A hybrid intelligent system: genetic algorithm and rough set incorporated neural fuzzy inference network

Wang, Di

2014

Wang, D. (2014). A hybrid intelligent system : genetic algorithm and rough set incorporated neural fuzzy inference network. Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/61614>

<https://doi.org/10.32657/10356/61614>

NANYANG
TECHNOLOGICAL
UNIVERSITY

**A HYBRID INTELLIGENT
SYSTEM: GENETIC ALGORITHM
AND ROUGH SET
INCORPORATED NEURAL
FUZZY INFERENCE NETWORK**

WANG DI
SCHOOL OF COMPUTER ENGINEERING
2014

**A HYBRID INTELLIGENT
SYSTEM: GENETIC ALGORITHM
AND ROUGH SET
INCORPORATED NEURAL
FUZZY INFERENCE NETWORK**

WANG DI

School of Computer Engineering

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

2014

Abstract

Neural Fuzzy Inference System (NFIS) has been intensively investigated due to its aptitudes in accurate data processing as well as extractable and human interpretable inference rule base. However, most NFIS architectures focus primarily on modeling accuracy. Lesser attention has been devoted to improve the interpretability of the fuzzy inference rule base. In this thesis, a high-level interpretability of the rule base is manifested through the use of a small number of features and a small number of simple rules without redundancies and inconsistencies. Furthermore, a small number of fuzzy membership functions without much overlap between the adjacent ones promote greater legibility. With a high-level accuracy and better interpretability, intelligent systems become more convincing to developers, experts, and users. However, many established methods that perform knowledge reduction are applied to the derived rule base. Lesser attention has been devoted to derive a compact fuzzy inference rule base from the initial design and during an iterative optimization process. This thesis aims to construct an NFIS that achieves a high-level interpretability with competitive accuracy in an integrated framework focusing on leveraging the trade-off between accuracy and interpretability from the design phase and throughout the iterative optimization process. To realize this objective, rough set theory is incorporated to perform knowledge reduction and genetic algorithm is incorporated to search for pseudo optimal solutions.

A hybrid architecture, named Genetic Algorithm and Rough Set Incorporated Neural Fuzzy Inference System (GARSINFIS), is proposed in this thesis. GARSINFIS employs Genetic Algorithm based Rough Set

Clustering (GARSC) technique, which systematically divides data into clusters and represents them using minimal amount of necessary knowledge. GARSC ensures a relatively small network size of GARSINFIS. The trade-off between interpretability and accuracy can be tuned by adjusting the coefficients of different components defined in the fitness evaluation function. Moreover, the Recursive Least Squares (RLS) algorithm, which boosts accuracy but degrades interpretability, can be applied to increase the order of the employed fuzzy rules and further fine-tunes them after the network structure has been automatically determined according to the clustering result. A committee of multiple GARSINFIS networks with the weighted voting scheme can be employed to increase the overall accuracy. GARSINFIS is applied to eight publicly available and widely used data sets for performance comparisons against other classical and similar models. Furthermore, it is applied to two real-world applications for performance evaluations, wherein the automatically selected input features and systematically derived inference rules are compared to the ones used by the experts to demonstrate how well GARSINFIS performs in terms of knowledge reduction without prior knowledge and human intervention.

Acknowledgments

I hereby wish to express my greatest gratitude and indebtedness to my supervisor, Dr. Quek Hiok Chai, and my former supervisor, Dr. Ng Geok See, who is now with Singapore University of Technology and Design. Dr. Quek offered me plenty of invaluable advices and sincere encouragements. His incisive analyses and insightful knowledge have become critical assistances to me, especially when I met difficulties. His enthusiasm in research will keep on influencing me in my future career. Dr. Ng introduced the research world to me in the first place. He provided me countless suggestions and continuous supervisions. His careful attitude in research will keep on benefitting me in the future. They both gave me their unselfish help and everlasting support. Without them, I would not be able to finish my doctoral research works as well as this thesis.

I also would like to express my sincere gratefulness and appreciation to Dr. Tan Ah Hwee, who is the principal investigator of the project that I am currently working on. Although the current research project is irrelevant to this thesis, I also would like to thank him in here for his guidance and support in my academic career.

