained in Section 5.2.3. In Section 5.3, eT2FIS is compared against existing models in two standard benchmarking experiments, and excellent performances have been achieved to demonstrate its potential as an evolving neuro-fuzzy system.

A real-world application of eT2FIS is presented on a financial case study in Chapter 6. The feasibility of employing the eT2FIS model in financial research is first studied using an artificially generated stock data, and the excellent performances of the model in both noise-free and noisy learning environments indicate its potential in financial modeling. The proposed model is next employed to track the movement of a real-life volatile financial stock data in an online manner, where the effectiveness of the model is evaluated in terms of the trading profits achieved when engaging eT2FIS in an online trading of the stock. Experimental results have shown that using eT2FIS for the online tracking of stock price movement provides intuitive insights to the stock such that the driving forces behind the movement of the stock price at a time instant can be effectively accounted for. This essentially provides investors with an informative platform for understanding the current trends in the stock movement. In addition, when benchmarked against existing models, eT2FIS achieves superior trading profits for different forecasting horizons. By evaluating the benchmarking models using the trading profits achieved, it provides investors with a more meaningful assessment of the models.

7.2 Future works

This section will present two main areas of recommendations for future research directions; namely, in the design aspects of neuro-fuzzy systems, and the application areas of the proposed models.

7.2.1 Recommendations on designs of neuro-fuzzy systems

Feature analysis involves selecting a representative reduced subset of input features from the given numerical data, which will then be used in a modeling system [Guyon & Elisseeff 03][Lee & Verleysen 09]. There are two main approaches to perform feature analysis in the literature [Blum & Langley 97]; namely, (1) filter approach, where the features are selected independently from the model; and (2) wrapper approach, where the features are selected using the modeling system. The filter approach generally uses statistical information extracted from the numerical data [Fu & Wang 03][Chow & Huang 05] or semantics-preserving information contained in the data [Shen & Chouchoulas 02][Jensen & Shen 04][Lim & Jin 06][Lespinaats et. al. 07]; while the wrapper approach yields better performances at the expense of an increased computational complexity [Kohavi & John 97][Inza et. al. 02][Wang, Zhou & Chu 08].

In retrospect, existing neuro-fuzzy systems can be broadly classified into two classes; namely, linguistic fuzzy models, which are focused on delivering a good level of interpretability to the structures of the systems; and precise fuzzy models, which are focused on achieving high accuracies in the modeling tasks. While Mamdani models form the main focus in the former class; the latter class comprises of mainly TSK models. In general, having a good fuzzy rulebase interpretability and a high modeling accuracy are two contradictory requirements in the design of a neuro-fuzzy system [Ang & Quek 05].

Figure 7.2: Flowchart of performing feature analysis as a pre-processing stage *prior* to the design of a neuro-fuzzy system.

With its success in data-mining applications, feature analysis techniques are often performed either implicitly or explicitly as a pre-processing phase prior to the design stage of a neuro-fuzzy system. This means that only the filter approach for feature analysis is applicable in such cases. Figure 7.2 illustrates a flowchart of designing a neuro-fuzzy system, where feature analysis is performed as a pre-processing stage. Some examples of such works are [Cho, Chow & Leung 99] , [Lim, Tian & Mulhem 03] , [Tan, Quek & Ng 05] and [Tung & Quek 05a]. The main advantage for performing this pre-processing stage is an improvement to the modeling accuracy through the removal of redundant/derogative features and the identification of relevant features for the underlying application problem [Lee & Verleysen 09]. Nevertheless, the decoupled stages (i.e. the pre-processing stage of feature analysis, and the design stage of a neuro-fuzzy system) do not consider the close relationship between feature analysis and system design [Pal 99], i.e. , to identify features that are important for the application problem *and* are relevant to the output of the neuro-fuzzy system. In other words, the use of wrapper approach for feature analysis is not prevailing in existing neuro-fuzzy systems. This is despite the wrapper approach yielding better performances [Kohavi & John 97][Inza et. al. 02][Wang, Zhou & Chu 08].

A potential future research direction is thus to incorporate feature analysis into the design

Figure 7.3: Proposed flowchart for performing feature analysis *integrated* into the design of a neuro-fuzzy system.

stage of a neuro-fuzzy system. Figure 7.3 shows a proposed flowchart of designing a neuro-fuzzy system, where feature analysis is integrated into the design. As seen, the original set of numerical data is first used to design a neuro-fuzzy system. This is followed by performing feature analysis using both the knowledge in the data and the neuro-fuzzy system. Finally, the neuro-fuzzy system is modified with an updated version of the numerical data, where only relevant features are present. By proposing this integrated design, it aims to provide a regulated balance between the interpretability and accuracy trade-off in a neuro-fuzzy system. The objective is two-folds; namely, (1) to improve the interpretability of the system by identifying features relevant to its computational structure; and (2) to improve the accuracy of the system by identifying features relevant to the application problem [TUNG, Quek & Guan 11]. In [Ang & Quek 06a], a wrapper-based feature selection method that employs the mutual information maximization scheme is proposed. The presented methodology is, however, applicable only for classification tasks. Recently, a novel feature analysis integrated neuro-fuzzy system has been proposed in [TUNG, Quek & Guan 11], and preliminary results have demonstrated some encouraging performances in regression modeling. Nevertheless, the feature analysis technique has been coupled with POPFNN [Quek & Zhou 96] where redundancy is huge, since the neuro-fuzzy system initializes the rulebase with the maximum combinatoric number of fuzzy rules. As pointed out in the paper, further inves-

Figure 7.4: Proposed flowchart for performing online feature analysis *integrated* into the design of an evolving neuro-fuzzy system.

tigations are necessary with regards to addressing the curse-of-dimensionality issue of the system. In addition, the system has not been benchmarked against other feature selection techniques such as the Mutual Information Feature Selection (MIFS) [Battiti 94].

Investigating further into the notion of integrating the design of a neuro-fuzzy system with feature analysis, the proposed design flowchart in Figure 7.3 is applicable only to neuro-fuzzy systems adopting batch learning. This is because the block of training data is assumed to be available prior to the design of the neuro-fuzzy system (and also before feature analysis is performed). On the other hand, for a dynamically changing learning environment, input features that are relevant in a particular time instant are highly likely to become irrelevant in the next time instant, only to be relevant again much later. Hence, it is desirable to explore ways to address such time-varying relevance of features. This means that for a *fixed* vector length of features, the components within the vector can change after performing feature analysis in each time instant. This helps to improve the modeling accuracies of evolving neuro-fuzzy systems by identifying relevant features incrementally, such that only relevant features for that time instant participate in the modeling of the learning environment. Another desirable trait is the handling of new features that are previously unknown, but

are presented to the evolving neuro-fuzzy system mid-way through the learning process. This means that for a *varying* vector length of features, the components within the vector can change after performing feature analysis in each time instant. Hence, the case of a fixed vector length of features is a special case of this. Figure 7.4 shows a proposed generic flowchart for performing feature analysis incrementally in an evolving neuro-fuzzy system, where the data tuple is presented to the system in a sequential manner. There are currently some preliminary works on online feature selection. For example, Perkins and Theiler proposed an online feature selection algorithm using grafting [Perkins & Theiler 03], and they work on the assumption that the size of the training data is fixed but the length of the features increases over time. On the other hand, Wong and Cho proposed a local experts organization model to deal with the inconsistent length of features when performing face emotion recognition [Wong & Cho 09]. Nevertheless, there are currently no existing works on online feature selection in evolving neuro-fuzzy systems.

Summarizing, the following traits are desirable for a feature analysis integrated neuro-fuzzy system:

1. The features selected should be relevant to both the application problem *and* the neuro-fuzzy system, i.e. , both the environment *and* the learning system.
2. The neuro-fuzzy system should be able to handle the time-varying relevance of each individual input feature, i.e. , the system is able to incrementally select relevant features in a *fixed* vector length of features.
3. Since it is possible for the vector length of features to differ at each time instant, the neuro-fuzzy system should be able to effectively handle incoming and outgoing of features, i.e. , the system is able to incrementally select relevant features in a *varying* vector length of features.

7.2.2 Recommendations on application areas

The potentials of the proposed SaFIN model and the proposed eT2FIS model have been demonstrated respectively in the modeling of highway traffic flow density and the tracking of stock price movement. The experimental results have shown great potentials in the proposed models for both offline and online learning and modeling of data, and thus encourage further exploitations in other application areas. Here, two prospective areas of applications are discussed; namely, in assistive biomedical research, and for hydrology research. Nevertheless, it should be noted that the list of possible application areas is non-exhaustive.

Assistive biomedical research

The amplified signals obtained using electrodes placed on the scalp of a patient offer a tremendous avenue for investigating the electromagnetic states of the human brain. These signals, also known as electroencephalogram (EEG) [Niedermeyer & Da Silva 04], are often studied in the works of brain-computer interface (BCI) [Li & Guan 06], which provide an alternative communications channel between a human and the environment. This is because communications can be performed independently from the usual peripheral nerves and muscles, making the technology especially useful for patients with severe neuromuscular disorders. Despite the great potential in this technology, the BCI community has repeatedly reported the presence of noise disturbance present in the data when using EEG as a source of neural signals. This, subsequently, requires an urgent research on ways to effectively handle such noise present.

In general, there are three noise sources in the EEG signals. They are listed below, together with some common techniques proposed for noise handling [Knight 03]:

1. *Psychological aspects.* Varying states of a subject's awareness and motivation, and varying behavioral, cognitive and psycho-emotional factors are all contributing ele-

ments to the non-stationaries present in session-to-session usage of the EEG signals. To overcome this inter-session variability in the brain dynamics, a regular update of a BCI system is practiced. However, the temporal scale for such updates has been largely subjective, where the choice of how often an update is performed remains more of an art than a science. For instance, Shenoy et. al. suggested a session-to-session calibration of the BCI [Shenoy et. al. 06]; while Vidaurre et. al. proposed several updates within a single trial session [Vidaurre et. al. 06].

2. *Physical aspects.* Large amount of EEG data is often contaminated by artifacts such as eye blinks, eye movements, heart beat, pulse signals, breathing, or other muscular activities. These artifacts affect the quality of the EEG signals, making the analysis of the data difficult. The easiest way around this problem is to discard the affected segment of data when an artifact is detected, but this leads to high wastage of valuable information. In addition, detection of artifacts can be cumbersome when physicians are required to manually mark out the affected segments. Computationally, methods such as Principle Component Analysis (PCA) [Lee & Choi 03][De Cheveigne & Simon 07] and Independent Component Analysis (ICA) [Vigario et. al. 00][Iriarte et. al. 03] have been proposed. However, such techniques have their own sets of issues, such as over-simplified assumptions, and intractable independence measures.
3. *Environmental aspects.* Other possible sources of contamination come from the environment where noise is introduced into the captured data during collection through the power line. This is usually resolved by applying a bandpass filter during data collection.

There have been many existing works of applying fuzzy systems to BCI applications [Pala-niappan 02][Lin et. al. 05][Guler & Ubeyli 05][Lotte 06][Coyle, Prasad & McGinnity 09], mainly due to its potential in minimizing the effects of uncertainties in the data. How-

ever, most of these works utilize Type-1 models, with only a handful using Type-2 models [Herman, Prasad & McGinnity 05][Herman, Prasad & McGinnity 07][Herman, Prasad & McGinnity 08]. In addition, existing works on Type-2 systems in BCI applications have been combined with genetic algorithms or support vector machines. On the other hand, Type-2 neuro-fuzzy systems have demonstrated great potential in handling higher noise levels compared to their Type-1 counterparts. This, subsequently, makes the proposed eT2FIS model a natural candidate for potentially addressing the noise issues faced by BCI researchers. In addition, eT2FIS is an evolving neuro-fuzzy system, which is useful in addressing the issue of non-stationaries in the data.

Hydrology research

Amid climate changes resulting from global warming and changing landscapes linked to urbanization, it is increasingly important to understand the potential impacts of such changes to our water system. In particular, the relationship between rainfall and runoff is an important indicator to these impacts, i.e. , to calculate the amount of rainfall that is eventually converted into runoff. In reality, it is important to have a reliable rainfall-runoff model because of its vast applications in different hydrologic problems like flood forecasting, design of spillways and waterways, water quality modeling, and urban planning and management [Talei, Chua & Quek 10].

Generally, there are two main approaches to model rainfall-runoff; namely, using physically-based models, and using system theoretic models. Physically-based models are designed to approximate the general internal sub-processes and physical mechanisms which govern the hydrologic cycle. An example of such a model is the kinematic wave (KW) model [Lighthill & Whitham 55], whose solution relies on solving simplified form of the shallow water equations which are based on the conservation laws of mass and momentum for the flow in channels and on overland planes. The incorporation of physical laws in the models is the

characteristics of physically-based models. Although these models are able to provide some insights to the physics of hydrologic processes, much expertise is needed for the use of the models due to their sophisticated mathematical requirements.

On the other hand, system theoretic models offer a different approach to this problem of rainfall-runoff modeling, where no detailed consideration of the underlying physical processes is required. Linear time series models such as autoregressive moving average, and regression models such as neural networks, are some examples of this class of models. As explained in [Jacquin & Shamseldin 06], system theoretic models are especially useful when hydrological data is scarce, and this is often the situation in many developing countries with a scarce hydrological network. In addition, system theoretic models have shown comparable accuracies compared with more sophisticated models [World Meteorological Organization], making them a viable option for rainfall-runoff modeling. For example, models using the concepts of nearest neighbour [Shamseldin & O'Connor 96] and neural networks [Rajurkar, Kothiyari & Chaube][Chua & Wong 10] have appeared in the literature of rainfall-runoff modeling. Although these methods allow the direct modeling of the problem from the data without the need to incorporate complicated physical laws, they function as black-boxes where no human interpretable knowledge can be extracted to understand the underlying problem. On the other hand, neuro-fuzzy systems offer valuable knowledge to the application problem, thus igniting interest in employing them for the modeling of rainfall-runoff in recent years [Nayak et. al. 04][Jacquin & Shamseldin 06][Dorum et. al. 10][Talei, Chua & Quek 10]. However, the existing works focus on the use of precise fuzzy models. This, subsequently, can limit the interpretability of the neuro-fuzzy systems. Hence, it is worthwhile to investigate the use of linguistic fuzzy models for the task of rainfall-runoff modeling, so as to exploit on the high interpretability of the systems. In particular, the proposed SaFIN model has demonstrated a high level of interpretability and excellent modeling performances in several benchmarking and real-world applications, thus possessing great potential in ad-

addressing the issues in rainfall-runoff modeling, i.e. , to provide a reliable rainfall-runoff model, while providing critical insights to the application problem.

Bibliography

- [Abraham & Robins 05] Abraham W. C. and Robins A. (2005). Memory retention–The synaptic stability versus plasticity dilemma. *Trends in Neurosciences*, 28(2), 73–78.
- [Ang, Quek & Pasquier 03] Ang K. K., Quek C. and Pasquier M. (2003). POPFNN-CRI(S): A pseudo outer product based fuzzy neural network using the compositional rule of inference and singleton fuzzifier. *IEEE Transactions on Systems, Man, and Cybernetics–B*, 33(6), 838–849.
- [Ang & Quek 05] Ang K. K. and Quek C. (2005). RSPOP: Rough set-based pseudo outer-product fuzzy rule identification algorithm. *Neural Computation*, 17(1), 205–243.
- [Ang & Quek 06a] Ang K. K. and Quek C. (2006). Rough set-based neuro-fuzzy system. *International Joint Conference on Neural Networks*, 742–749.
- [Ang & Quek 06b] Ang K. K. and Quek C. (2006). Stock trading using RSPOP: A novel rough set-based neuro-fuzzy approach. *IEEE Transactions on Neural Networks*, 17(5), 1301–1315.
- [Angelov & Filev 04] Angelov P. P. and Filev D. P. (2004). An approach to online identification of Takagi-Sugeno fuzzy models. *IEEE Transactions on Systems, Man, and Cybernetics–B*, 34(1), 484–498.
- [Arena, Haggard & Yan 08] Arena M. P., Haggard K. S. and Yan X. (2008). Price momentum and idiosyncratic volatility. *Financial Review*, 43(2), 159–190.
- [Aruna, Puviarasan & Palaniappan 05] Aruna P., Puviarasan N. and Palaniappan B. (2005). An investigation of neuro-fuzzy systems in psychosomatic disorders. *Expert Systems with Applications*, 28(4), 673–679.
- [Badrinath & Wahal 02] Badrinath S. G. and Wahal S. (2002). Momentum trading by institutions. *Journal of Finance*, 57(6), 2449–2478.
- [Baraldi & Blonda 99a] Baraldi A. and Blonda P. (1999). A survey of fuzzy clustering algorithms for pattern recognition–Part I. *IEEE Transactions on Systems, Man, and Cybernetics–B*, 29(6), 778–785.
- [Baraldi & Blonda 99b] Baraldi A. and Blonda P. (1999). A survey of fuzzy clustering algorithms for pattern recognition–Part II. *IEEE Transactions on Systems, Man, and Cybernetics–B*, 29(6), 786–801.

- [Barberis, Shleifer & Vishny 98] Barberis N., Shleifer A. and Vishny R. (1998). A model of investor sentiment. *Journal of Financial Economics*, 49, 307–343.
- [Battiti 94] Battiti R. (1994). Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4), 537–550.
- [Berenji & Khedkar 92] Berenji H. R. and Khedkar P. (1992). Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Transactions on Neural Networks*, 3, 724–740.
- [Bernard & Thomas 90] Bernard V. and Thomas J. (1990). Evidence that stock markets do not fully reflect the implications of current earnings for future earnings. *Journal of Accounting and Economics*, 13, 305–341.
- [Bezdek 81] Bezdek J. C. (1981). *Pattern recognition with fuzzy objective function algorithms*. Plenum, New York.
- [Bezdek, Tsao & Pal 92] Bezdek J. C., Tsao E. C. -K. and Pal N. R. (1992). Fuzzy Kohonen clustering networks. *IEEE Conference on Fuzzy Systems*, 1035–1043.
- [Bishop 95] Bishop C. M. (1995). *Neural networks for pattern recognition*. Clarendon, Oxford.
- [Blum & Langley 97] Blum A. and Langley P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97, 245–271.
- [Brown & Harris 94] Brown M. and Harris C. (1994). *Neurofuzzy adaptive modeling and control*. Prentice Hall, New Jersey.
- [Carpenter et. al. 92] Carpenter G. A., Grossberg S., Markuzon N., Reynolds J. H. and Rosen D. B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, 3(5), 698–713.
- [Casillas et. al. 03] Casillas J., Cordon O., Herrera F. and Magdalena L. (2003). *Interpretability in fuzzy modeling*. Springer, New York, 605–620.
- [Castellano et. al. 05] Castellano G., Castiello C., Fanelli A. M. and Mencar C. (2005). Knowledge discovery by a neuro-fuzzy modeling framework. *Fuzzy Sets and Systems*, 149, 187–207.
- [Chan, Jegadeesh & Lakonishok 97] Chan L., Jegadeesh N. and Lakonishok J. (1997). Momentum strategies. *Journal of Finance*, 51, 1681–1713.
- [Chen & Tai 10] Chen C. L. and Tai C. L. (2010). Adaptive fuzzy colour segmentation with neural network for road detections. *Engineering Applications of Artificial Intelligence*, 23(3), 400–410.
- [Cherkassky & Mulier 07] Cherkassky V. and Mulier F. (2007). *Learning from data: Concepts, theory, and methods*. John Wiley & Sons, New Jersey.