Special gratitude also goes to my senior and fellow students: Tung Whye Loon, Gao Shi Yu, Ma Li and Liu Feng, for creating such a wonderful and pleasant research environment. Their generous help and the extensive discussions with them ease my choices of research directions and resolve various problems along the way.

I owe my deepest appreciation to my parents and my wife who delivered their love, support, and encouragements throughout my research life. Without them, I would not have persevered to the end.

Table of Contents

Abstract	i
Acknowledgments	iii
Table of Contents	iv
List of Figures	ix
List of Tables	xi
List of Abbreviations	xiv
1 Introduction	1
1.1 Background	1
1.2 Motivations	2
1.2.1 Why Neural Fuzzy Inference System?	2
1.2.2 Why Clustering Technique?	5
1.2.3 Why Interpretability Important?	6
1.3 Objectives	8
1.4 Organization of Thesis	10
2 Literature Review	12
2.1 Neural Networks	13
2.1.1 Bio-inspired Synthetic Models	13
2.1.2 Multi-Layer Perceptron Network	15
2.1.3 Radial Basis Function Network	17
2.2 Fuzzy Systems	19
2.2.1 Modeling Uncertainties in the World	19
2.2.2 Fuzzy Sets	19
2.2.3 Fuzzy Rules	20

2.2.4	Fuzzy Inference Systems	22
2.3	Neural Fuzzy Inference Systems	24
2.3.1	Hybrid Intelligent Systems	24
2.3.2	Knowledge Acquisition	25
2.3.3	Knowledge Reduction	27
2.3.4	Trade-off Between Accuracy and Interpretability . .	29
2.3.5	Review of Existing Models	30
2.4	Clustering Techniques	32
2.4.1	Grouping Similar Ones Together	32
2.4.2	Clustering Techniques Differ	33
2.4.3	Review of Existing Clustering Techniques	36
2.5	Genetic Algorithms	38
2.5.1	Robust Searching Algorithms	38
2.5.2	Genetic Algorithm Strategies	39
2.5.3	Species and Niches	45
2.5.4	Applications of Genetic Algorithms	45
2.6	Rough Sets	47
2.6.1	Modeling Vagueness in the World	47
2.6.2	Knowledge Representation System	47
2.6.3	Indiscernible Relation and Approximations	48
2.6.4	Decision Table	49
2.6.5	Attribute Reduction	50
2.6.6	Decision Rule Reduction	51
2.6.7	Applications of Rough Set Theory	51
2.6.8	Chronology of Rough Set Theory	52
2.7	Summary	53
3	GARSC: Systematic Clustering Technique	55
3.1	Technique Overview	56
3.2	A Systematic Knowledge Reduction Clustering Technique .	57
3.2.1	Overall Clustering Process	57

3.2.2	Constraints on Discretization	60
3.2.3	Decision Table Reduction	61
3.2.4	Removal of Inconsistent Rules	62
3.2.5	General Parameters and Strategies of Genetic Algorithm	64
3.2.6	Fitness Evaluation Function	65
3.2.7	Tournament Selection Strategy	71
3.2.8	Modified Uniform Crossover Operator	73
3.2.9	Modified Mutation Operators	75
3.3	Summary	76
4	GARSINFIS: Self-organizing Neural Fuzzy Inference System	78
4.1	GARSINFIS Architecture	79
4.2	Input Layer	81
4.3	Condition Layer	82
4.4	Rule-base Layer	84
4.5	Normalization Layer	85
4.6	Consequence Layer	86
4.7	Output Layer	90
4.8	Summary	91
5	Experimental Results and Analyses	93
5.1	Miscellaneous Experimental Setups	94
5.1.1	Design of Experiments	94
5.1.2	Control Parameters	96
5.1.3	Evaluation of Results	97
5.2	Necessity of Translation from Crisp to Fuzzy	100
5.2.1	Iris Classification Data Set	101
5.2.2	Wine Recognition Data Set	110
5.3	Accuracy Increase without Sacrificing Interpretability	115
5.3.1	Thyroid Gland Disease Diagnosis Data Set	115