- [Cho, Chow & Leung 99] Cho S. Y., Chow T. W. S. and Leung C. T. (1999). A neural-based crowd estimation by hybrid global learning algorithm. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 29(4), 535–541.
- [Cho et. al. 09] Cho S. Y., Quek C., Seah S. X. and Chong C. H. (2009). Hebb²-Traffic: A novel application of neuro-fuzzy network for visual based traffic monitoring system. *Expert Systems with Applications*, 36(3), 6343–6356.
- [Chow & Huang 05] Chow T. W. and Huang D. (2005). Estimating optimal feature subsets using efficient estimation of high-dimensional mutual informations. *IEEE Transactions on Neural Networks*, 16(1), 213–224.
- [Chua & Wong 10] Chua H. C. L. and Wong S. W. W. (2010). Improving event-based rainfall-runoff modeling using a combined artificial neural network-kinematic wave approach. *Journal of Hydrology*, 390(1–2), 92–107.
- [Cloete & Zurada 00] Cloete I. and Zurada J. M. (2000). *Knowledge-based neurocomputing*. MIT Press, Cambridge.
- [Cohen, Gompers & Vuolteenaho 02] Cohen R. B., Gompers P. A. and Vuolteenaho T. (2002). Who underreacts to cash-flow news? Evidence from trading between individuals and institutions. *Journal of Financial Economics*, 66(2–3), 409–462.
- [Cover & Hart 67] Cover T. M. and Hart P. E. (1967). Nearest pattern classification. *IEEE Transactions on Information Theory*, 13, 21–27.
- [Coyle, Prasad & McGinnity 09] Coyle D., Prasad G. and McGinnity T. M. (2009). Faster self-organizing fuzzy neural network training and a hyperparameter analysis for a brain-computer interface. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 39(6), 1458–1471.
- [Cutler, Poterba & Summers 91] Cutler N., Poterba J. and Summers L. (1991). Speculative dynamics. *Review of Economic Studies*, 58, 529–546.
- [Daniel, Hirshleifer & Subrahmanyam 98] Daniel K., Hirshleifer D. and Subrahmanyam A. (1998). Investor psychology and security market under and overreactions. *Journal of Finance*, 53, 1839–1886.
- [Darbha, Rajagopal & Tyagi 08] Darbha S., Rajagopal K. R. and Tyagi V. (2008). A review of mathematical models for the flow of traffic and some recent results. *Nonlinear Analysis*, 69, 950–970.
- [De Bondt & Thaler 85] De Bondt W. and Thaler R. (1985). Does the stock market overreact? *Journal of Finance*, 40, 793–808.
- [De Cheveigne & Simon 07] De Cheveigne A. and Simon J. Z. (2007). Denoising based on time-shift PCA. *Journal of Neuroscience Methods*, 165(2), 297–305.
- [Dia 01] Dia H. (2001). An object-oriented neural network approach to short-term traffic forecasting. *European Journal of Operational Research*, 131(2), 253–261.

- [Dimitriou, Tsekeris & Stathopoulos 08] Dimitriou L., Tsekeris T. and Stathopoulos A. (2008). Adaptive hybrid fuzzy rule-based system approach for modeling and prediction urban traffic flow. *Transportation Research Part C: Emerging Technologies*, 16(5), 554–573.
- [Dorum et. al. 10] Dorum A., Yazar A., Sevimli M. F. and Onüçyildiz M. (2010). Modeling the rainfallrunoff data of susurluk basin. *Expert Systems with Applications*, 37(9), 6587–6593.
- [Dougherty & Kirby 94] Dougherty M. S. and Kirby H. R. (1994). The use of neural networks to recognize and predict traffic congestion. *Traffic Engineering and Control*, 34(6), 311–314.
- [Driankov, Hellendoom & Reinfrank 96] Driankov D., Hellendoom H. and Reinfrank M. (1996). *An introduction to fuzzy control*. Springer-Verlag, New York.
- [Du 10] Du K. -L. (2010). Clustering: A neural network approach. *Neural Networks*, 23, 89–107.
- [Fama & French 92] Fama E. and French K. (1992). The cross-section of expected stock returns. *Journal of Finance*, 47, 427–465.
- [Frayman & Wang 98] Frayman Y. and Wang L. P. (1998). Data mining using dynamically constructed recurrent fuzzy neural networks. *Research and Development in Knowledge Discovery and Data Mining, Lecture Notes in Computer Science*, 1394, 122–131.
- [Fu & Wang 03] Fu X. and Wang L. (2003). Data dimensionality reduction with application to simplifying RBF network structure and improving classification performance. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 33(3), 399–409.
- [Jain, Murty & Flynn 99] Jain A. K., Murty M. N. and Flynn P. J. (1999). Data clustering: a review. *ACM Computing Surveys*, 31(3), 264–323.
- [Gabrys & Bargiela 00] Gabrys B. and Bargiela A. (2000). General fuzzy min-max neural network for clustering and classification. *IEEE Transactions on Neural Networks*, 11, 769–783.
- [Garber & Hoel 02] Garber N.J. and Hoel L. A. (2002) *Traffic and highway engineering*. Thomson Learning, California.
- [Ghatee & Hashemi 09] Ghatee M. and Hashemi S. M. (2009). Traffic assignment model with fuzzy level of travel demand: An efficient algorithm based on quasi-Logit formulas. *European Journal of Operational Research*, 194(2), 432–451.
- [Goh, Lim & Quek 09] Goh H., Lim J. H. and Quek C. (2009). Fuzzy associated conjuncted maps network. *IEEE Transactions on Neural Networks*, 20(8), 1302–1318.
- [Greenberg 59] Greenberg H. (1959). An analysis of traffic flow. *Operations Research*, 7(1), 79–85.
- [Greenshields et. al. 35] Greenshields B. D., Biddins J. R., Channing W. S. and Miller H. H. (1935). A study of traffic capacity. *Highway Research Board Proceedings*, 14(1), 448–477.

- [Griffin & Tversky 92] Griffin D. and Tversky A. (1992). The weighing of evidence and the determinants of confidence. *Cognitive Psychology*, 24, 411–435.
- [Griffin, Harris & Topaloglu 03] Griffin J. M., Harris J. H. and Topaloglu S. (2003). The dynamics of institutional and individual trading. *Journal of Finance*, 58(6), 2285–2320.
- [Grinblatt & Keloharju 01] Grinblatt M. and Keloharju M. (2001). What makes investors trade? *Journal of Finance*, 56, 589–616.
- [Grossberg 82] Grossberg S. (1982). *Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control*. Reidel Press, Boston.
- [Guillaume 01] Guillaume S. (2001). Designing fuzzy inference systems from data: An interpretability-oriented review. *IEEE Transactions on Fuzzy Systems*, 9(3), 426–442.
- [Guler & Ubeyli 05] Guler I. and Ubeyli E. D. (2005). Adaptive neuro-fuzzy inference system for classification of EEG signals using wavelet coefficients. *Journal of Neuroscience Methods*, 148(2), 113–121.
- [Guyon & Elisseeff 03] Guyon A. and Elisseeff A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, 1157–1182.
- [Hagras 04] Hagras H. A. (2004). A hierarchical Type-2 fuzzy logic control architecture for autonomous mobile robots. *IEEE Transactions on Fuzzy Systems*, 12(4), 524–539.
- [Hayashi, Nomura & Yamasaki 92] Hayashi I., Nomura H. and Yamasaki H. (1992). Construction of fuzzy inference rules by NDF and NDFL. *International Journal of Approximate Reasoning*, 6(2), 241–266.
- [Haykin 98] Haykin S. (1998). *Neural networks: A comprehensive foundation*. Prentice Hall, New Jersey.
- [Held et. al. 06] Held C. M., Heiss J. E., Estevez P. A., Perez C. A., Garrido M., Algarin C. and Peirano P. (2006). Extracting fuzzy rules from polysomnographic recordings for infant sleep classification. *IEEE Transactions on Biomedical Engineering*, 53(10), 1954–1962.
- [Herman, Prasad & McGinnity 05] Herman P., Prasad G. and McGinnity T. M. (2005). Investigation of the Type-2 fuzzy logic approach to classification in an EEG-based brain-computer interface. *IEEE Conference on Engineering in Medicine and Biology Society*, 5354–5357.
- [Herman, Prasad & McGinnity 07] Herman P., Prasad G. and McGinnity T. M. (2007). Support vector-enhanced design of a T2FL approach to motor imagery-related EEG pattern recognition. *IEEE Conference on Fuzzy Systems*, 1–6.
- [Herman, Prasad & McGinnity 08] Herman P., Prasad G. and McGinnity T. M. (2008). Design and on-line evaluation of Type-2 fuzzy logic system-based framework for handling uncertainties in BCI classification. *IEEE Conference on Engineering in Medicine and Biology Society*, 4242–4245.

- [Hong & Stein 99] Hong H. and Stein J. (1999). A unified theory of underreaction, momentum trading, and overreaction in asset markets. *Journal of Finance*, 54, 2143–2184.
- [Hoogendoorn & Bovy 01] (2001). State-of-the-art vehicular traffic flow modeling. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 215(4), 283–303.
- [Horenko 10] Horenko I. (2010). On clustering of non-stationary meteorological time series. *Dynamics of Atmospheres and Oceans*, 49, 164–187.
- [Hu et. al. 04] Hu W., Xiao X., Xie D., Tan T. and Maybank S. (2004). Traffic accident prediction using 3-D model-based vehicle tracking. *IEEE Transactions on Vehicular Technology*, 53(3), 677–694.
- [Inza et. al. 02] Inza I., Sierra B., Blanco R. and Larrañaga P. (2002). Gene selection by sequential search wrapper approaches in microarray cancer class prediction. *Journal of Intelligent and Fuzzy Systems*, 12(1), 25–33.
- [Iriarte et. al. 03] Iriarte J., Urrestarazu E., Valencia M., Alegre M., Malanda A., Viteri C. and Artieda J. (2003). Independent component analysis as a tool to eliminate artifacts in EEG: A quantitative study. *Journal of Clinical Neurophysiology*, 20(4), 249–257.
- [Jacquin & Shamseldin 06] Jacquin A. P. and Shamseldin A. Y. (2006). Development of rainfall-runoff models using Takagi-Sugeno fuzzy inference systems. *Journal of Hydrology*, 329(1–2), 154–173.
- [Jang 93] Jang J. S. R. (1993). ANFIS: Adaptive-network-based fuzzy inference systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 23, 665–685.
- [Jang & Sun 95] Jang J. S. R. and Sun C. T. (1995). Neuro-fuzzy modeling and control. *Proceedings of the IEEE*, 83(3), 378–406.
- [Jegadeesh & Titman 93] Jegadeesh N. and Titman S. (1993). Returns to buying winners and selling losers: Implications for stock market efficiency. *Journal of Finance*, 48, 65–69.
- [Jensen & Shen 04] Jensen R. and Shen Q. (2004). Semantics-preserving dimensionality reduction: Rough and fuzzy-rough-based approaches. *IEEE Transactions on Knowledge and Data Engineering*, 16(12), 1457–1471.
- [Jin, Von Seelen & Sendhoff 99] Jin Y., Von Seelen W. and Sendhoff B. (1999). On generating flexible, complete, consistent and compact fuzzy rule systems from data using evolution strategies. *IEEE Transactions on Systems, Man, and Cybernetics–B*, 29(6), 829–845.
- [Juang & Lin 98] Juang C. F. and Lin C. T. (1998). An on-line self-constructing neural fuzzy inference network and its applications. *IEEE Transactions on Fuzzy Systems*, 6(1), 12–32.
- [Juang & Tsao 08a] Juang C. F. and Tsao Y. W. (2008). A self-evolving interval Type-2 fuzzy neural network with online structure and parameter learning. *IEEE Transactions on Fuzzy Systems*, 16(6), 1411–1424.

- [Juang & Tsao 08b] Juang C. F. and Tsao Y. W. (2008). A Type-2 self-organizing neural fuzzy system and its FPGA implementation. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 38(6), 1537–1548.
- [Juang & Hsu 09] Juang C. F. and Hsu C. H. (2009). Reinforcement interval Type-2 fuzzy controller design by online rule generation and Q-value-aided ant colony optimization. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 39(6), 1528–1542.
- [Juang, Huang & Cheng 10] Juang C. F., Huang R. B. and Cheng W. Y. (2010). An interval Type-2 fuzzy-neural network with support-vector regression for noisy regression problems. *IEEE Transactions on Fuzzy Systems*, 18(4), 686–699.
- [Karnik, Mendel & Liang 99] Karnik N. N., Mendel J. M. and Liang Q. (1999). Type-2 fuzzy logic systems. *IEEE Transactions on Fuzzy Systems*, 7(6), 643–658.
- [Kasabov 01] Kasabov N. (2001). Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 31(6), 902–918.
- [Kasabov & Song 02] Kasabov N. and Song Q. (2002). DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *IEEE Transactions on Fuzzy Systems*, 10(2), 144–154.
- [Kecman 01] Kecman V. (2001). *Learning and soft computing: Support vector machines, neural networks, and fuzzy logic models*. MIT Press, Cambridge.
- [Keller, Yager & Tahani 92] Keller J. M., Yager R. R. and Tahani H. (1992). Neural network implementation of fuzzy logic. *Fuzzy Sets and Systems*, 45, 1–12.
- [Kim & Kasabov 99] Kim J. and Kasabov N. (1999). HyFIS: Adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems. *Neural Networks*, 12(9), 1301–1319.
- [Knight 03] Knight J. N. (2003). Signal fraction analysis and artifact removal in EEG. *Masters Thesis, Department of Computer Science, Colorado State University*.
- [Kohavi & John 97] Kohavi R. and John G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2), 273–324.
- [Kohonen 82] Kohonen T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43, 59–69.
- [Ledoux 97] Ledoux C. (1997). An urban traffic flow model integrating neural networks. *Transportation Research Part C: Emerging Technologies*, 5(5), 287–300.
- [Lee & Choi 03] Lee H. and Choi S. (2003). PCA+HMM+SVM for EEG pattern classification. *International Symposium on Signal Processing and its Applications*, 1, 541–544.
- [Lee & Verleysen 09] Lee J. A. and Verleysen M. (2009). Quality assessment of dimensionality reduction: rank-based criteria. *Neurocomputing*, 72(7–9), 1431–1443.

- [Lespinsats et. al. 07] Lespinats S., Verleysen M., Giron A. and Fertil B. (2007). DD-HDS: a method for visualization and exploration of high-dimensionality data. *IEEE Transactions on Neural Networks*, 18(5), 1265–1279.
- [Li & Lee 03] Li C. and Lee C. -Y. (2003). Self-organizing neuro-fuzzy system for control of unknown plants. *IEEE Transactions on Fuzzy Systems*, 11(1), 135–150.
- [Li & Guan 06] Li Y. and Guan C. (2006). An extended EM algorithm for joint feature extraction and classification in brain-computer interfaces. *Neural Computation*, 18(11), 2730–2761.
- [Liang & Mendel 00] Liang Q. and Mendel J. M. (2000). Interval Type-2 fuzzy logic systems: Theory and design. *IEEE Transactions on Fuzzy Systems*, 8(5), 535–550.
- [Lighthill & Whitham 55] Lighthill M. J. and Whitham C. B. (1955). On kinematic wave. I. Flood management in long rivers. *Proceedings of the Royal Society of London Series A*, 229, 281–316.
- [Lim & Jin 06] Lim J. H. and Jin J. S. (2006). Discovering recurrent image semantics from class discrimination. *EURASIP Journal of Applied Signal Processing*, 21, 1–11.
- [Lim, Tian & Mulhem 03] Lim J. H., Tian Q. and Mulhem P. (2003). Home photo content modeling for personalized event-based retrieval. *IEEE on Multimedia*, 10(4), 28–37.
- [Lin & Lin 97] Lin C. J. and Lin C. T. (1997). An ART-based fuzzy adaptive learning control network. *IEEE Transactions on Fuzzy Systems*, 5(4), 477–496.
- [Lin & Lee 91] Lin C. T. and Lee C. S. G. (1991). Neural-network-based fuzzy logic control and decision system. *IEEE Transactions on Computers*, 40(12), 1320–1336.
- [Lin 94] Lin C. T. (1994). *Neural fuzzy control systems with structure and parameter learning*. World Scientific.
- [Lin & Lee 96] Lin C. T. and Lee C. S. G. (1996). *Neural fuzzy systems: A neuro-fuzzy synergism to intelligent systems*. Prentice Hall, New York.
- [Lin et. al. 05] Lin C. T., Chen Y. C., Wu R. C., Liang S. F. and Huang T. Y. (2005). Assessment of driver's driving performance and alertness using EEG-based fuzzy neural networks. *IEEE International Symposium on Circuits and Systems*, 1, 152–155.
- [Liu, Quek & Ng 07] Liu F., Quek C. and Ng G. S. (2007). A novel generic hebbian ordering-based fuzzy rule base reduction approach to Mamdani neuro-fuzzy system. *Neural Computation*, 19, 1656–1680.
- [Lotte 06] Lotte F. (2006). The use of fuzzy inference systems for classification in EEG-based brain-computer interfaces. *International Brain-Computer Interface Workshop*, 42–43.
- [Mackey & Glass 77] Mackey M. and Glass L. (1977). Oscillation and chaos in physiological control systems. *Science*, 197, 287–289.
- [Mahnke, Kaupužs & Lubashevsky 05] Mahnke R., Kaupužs J. and Lubashevsky I. (2005). Probabilistic description of traffic flow. *Physics Reports*, 408, 1–130.