5.3.2	Ionosphere Detection Data set	119
5.4	Accuracy Increase in Complex Application Domains	123
5.4.1	Glass Identification Data Set	124
5.4.2	Material Discrimination Data Set	127
5.5	Accuracy Increase by Formation of a Committee	130
5.5.1	Diabetes Diagnosis Data Set	131
5.5.2	Colon Cancer Diagnosis Data Set	134
5.6	Results Analyses on all Data Sets	138
5.7	Summary	144
6	Real-World Applications	148
6.1	Ovarian Cancer Diagnosis System	149
6.1.1	Overview of Ovarian Cancer Diagnosis Application	149
6.1.2	Experimental Setups	150
6.1.3	Diagnosis-1: Normal vs. Abnormal	152
6.1.4	Diagnosis-2: Benign vs. Malignant	156
6.1.5	Diagnosis-3: Borderline vs. beyond Borderline . . .	158
6.1.6	Diagnosis-4: Early Stage vs. Late Stage	161
6.1.7	Summary on Ovarian Cancer Diagnosis Application	163
6.2	Bank Failure Prediction System	165
6.2.1	Overview of Bank Failure Prediction Application .	165
6.2.2	Experimental Setups	166
6.2.3	Scenario-1: Last Year Available	168
6.2.4	Scenario-2: One Year Prior	172
6.2.5	Scenario-3: Two Year Prior	173
6.2.6	Summary on Bank Faliure Prediction Application .	175
6.3	Summary	177
7	Conclusion and Future Work	178
7.1	Conclusion	178
7.2	Future Work	184

List of Publications	187
References	190

List of Figures

2-1	Multi-layer perceptron network architecture	16
2-2	Radial basis function network architecture	18
2-3	Example of fuzzy linguistic labels	20
2-4	Example of fuzzy reasoning process	23
2-5	Fuzzy inference system architecture	24
2-6	Trade-off between accuracy and interpretability	30
2-7	State diagram of generic algorithms	39
2-8	Milestones of rough set theory	52
3-1	Illustration of knowledge transfer from crisp to fuzzy . . .	59
3-2	Flowchart of genetic algorithm based rough set clustering process	60
3-3	Illustration of modified uniform crossover operator	74
4-1	GARSINFIS Architecture	80
5-1	Plot of iris classification data set on two distinguishable fea- tures	102
5-2	Derived fuzzy membership functions by GARSC	104
5-3	Derived fuzzy membership functions by ECM	105
5-4	Theoretical and actual mutation rates of iris classification data set	107
5-5	Theoretical and actual mutation rates of wine recognition data set	112
5-6	Visualization of relative level of generalization	140

5-7	Visualization of relative accuracy on the whole data set . .	141
5-8	Visualization of relative accuracy on the testing data set .	142
5-9	Visualization of relative number of employed rules	143
5-10	Visualization of relative number of selected features	144
6-1	Ovarian cancer diagnosis system	151
6-2	Derived fuzzy membership functions of ovarian cancer diagnosis- 1 data set	153
6-3	Visualization of accuracy on ovarian cancer diagnosis appli- cation	164
6-4	Derived fuzzy membership functions of bank failure predic- tion scenario-1 data set	169
7-1	Overview of research areas studied in this thesis	179
7-2	Illustration of the iterative process of GARSC technique .	182

List of Tables

2-1	Comparisons of some existing neural fuzzy inference systems	31
2-2	Comparisons of some existing clustering techniques	36
2-3	Summary of genetic algorithm strategies	44
2-4	Example of knowledge representation system	48
2-5	Example of decision table	49
2-6	Example of decision table with attributes reduct	51
2-7	Example of decision table with attributes and rules reduct	51
3-1	The Proposed GARSC Algorithm	77
5-1	Summary of different configurations of GARSINFIS	95
5-2	Summary of applying GARSINFIS on iris classification data set	102
5-3	Derived fuzzy rules of iris classification data set	104
5-4	Derived fuzzy rules if rough set theory is excluded	105
5-5	Benchmarks on iris classification data set	108
5-6	Derived crisp rules of iris classification data set	108
5-7	Input features of wine recognition data set	110
5-8	Summary of applying GARSINFIS on wine recognition data set	110
5-9	Benchmarks on wine recognition data set	112
5-10	Summary of applying GARSINFIS on thyroid diagnosis data set	116
5-11	Benchmarks on thyroid diagnosis data set	117