- [Malhotra & Malhotra 02] Malhotra R. and Malhotra D. K. (2002). Differentiating between good credits and bad credits using neuro-fuzzy systems. *European Journal of Operational Research*, 136(1), 190–211.
- [Mamdani 77] Mamdani E. H. (1977). Application of fuzzy logic to approximate reasoning using linguistic systems. *IEEE Transactions on Computers*, C-26, 1182–1191.
- [Mandler & Bauer 88] Mandler J. M. and Bauer P. J. (1988). The cradle of categorization: Is the basic level basic? *Cognitive Development*, 3, 247–264.
- [Mandler, Bauer & McDonough 91] Mandler J. M., Bauer P. J. and McDonough L. (1991). Separating the sheep from the goats: Differentiating global categories. *Cognitive Psychology*, 23, 263–298.
- [Mandler & McDonough 93] Mandler J. M. and McDonough L. (1993). Concept formation in infancy. *Cognitive Development*, 8, 291–318.
- [Mandler & McDonough 98a] Mandler J. M. and McDonough L. (1998). On developing a knowledge base in infancy. *Development Psychology*, 34, 1274–1288.
- [Mandler & McDonough 98b] Mandler J. M. and McDonough L. (1998). Studies in inductive inference in infancy. *Cognitive Psychology*, 37, 60–96.
- [Mandler 08] Mandler J. M. (2008). On the birth and growth of concepts. *Philosophical Psychology*, 21(2), 207–230.
- [Mantaras 90] Mantaras R. L. (1990). *Approximate reasoning models*. Ellis Horwood, England.
- [Maren & Baudry 95] Maren S. and Baudry M. (1995). Properties and mechanisms of long-term synaptic plasticity in the mammalian brain: relationships to learning and memory. *Neurobiology Learning and Memory*, 63(1), 1–18.
- [Margaliot & Langholz 00] Margaliot M. and Langholz G. (2000). *New approaches to fuzzy modeling and control: Design and analysis*. World Scientific, Singapore.
- [McCulloch & Pitts 43] McCulloch W. S. and Pitts W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, 115–133.
- [McLean 10] McLean R.D. (2010). Idiosyncratic risk, long-term reversal, and momentum. *Journal of Financial and Quantitative Analysis*, 45(4), 883–906.
- [Medasani, Kim & Krishnapuram 98] Medasani S., Kim J. and Krishnapuram R. (1998). An overview of membership function generation techniques for pattern recognition. *International Journal of Approximate Reasoning*, 19, 391–417.
- [Mele & Fornari 00] Mele A. and Fornari F. (2000). *Stochastic volatility in financial markets: Crossing the bridge to continuous time*. Kluwer Academic Publisher, Norwell.
- [Mendel 01] Mendel J. M. (2001). *Uncertain rule-based fuzzy logic systems: Introduction and new directions*. Prentice Hall, Upper Saddle River.

- [Mendel & John 02] Mendel J. M. and John R. I. B. (2002). Type-2 fuzzy sets made simple. *IEEE Transactions on Fuzzy Systems*, 10(2), 117–127.
- [Mendel, John & Liu 06] Mendel J. M., John R. I. and Liu F. (2006). Interval Type-2 fuzzy logic systems made simple. *IEEE Transactions on Fuzzy Systems*, 14(6), 808–821.
- [Mendel 07] Mendel J. M. (2007). Type-2 fuzzy sets and systems: An overview. *IEEE Computational Intelligence Magazine*, 2(1), 20–29.
- [Mitra & Hayashi 00] Mitra S. and Hayashi Y. (2000). Neuro-fuzzy rule generation: Survey in soft computing framework. *IEEE Transactions on Neural Networks*, 11(3), 748–768.
- [Mizumoto & Tanaka 76] Mizumoto M. and Tanaka K. (1976). Some properties of fuzzy sets of type 2. *Information and Control*, 31, 312–340.
- [Moody et. al. 98] Moody J., Wu L., Liao Y. and Saffell M. (1998). Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting*, 17(5), 441–470.
- [Nakanishi, Turksen & Sugeno 93] Nakanishi H., Turksen I. and Sugeno M. (1993). A review and comparison of six reasoning methods. *Fuzzy Sets and Systems*, 57(3), 257–294.
- [Nauck, Klawonn & Kruse 97] Nauck D., Klawonn F. and Kruse R. (1997). *Foundations of neuro-fuzzy systems*. John Wiley & Sons, New York.
- [Nayak et. al. 04] Nayak P. C., Sudheer K. P., Rangan D. M. and Ramasastri K. S. (2004). A neuro-fuzzy computing technique for modeling hydrological time series. *Journal of Hydrology*, 291(1–2), 52–66.
- [Ngo & Li 94] Ngo C. Y. and Li O. K. (1994). Freeway traffic control using fuzzy logic controllers. *Information Sciences – Applications*, 1(2), 59–76.
- [Niedermeyer & Da Silva 04] Niedermeyer E. and Da Silva F. H. L. (2004). *Electroencephalography: Basic principles, clinical applications, and related fields*. Lippincott Williams & Wilkins, Philadelphia.
- [Nishina & Hagiwara 97] Nishina T. and Hagiwara M. (1997). Fuzzy inference neural network. *Neurocomputing*, 14(3), 223–239.
- [Oentaryo, Pasquier & Quek 08] Oentaryo R. J., Pasquier M. and Quek C. (2008). GenSoFNN-Yager: A novel brain-inspired generic self-organizing neuro-fuzzy system realizing Yager inference. *Expert Systems with Applications*, 35(4), 1825–1840.
- [Pal 99] Pal N. R. (1999). Soft computing for feature analysis. *Fuzzy Sets and Systems*, 103(2), 201–221.
- [Palaniappan 02] Palaniappan R., Paramesran R., Nishida S. and Saiwaki N. (2002). A new brain-computer interface design using fuzzy ARTMAP. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 10(3), 140–148.

- [Panella & Gallo 05] Panella M. and Gallo A. S. (2005). An input-output clustering approach to the synthesis of ANFIS networks. *IEEE Transactions on Fuzzy Systems*, 13(1), 69–81.
- [Pasquier, Quek & Toh 01] Pasquier M., Quek C. and Toh M. (2001). Fuzzylot: A self-organizing fuzzy-neural rule-based pilot system for automated vehicles. *Neural Networks*, 14(8), 1099–1112.
- [Perkins & Theiler 03] Perkins S. and Theiler J. (2003). Online feature selection using grafting. *International Conference on Machine Learning*.
- [Plummer & Ridley 03] Plummer T. and Ridley A. (2003). *Forecasting financial markets: The psychology of successful investing (4th ed.)*. Kogan Page, London.
- [Quah & Quek 06] Quah K. H. and Quek C. (2006). FITSK: Online local learning with generic fuzzy input Takagi-Sugeno-Kang fuzzy framework for nonlinear system estimation. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 36(1), 166–178.
- [Quek & Zhou 96] Quek C. and Zhou R. W. (1996). POPFNN: A pseudo outer-product based fuzzy neural network. *Neural Networks*, 9(9), 1569–1581.
- [Quek & Zhou 99] Quek C. and Zhou R. W. (1999). POPFNN-AAR(S): A pseudo outer-product based fuzzy neural network. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 29(6), 859–870.
- [Quek & Singh 05] Quek C. and Singh A. (2005). POP-Yager: A novel self-organizing fuzzy neural network based on Yager inference. *Expert Systems with Applications*, 29, 229–242.
- [Quek, Pasquier & Lim 09] Quek C., Pasquier M. and Lim B. (2009). A novel self-organizing fuzzy rule-based system for modelling traffic flow behaviour. *Expert Systems with Applications*, 36(10), 12167–12178.
- [Rajurkar, Kothiyari & Chaube] Rajurkar M. P., Kothiyari U. C. and Chaube U. C. (2004). Modeling of the daily rainfall-runoff relationship with artificial neural network. *Journal of Hydrology*, 285(1–4), 96–113.
- [Rong et. al. 06] Rong H. J., Sundararajan N., Huang G. B. and Saratchandra P. (2006). Sequential adaptive fuzzy inference system (SAFIS) for nonlinear system identification and prediction. *Fuzzy Sets and Systems*, 157(9), 1260–1275.
- [Rouwenhorst 98] Rouwenhorst G. (1998). International momentum strategies. *Journal of Finance*, 53, 267–284.
- [Rumelhart & Zipser 85] Rumelhart D. E. and Zipser D. (1985). Feature discovery by competitive learning. *Cognitive Science*, 9(1), 75–112.
- [Ruspini 82] Ruspini E. H. (1982). Recent development in fuzzy clustering. *Fuzzy Set and Possibility Theory*, 113–147.
- [Schadschneider 02] Schadschneider A. (2002). Traffic flow: A statistical physics point of view. *Physica A: Statistical Mechanics and its Applications*, 313(1–2), 153–187.

- [Sepúlveda et. al. 07] Sepúlveda R., Castillo O., Melin P., Rodríguez-Díaz A. and Montiel O. (2007). Experimental study of intelligent controllers under uncertainty using Type-1 and Tye-2 fuzzy logic. *Information Sciences*, 177(10), 2023–2048.
- [Shamseldin & O'Connor 96] Shamseldin A. and O'Connor K. M. (1996). A nearest neighbour linear perturbation model for river flow forecasting. *Journal of Hydrology*, 179, 353–375.
- [Shen & Chouchoulas 02] A rough-fuzzy approach for generating classification rules. *Pattern Recognition*, 35(11), 2425–2438.
- [Shenoy et. al. 06] Shenoy P., Krauledat M., Blankertz B., Rao R. P. N. and Müller K. -R. (2006). Towards adaptive classification for BCI. *Journal of Neural Engineering*, 3(1), 13–23.
- [Simmonds & Hack 00] Simmonds R. and Hack G. (2000). *Global city regions: Their emerging forms*. Spon Press, London.
- [Simpson 92] Simpson P. K. (1992). Fuzzy min-max neural networks–Part I: Classification. *IEEE Transactions on Neural Networks*, 3(5), 776–786.
- [Song et. al. 09] Song H., Miao C., Shen Z., Miao Y. and Lee B. -S. (2009). A fuzzy neural network with fuzzy impact grades. *Neurocomputing*, 72(13–15), 3098–3122.
- [Stathopoulos & Karlaftis 01] Stathopoulos A. and Karlaftis M. G. (2001). Temporal and spatial variations of real-time traffic data in urban areas. *Transportation Research Record*, 1768, 135–140.
- [Sugeno & Kang 88] Sugeno M. and Kang G. T. (1988). Structure identification of fuzzy model. *Fuzzy Sets and Systems*, 28(1), 15–33.
- [Sugeno & Yasukawa 93] Sugeno M. and Yasukawa T. (1993). A fuzzy-logic-based approach to qualitative modeling. *IEEE Transactions on Fuzzy Systems*, 1(1), 7–31.
- [Takagi & Sugeno 85] Takagi T. and Sugeno M. (1985). Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1), 116–132.
- [Talei, Chua & Quek 10] Talei A., Chua H. C. L. and Quek C. (2010). A novel application of a neuro-fuzzy computational technique in event-based rainfall-runoff modeling. *Expert Systems with Applications*, 37(12), 7456–7468.
- [Tan, Quek & Ng 05] Tan T. Z., Quek C. and Ng G. S. (2005). Ovarian cancer diagnosis by hippocampus and neocortex-inspired learning memory structures. *Neural Networks*, 18(5–6), 818–825.
- [Tan et. al. 07] Tan T. Z., Quek C., Ng G.S. and Ng E. Y. K. (2007). A novel cognitive interpretation of breast cancer thermography with complementary learning fuzzy neural memory structure. *Expert Systems with Applications*, 33(3), 652–666.
- [Transportation Research Board 00] Transportation Research Board, National Research Council. (2000). *Highway capacity manual 4th ed.*, Special Report 209, Washington.

- [TUNG, Quek & Guan 08] Tung S. W., Quek C. and Guan C. (2008). HyFIS-Yager-gDIC: A self-organizing hybrid neural fuzzy inference system realizing Yager inference. *International Conference on Neural Information Processing of the Asia-Pacific Neural Network Assembly*, 1137–1145.
- [TUNG, Quek & Guan 09] Tung S. W., Quek C. and Guan C. (2009). T2-HyFIS-Yager: Type-2 hybrid neural fuzzy inference system realizing Yager inference. *IEEE Conference on Fuzzy Systems*, 80–85.
- [TUNG, Quek & Guan 11] Tung S. W., Quek C. and Guan C. (2011). FAPOP: Feature analysis enhanced pseudo outer-product fuzzy rule identification system. *IEEE Conference on Fuzzy Systems*, 1530–1537.
- [Tung & Quek 02] Tung W. L. and Quek C. (2002). GenSoFNN: A generic self-organizing fuzzy neural network. *IEEE Transactions on Neural Networks*, 13(5), 1075–1086.
- [Tung, Quek & Cheng 04] Tung W. L., Quek C. and Cheng P. (2004). GenSo-EWS: A novel neural-fuzzy based early warning system for predicting bank failures. *Neural Networks*, 17(4), 567–587.
- [Tung & Quek 05a] Tung W. L. and Quek C. (2005). GenSo-FDSS: A neural-fuzzy decision support system for pediatric ALL cancer subtype identification using gene expression data. *Artificial Intelligence in Medicine*, 33, 61–88.
- [Tung & Quek 05b] Tung W. L. and Quek C. (2005). GenSo-OPATS: A brain-inspired dynamically evolving option pricing model and arbitrage trading system. *IEEE Congress on Evolutionary Computation*, 2429–2436.
- [Tung & Quek 10] Tung W. L. and Quek C. (2010). eFSM – A novel online neural-fuzzy semantic memory model. *IEEE Transactions on Neural Networks*, 21(1), 136–157.
- [Tung & Quek 11] Tung W. L. and Quek C. (2011). Financial volatility trading using a self-organising neural-fuzzy semantic network and option straddle-based approach. *Expert Systems with Applications*, 38(5), 4668–4688.
- [Turksen & Zhong 90] Turksen I. B. and Zhong Z. (1990). An approximate analogical reasoning scheme based on similarity measures and interval valued fuzzy sets. *Fuzzy Sets and Systems*, 34, 323–346.
- [Tzafestas & Zikidis 01] Tzafestas S. G. and Zikidis K. C. (2001). NeuroFAST: On-line neuro-fuzzy ART-based structure and parameter learning TSK model. *IEEE Transactions on Systems, Man, and Cybernetics-B*, 31(5), 797–802.
- [UCI] The UCI website. [Online]. Available: <http://archive.ics.uci.edu/ml/>
- [Vidaurre et. al. 06] Vidaurre C., Schlogl A., Cabeza R., Scherer R. and Pfurtscheller G. (2006). A fully on-line adaptive BCI. *IEEE Transactions on Biomedical Engineering*, 53(6), 1214–1219.
- [Vigario et. al. 00] Vigario R., Sarela J., Jousmiki V., Hamalainen M. and Oja E. (2000). Independent component approach to the analysis of EEG and MEG recordings. *IEEE Transactions on Biomedical Engineering*, 47(5), 589–593.

- [Wahab et. al. 09] Wahab A., Quek C., Tan C. K. and Takeda K. (2009). Driving profile modeling and recognition based on soft computing approach. *IEEE Transactions on Neural Networks*, 20(4), 563–582.
- [Wang & Fu 05] Wang L. P. and Fu X. J. (2005). *Data mining with computational intelligence*. Springer, Berlin.
- [Wang & Langari 95] Wang L. and Langari R. (1995). Building Sugeno-type models using fuzzy discretization and orthogonal parameter estimation techniques. *IEEE Transactions on Fuzzy Systems*, 3(4), 454–458.
- [Wang & Lee 02] Wang J. S. and Lee C. S. G. (2002). Self-adaptive neuro-fuzzy inference systems for classification applications. *IEEE Transactions on Fuzzy Systems*, 10(6), 790–802.
- [Wang & Mendel 92] Wang L. X. and Mendel J. M. (1992). Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, 22, 1414–1427.
- [Wang, Zhou & Chu 08] Wang L., Zhou N. and Chu F. (2008). A general wrapper approach to selection of class-dependent features. *IEEE Transactions on Neural Networks*, 19(7), 1267–1278.
- [Wong & Cho 09] Wong J. J. and Cho S. Y. (2009). A local experts organization model with application to face emotion recognition. *Expert Systems with Applications*, 36(1), 804–819.
- [World Meteorological Organization] World Meteorological Organization. (1975). Intercomparison of conceptual hydrological models used in operational hydrological forecasting. *Operational Hydrology Report*, 7(429), Geneva.
- [Wu, Er & Gao 01] Wu S., Er M. J. and Gao Y. (2001). A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks. *IEEE Transactions on Fuzzy Systems*, 9(4), 578–594.
- [Yang 93] Yang M. -S. (1993). A survey of fuzzy clustering. *Mathematical and Computer Modelling*, 18(11), 1–16.
- [Yang, Murata & Amari 98] Yang H. H., Murata N. and Amari S. (1998). Statistical inference: learning in artificial neural networks. *Trends in Cognitive Sciences*, 2(1), 4–10.
- [Zadeh 65] Zadeh L. A. (1965). Fuzzy sets. *Information and Control*, 8, 338–353.
- [Zadeh 75a] Zadeh L. A. (1975). The concept of a linguistic variable and its application to approximate reasoning–I. *Information Sciences*, 8(3), 199–249.
- [Zadeh 75b] Zadeh L. A. (1975). Calculus of fuzzy restrictions. *Fuzzy Sets and Their Applications to Cognitive and Decision Preocess*, 1–39.
- [Zadeh 94] Zadeh L. A. (1994). Fuzzy logic, neural networks, and soft computing. *Communications of the ACM*, 37(3).

Appendix A

Parameter Adaptation in SaFIN

The parameter adaptation in SaFIN succeeds the structure learning phase of the system. The tunable parameters of SaFIN are the centres and widths of the fuzzy labels embedded in the antecedent and consequent segments. They are optimally tuned using a supervised learning algorithm based on a gradient descent approach to minimize the error function defined as in (A.1):

$$E = \frac{1}{2} \sum_q (d_q - y_q)^2 \quad (\text{A.1})$$

where $d = [d_1, \dots, d_Q]^T$ denotes the desired output vector, and $y = [y_1, \dots, y_Q]^T$ denotes the computed output vector (see Figure 3.1).