5-12	Summary of applying GARSINFIS on ionosphere detection data set	120
5-13	Benchmarks on ionosphere detection data set	121
5-14	Distribution of data samples in glass identification data set	124
5-15	Summary of applying GARSINFIS on glass identification data set	125
5-16	Benchmarks on glass identification data set	126
5-17	Summary of applying GARSINFIS on material discrimination data set	128
5-18	Benchmarks on material discrimination data set	129
5-19	Input features of diabetes diagnosis data set	132
5-20	Summary of applying GARSINFIS committee on diabetes diagnosis data set	133
5-21	Benchmarks on diabetes diagnosis data set	133
5-22	Summary of applying GARSINFIS committee on colon cancer diagnosis data set	135
5-23	Benchmarks on colon cancer diagnosis data set	136
5-24	Summary of applying GARSINFIS on various data sets . .	138
6-1	Input features of ovarian cancer diagnosis data set	150
6-2	Summary of applying GARSINFIS on ovarian cancer diagnosis-1 data set	152
6-3	Derived fuzzy rules of ovarian cancer diagnosis-1 data set .	153
6-4	Benchmarks on ovarian cancer diagnosis-1 data set	155
6-5	Summary of applying GARSINFIS on ovarian cancer diagnosis-2 data set	156
6-6	Benchmarks on ovarian cancer diagnosis-2 data set	157
6-7	Summary of applying GARSINFIS on ovarian cancer diagnosis-3 data set	159
6-8	Benchmarks on ovarian cancer diagnosis-3 data set	160
6-9	Summary of applying GARSINFIS on ovarian cancer diagnosis-4 data set	161

6-10	Benchmarks on ovarian cancer diagnosis-4 data set	162
6-11	Summary of selected features on ovarian cancer diagnosis data set	163
6-12	Accumulated accuracy of different ovarian cancer diagnosis systems	163
6-13	Input features of bank failure prediction data set	166
6-14	Distributions of data samples in bank failure prediction data set	167
6-15	Summary of applying GARSINFIS on bank failure predic- tion scenario-1 data set	168
6-16	Derived fuzzy rules of bank failure prediction scenario-1 data set	169
6-17	Benchmarks on bank failure prediction scenario-1 data set	170
6-18	Summary of applying GARSINFIS on bank failure predic- tion scenario-2 data set	172
6-19	Benchmarks on bank failure prediction scenario-2 data set	173
6-20	Summary of applying GARSINFIS on bank failure predic- tion scenario-3 data set	174
6-21	Benchmarks on bank failure prediction scenario-3 data set	175
6-22	Summary of selected features on bank failure prediction data set	176
7-1	Summaries on major accomplishments presented in this thesis	179

List of Abbreviations

AARS	Approximate Analogical Reasoning Schema
ANFIS	Adaptive Network-based Fuzzy Inference System
ANN	Artificial Neural networks
CRI	Compositional Rule of Inference
DENFIS	Dynamic Evolving Neural Fuzzy Inference System
DIC	Discrete Incremental Clustering
ECM	Evolving Clustering Method
FCM	Fuzzy C-Means
FKP	Fuzzy Kohonen Partitioning
FNN	Fuzzy Neural Network
GA	Genetic Algorithm
GARSC	Genetic Algorithm based Rough Set Clustering
GARSINFIS	Genetic Algorithm and Rough Set Incorporated Neural Fuzzy Inference System
GP	Genetic programming
KRS	Knowledge Representation System
LVQ	Learning Vector Quantization
MF	Membership Function
MLP	Multi-Layer Perceptron
NFIS	Neural Fuzzy Inference System
RBF	Radial Basis Function
RLS	Recursive Least Squares
TSK	Takagi-Sugeno-Kang
TVR	Truth-Value Restriction

Chapter 1

Introduction

Ideals are like stars: you will not succeed in touching them with your hands, but like the seafaring man on the ocean desert of waters, you choose them as your guides, and following them, you reach your destiny.