The general updating rule for the parameters is given in (A.2):

$$W(t+1) = W(t) + \Delta W(t) \quad (\text{A.2})$$

where W denotes the tunable parameters in SaFIN and the modification ΔW is computed using (2.4). The error signals and updating steps backpropagate from the output layer to the antecedent layer of SaFIN, and the backpropagation computations are discussed here.

For the rest of this section, the following generic notations are adopted:

$$N_q = \sum_{l_q \in \{1_q \dots L_q\}} o_{l_q} c_{l_q} \sigma_{l_q} ; \quad D_q = \sum_{l_q \in \{1_q \dots L_q\}} o_{l_q} \sigma_{l_q}$$

such that the output value of O_q , defined as in (3.7), is given to be $o_q = N_q/D_q$. Also, the error signal for an arbitrary node is given as $\delta = -\partial E/\partial o$.

Layer 5: The error signal of O_q can be computed as in (A.3):

$$\delta_q = -\frac{\partial E}{\partial o_q} = -\frac{\partial E}{\partial y_q} = d_q - y_q . \quad (\text{A.3})$$

To derive the updating steps for tuning the centres and widths of the fuzzy labels embedded in the consequent segments, one needs to compute the partial derivatives of (A.4):

$$\frac{\partial o_q}{\partial c_{l_q}} = \frac{o_{l_q} \sigma_{l_q}}{D_q} ; \quad \frac{\partial o_q}{\partial \sigma_{l_q}} = \frac{D_q o_{l_q} c_{l_q} - N_q o_{l_q}}{D_q^2} . \quad (\text{A.4})$$

Subsequently, applying (A.2), the updating steps for the consequent fuzzy labels are given as in (A.5):

$$\begin{aligned} c_{l_q}(t+1) &= c_{l_q}(t) + \eta \delta_q(t) \frac{\partial o_q(t)}{\partial c_{l_q}(t)} \\ \sigma_{l_q}(t+1) &= \sigma_{l_q}(t) + \eta \delta_q(t) \frac{\partial o_q(t)}{\partial \sigma_{l_q}(t)} \end{aligned} \quad (\text{A.5})$$

where $\eta > 0$ is a learning constant.

Layer 4: Using chain rule, the error signal of C_{l_q} can be computed as in (A.6):

$$\delta_{l_q} = -\frac{\partial E}{\partial o_{l_q}} = \delta_q \frac{D_q c_{l_q} \sigma_{l_q} - N_q \sigma_{l_q}}{D_q^2} . \quad (\text{A.6})$$

Layer 3: Since a rule node R_k can contribute to more than one consequent fuzzy label C_{l_q}

by (3.6), the cumulative error signal of R_k can be computed as in (A.7):

$$\delta_k = -\frac{\partial E}{\partial o_k} = \sum_{l_q} \delta_{l_q}^{(k)} \quad (\text{A.7})$$

where $\delta_{l_q}^{(k)}$ is the error signal of the consequent node $C_{l_q}^{(k)}$, and $C_{l_q}^{(k)}$ is a consequent node inheriting the firing strength of R_k by (3.6).

Layer 2: As before, an antecedent node A_{j_p} can contribute to more than one rule node R_k by (3.5), hence the cumulative error signal of A_{j_p} can be computed as in (A.8):

$$\delta_{j_p} = \sum_k \delta_k^{(j_p)} \quad (\text{A.8})$$

where $\delta_k^{(j_p)}$ is the error signal of the rule node $R_k^{(j_p)}$, and $R_k^{(j_p)}$ is a rule node inheriting the firing strength of A_{j_p} by (3.5).

Next, to derive the updating steps for tuning the centres and widths of the fuzzy labels embedded in the antecedent segments, one needs to compute the partial derivatives of (A.9):

$$\frac{\partial o_{j_p}}{\partial c_{j_p}} = o_{j_p} \frac{2(x_p - c_{j_p})}{\sigma_{j_p}^2}; \quad \frac{\partial o_{j_p}}{\partial \sigma_{j_p}} = o_{j_p} \frac{2(x_p - c_{j_p})^2}{\sigma_{j_p}^3}. \quad (\text{A.9})$$

Subsequently, applying (A.2), the updating steps for the antecedent fuzzy labels are given as in (A.10):

$$\begin{aligned} c_{j_p}(t+1) &= c_{j_p}(t) + \eta \delta_{j_p}(t) \frac{\partial o_{j_p}(t)}{\partial c_{j_p}(t)} \\ \sigma_{j_p}(t+1) &= \sigma_{j_p}(t) + \eta \delta_{j_p}(t) \frac{\partial o_{j_p}(t)}{\partial \sigma_{j_p}(t)} \end{aligned} \quad (\text{A.10})$$

where $\eta > 0$ is a learning constant.

Appendix B

Parameter Adaptation in eT2FIS

The parameter adaptation in eT2FIS follows the structure learning phase of the system, and it is performed for each incoming data tuple. The tunable parameters of eT2FIS are the centres of the left and right formation gaussian functions of the fuzzy labels embedded in the antecedent and consequent segments. They are optimally tuned using a supervised learning algorithm based on a gradient descent approach to minimize the error function defined as in (B.1):

$$E = \frac{1}{2} \sum_q (d_q - y_q)^2 \quad (\text{B.1})$$

where $d(t) = [d_1(t), \dots, d_q(t), \dots, d_Q(t)]^T$ denotes the desired output vector, and $y(t) = [y_1(t), \dots, y_q(t), \dots, y_Q(t)]^T$ denotes the computed output vector at time instance t (see Figure 5.1). To simplify the notation, the time index has been dropped. Nevertheless, it should be noted that parameter adaptation is performed at each time instance with the arrival of a data tuple.

The general updating rule for the parameters is given in (B.2):

$$W(t+1) = W(t) + \Delta W(t) \quad (\text{B.2})$$

where W denotes the tunable parameters in eT2FIS and the modification ΔW is computed using (2.4). The error signals and updating steps backpropagate from the output layer to the antecedent layer of eT2FIS, and the backpropagation computations are discussed here.

For the rest of this section, the following generic notations are adopted:

$$N_q^L = \sum_{l_q \in \{1_q \dots L_q\}} z_{l_q} \theta_{l_q}^L; \quad D_q^L = \sum_{l_q \in \{1_q \dots L_q\}} \theta_{l_q}^L$$

$$N_q^R = \sum_{l_q \in \{1_q \dots L_q\}} z_{l_q} \theta_{l_q}^R; \quad D_q^R = \sum_{l_q \in \{1_q \dots L_q\}} \theta_{l_q}^R$$

such that the type reduced set $Y_q = [\underline{Y}_q, \overline{Y}_q]$, see (5.11), is bounded by $\underline{Y}_q = N_q^L/D_q^L$ and $\overline{Y}_q = N_q^R/D_q^R$. As explained before, the KM algorithm for computing Y_q requires the set $\{z_{l_q} = 1/2 \left(c_{l_q}^L + c_{l_q}^R \right)\}$ to be sorted in an ascending order. Let E_q be the matrix that performs such a sorting task, i.e., $E_q = (e)$ is a $L_q \times L_q$ matrix. Subsequently, the output sets of the consequent nodes $\{\underline{f}_{l_q}\}$ and $\{\overline{f}_{l_q}\}$, see (5.10), are sorted using E_q to form the sets $\{\underline{f}_{l_q}^S\}$ and $\{\overline{f}_{l_q}^S\}$ respectively. Then

$$\theta_{l_q}^L = \begin{cases} \overline{f}_{l_q}^S & \text{if } l_q = 1_q \dots l_q^{L^*} \\ \underline{f}_{l_q}^S & \text{if } l_q = l_q^{L^*} + 1 \dots L_q \end{cases}; \quad \theta_{l_q}^R = \begin{cases} \underline{f}_{l_q}^S & \text{if } l_q = 1_q \dots l_q^{R^*} \\ \overline{f}_{l_q}^S & \text{if } l_q = l_q^{R^*} + 1 \dots L_q \end{cases}$$

where $l_q^{L^*}$ and $l_q^{R^*}$ are the left and right cross-over points determined using the KM algorithm. Finally, reverse the sorting process of $\{\theta_{l_q}^L\}$ and $\{\theta_{l_q}^R\}$ using E_q^{-1} , and denote the sets as $\{\omega_{l_q}^L\}$ and $\{\omega_{l_q}^R\}$ respectively. Also, the error signal for an arbitrary node is given as $\delta = -\partial E / \partial f_o$.

Layer 5: The error signal of O_q can be computed as in (B.3):

$$\delta_q = -\frac{\partial E}{\partial f_{o_q}} = -\frac{\partial E}{\partial y_q} = d_q - y_q. \quad (\text{B.3})$$

To derive the updating steps for tuning the centres of the fuzzy labels embedded in the consequent segments, one needs to compute the partial derivatives of (B.4):

$$\frac{\partial f_{o_q}}{\partial c_{l_q}^L} = \frac{\partial f_{o_q}}{\partial c_{l_q}^R} = \frac{1}{4} \left(\frac{\omega_{l_q}^L}{D_q^L} + \frac{\omega_{l_q}^R}{D_q^R} \right). \quad (\text{B.4})$$

Subsequently, applying (B.2), the updating steps for the centres of the left and right formation gaussian functions of the consequent labels are given as in (B.5):

$$\begin{aligned} c_{l_q}^L(t+1) &= c_{l_q}^L(t) + \eta \delta_q(t) \frac{\partial f_{o_q}(t)}{\partial c_{l_q}^L(t)} \\ c_{l_q}^R(t+1) &= c_{l_q}^R(t) + \eta \delta_q(t) \frac{\partial f_{o_q}(t)}{\partial c_{l_q}^R(t)} \end{aligned} \quad (\text{B.5})$$

where $\eta > 0$ is a learning constant.

Layer 4: Using chain rule, the error signal of $\tilde{C}_{l_q}$ can be described as in (B.6):

$$\delta_{l_q} = -\frac{\partial E}{\partial f_{o_{l_q}}} = [\underline{\delta}_{l_q}, \bar{\delta}_{l_q}] \quad (\text{B.6})$$

where the bounds of the interval Type-1 set are computed accordingly as in (B.7):

$$\begin{aligned} \underline{\delta}_{l_q} &= -\frac{\partial E}{\partial f_{\underline{l}_q}} \\ &= \begin{cases} \frac{1}{2} \delta_q \left(\frac{1/2 (c_{l_q}^L + c_{l_q}^R) D_q^L - N_q^L}{(D_q^L)^2} \right) & \text{if } e_{b, l_q} = 1, b = l_q^{L^*} + 1 \dots L_q \\ 0 & \text{otherwise} \end{cases} \\ &+ \begin{cases} \frac{1}{2} \delta_q \left(\frac{1/2 (c_{l_q}^L + c_{l_q}^R) D_q^R - N_q^R}{(D_q^R)^2} \right) & \text{if } e_{b, l_q} = 1, b = 1 \dots l_q^{R^*} \\ 0 & \text{otherwise} \end{cases}; \end{aligned}$$

$$\begin{aligned}
\bar{\delta}_{l_q} &= -\frac{\partial E}{\partial \bar{f}_{l_q}} \\
&= \begin{cases} \frac{1}{2} \delta_q \left(\frac{1/2 (c_{l_q}^L + c_{l_q}^R) D_q^L - N_q^L}{(D_q^L)^2} \right) & \text{if } e_{b, l_q} = 1, b = 1 \dots l_q^{L^*} \\ 0 & \text{otherwise} \end{cases} \\
&+ \begin{cases} \frac{1}{2} \delta_q \left(\frac{1/2 (c_{l_q}^L + c_{l_q}^R) D_q^R - N_q^R}{(D_q^R)^2} \right) & \text{if } e_{b, l_q} = 1, b = l_q^{R^*} + 1 \dots L_q \\ 0 & \text{otherwise} \end{cases} . \quad (\text{B.7})
\end{aligned}$$

Layer 3: Since a rule node R_k can contribute to more than one consequent fuzzy label $\tilde{C}_{l_q}$ by (5.9), the cumulative error signal of R_k can be described as in (B.8):

$$\delta_k = -\frac{\partial E}{\partial f_{o_k}} = [\underline{\delta}_k, \bar{\delta}_k] \quad (\text{B.8})$$

where the bounds of the interval Type-1 set are computed accordingly as in (B.9):

$$\underline{\delta}_k = -\frac{\partial E}{\partial \underline{f}_k} = \sum_{l_q} \underline{\delta}_{l_q}^{(k)}; \quad \bar{\delta}_k = -\frac{\partial E}{\partial \bar{f}_k} = \sum_{l_q} \bar{\delta}_{l_q}^{(k)} \quad (\text{B.9})$$

where $\underline{\delta}_{l_q}^{(k)}$ (resp. $\bar{\delta}_{l_q}^{(k)}$) is the lower (resp. upper) bound of the error signal of $\tilde{C}_{l_q}^{(k)}$, and $\tilde{C}_{l_q}^{(k)}$ is a consequent node inheriting the lower (resp. upper) firing strength of R_k by (5.10).

Layer 2: As before, an antecedent node $\tilde{A}_{j_p}$ can contribute to more than one rule R_k by (5.7), hence the cumulative error signal of $\tilde{A}_{j_p}$ can be described as in (B.10):

$$\delta_{j_p} = -\frac{\partial E}{\partial f_{o_{j_p}}} = [\underline{\delta}_{j_p}, \bar{\delta}_{j_p}] \quad (\text{B.10})$$

where the bounds of the interval Type-1 set are computed accordingly as in (B.11):

$$\underline{\delta}_{j_p} = -\frac{\partial E}{\partial \underline{f}_{j_p}} = \sum_k \underline{\delta}_k^{(j_p)}; \quad \bar{\delta}_{j_p} = -\frac{\partial E}{\partial \bar{f}_{j_p}} = \sum_k \bar{\delta}_k^{(j_p)} \quad (\text{B.11})$$

where $\underline{\delta}_k^{(j_p)}$ (resp. $\bar{\delta}_k^{(j_p)}$) is the lower (resp. upper) bound of the error signal of $R_k^{(j_p)}$, and $R_k^{(j_p)}$ is a rule node inheriting the lower (resp. upper) firing strength of $\tilde{A}_{j_p}$ by (5.8).

Next, to derive the updating steps for tuning the centres of the fuzzy labels embedded in the antecedent segments, one needs to compute the partial derivatives of (B.12):

$$\begin{aligned} \frac{\partial f_{o_{j_p}}}{\partial c_{j_p}^L} &= \begin{cases} \bar{f}_{j_p} \frac{2(x_p - c_{j_p}^L)}{\sigma_{j_p}^2} & \text{if } x_p \leq c_{j_p}^L \\ \underline{f}_{j_p} \frac{2(x_p - c_{j_p}^L)}{\sigma_{j_p}^2} & \text{if } x_p > 1/2 (c_{j_p}^L + c_{j_p}^R) \\ 0 & \text{otherwise} \end{cases} ; \\ \frac{\partial f_{o_{j_p}}}{\partial c_{j_p}^R} &= \begin{cases} \underline{f}_{j_p} \frac{2(x_p - c_{j_p}^R)}{\sigma_{j_p}^2} & \text{if } x_p \leq 1/2 (c_{j_p}^L + c_{j_p}^R) \\ \bar{f}_{j_p} \frac{2(x_p - c_{j_p}^R)}{\sigma_{j_p}^2} & \text{if } x_p > c_{j_p}^R \\ 0 & \text{otherwise} \end{cases} . \end{aligned} \quad (\text{B.12})$$

Subsequently, applying (B.2), the updating steps for the centres of the left and right formation gaussian functions of the antecedent labels are given as in (B.13):

$$\begin{aligned} c_{j_p}^L(t+1) &= c_{j_p}^L(t) + \eta \left\{ \begin{array}{ll} \bar{\delta}_{j_p}(t) & \text{if } x_p \leq c_{j_p}^L \\ \underline{\delta}_{j_p}(t) & \text{if } x_p > 1/2 (c_{j_p}^L + c_{j_p}^R) \\ 0 & \text{otherwise} \end{array} \right\} \frac{\partial f_{o_{j_p}}(t)}{\partial c_{j_p}^L(t)} \\ c_{j_p}^R(t+1) &= c_{j_p}^R(t) + \eta \left\{ \begin{array}{ll} \underline{\delta}_{j_p}(t) & \text{if } x_p \leq 1/2 (c_{j_p}^L + c_{j_p}^R) \\ \bar{\delta}_{j_p}(t) & \text{if } x_p > c_{j_p}^R \\ 0 & \text{otherwise} \end{array} \right\} \frac{\partial f_{o_{j_p}}(t)}{\partial c_{j_p}^R(t)} \end{aligned} \quad (\text{B.13})$$

where $\eta > 0$ is a learning constant.

Appendix C

Publication List

Journals

Sau Wai Tung, Chai Quek and Cuntai Guan. eT2FIS: An evolving Type-2 neural fuzzy inference system. *Information Sciences*, in press.

Sau Wai Tung, Chai Quek and Cuntai Guan. SoHyFIS-Yager: A self-organizing Yager based hybrid neural fuzzy inference system. *Expert Systems with Applications*, in press.

Sau Wai Tung, Chai Quek and Cuntai Guan. SaFIN: A self-adaptive fuzzy inference network. *IEEE Transactions on Neural Networks*, vol. 22, no. 12, pp. 1928–1940, Dec. 2011.

Conferences

Sau Wai Tung, Chai Quek and Cuntai Guan. (2008). HyFIS-Yager-gDIC: A self-organizing hybrid neural fuzzy inference system realizing Yager inference. *International Conference on Neural Information Processing of the Asia-Pacific Neural Network Assembly (ICONIP)*, 1137–1145.

Sau Wai Tung, Chai Quek and Cuntai Guan. (2009). T2-HyFIS-Yager: Type-2 hybrid neural fuzzy inference system realizing Yager inference. *IEEE Conference on Fuzzy Systems* (FUZZ-IEEE), 80–85.

Sau Wai Tung, Chai Quek and Cuntai Guan. (2010). An evolving Type-2 neural fuzzy inference system. *Pacific Rim International Conference on Trends in Artificial Intelligence* (PRICAI), 535–546.

Sau Wai Tung, Chai Quek and Cuntai Guan. (2011). FAPOP: Feature analysis enhanced pseudo outer-product fuzzy rule identification system. *IEEE Conference on Fuzzy Systems* (FUZZ-IEEE), 1530–1537. (Best paper nomination, Student travel grant awarded)