- Carl Schurz

1.1 Background

The world does not only consist of 0s and 1s, but rather it is full of uncertainties. Fuzzy sets, first introduced in [Zadeh 65], use membership functions to describe the degree of belongingness. Since then, a new research trend was opened and people were fascinated by. More than a decade ago, researchers became more focused on *soft computing*

[Zadeh 94, Jang 96], which describes a possible way to mimic human reasoning process and to imitate human tolerance for imprecise information. This framework has a number of major components such as fuzzy systems, neural networks, evolutionary algorithms, etc. Many successful Neural Fuzzy Inference Systems (NFISs) [Jang 93, Kim 99, Kasabov 01, Kasabov 02, Tung 02b, Ang 05, Angelov 05, Rong 06, Liu 07, Juang 07, Angelov 08, Han 10, Rong 11] have been proposed in the literature. These systems perform well in terms of accuracy, but few of them [Ang 05, Liu 07] also focus on the interpretability of the fuzzy inference rule base in terms of both feature selection and rule reduction. Only in recent years, researchers have started to emphasize on constructing human interpretable and manipulable fuzzy rules [Casillas 03, Paiva 04, Ishibuchi 07]. Zadeh made an influential presentation on “how to compute with words” [Zadeh 04] to illustrate the increasing need and trend for high level of both accuracy and interpretability in soft computing. Hence, novel NFIS architectures should not only have strong reasoning ability for correct operations, but also possess interpretable rule base that makes the inference process meaningful, comprehensible, and convincing to developers, experts, and users.

1.2 Motivations

1.2.1 Why Neural Fuzzy Inference System?

Neural networks [Bishop 95, Haykin 98] are information processing systems designed to model brain functions and to imitate the learning capabilities

of human beings. Neural networks are normally layered. Neurons are usually connected to other neurons in adjacent layers with specific directions. Upon receiving input, each neuron makes local computation and propagates the output to all connected neurons in the path. Learning in neural networks is achieved by adjusting synaptic weights between connected neurons according to feedback signals or supervised information (sometimes, instructions are directly given by the experts or theoretically estimated by the learning algorithm without actually receiving any feedbacks). Neural networks are powerful learning machines, but most of them suffer from the black-box phenomenon [Bernd 99] that the weights are numerals used for computations without semantic meanings. The derived knowledge can be neither extracted nor interpreted by human beings directly without necessary knowledge transformations. How to transform a standard feed-forward neural network into the corresponding fuzzy rule base is shown in [Kolman 05].

Fuzzy systems [Cox 99] exploit linguistic models rather than complex mathematical models. A fuzzy system is essentially a rule based system consisting of fuzzy rules in the form of “IF - THEN”. The arguments in the antecedent part of each rule are defined as linguistic variables based on fuzzy sets [Zadeh 75]. The consequent part of each rule consists of either a linguistic variable or certain numerical function. These linguistic rules can be easily represented and computed with fuzzy implications and compositions, or modified by fuzzy hedges [Bezdek 99]. In the inference process, the response of each rule is weighted according to its firing strength or degree of activation, with respect to the input and the centroid of aggregated

responses, is computed to be the overall output. There is no formal and systematic way of knowledge acquisition in conventional fuzzy systems. The rules are generally formulated and tuned by experts to produce desirable results [Lee 90]. Inevitably, the rules may be biased or inaccurate. It is impossible or too difficult to define and maintain an effective fuzzy system in a complex application domain purely by human efforts.

Neural Fuzzy Inference Systems (NFISs) [Lin 96a, Nauck 97], also commonly denoted as *Fuzzy Neural Networks* (FNNs), are hybrid intelligent systems which overcome the two drawbacks mentioned above. Moreover, NFISs maintain the advantages of learning capabilities of neural networks and transparency of fuzzy systems, where transparency is a measure of human linguistic interpretability of rules [Paiva 04]. NFISs usually employ clustering techniques to systematically formulate Membership Functions (MFs) in each dimension [Kasabov 02], or require a predefined number of MFs that are systematically tuned afterwards [Jang 93]. Therefore, the fuzzy rule base is objective with minimum human manipulations. Each layer of the network approximates a respective step of the fuzzy reasoning or learning process. Unlike the opaqueness of neural networks, the connectionist structure of NFIS is highly transparent as interpretable rules can be easily extracted. There are many successful NFISs working as expert systems [Kasabov 99, Kuo 01, Tung 04, Juang 07, Wong 09, Tu 12] in various application domains. However, there are still many research problems that require further investigations, such as how to formulate systematic knowledge acquisition so that the system can be more compact and accurate, how to manipulate the inference rule base so that the system can

adapt to dynamic changes, how to define the inference process so that the system can be more competent, etc. Researchers are still actively engaged in this research field to construct better systems to handle applications more efficiently and convincingly.

1.2.2 Why Clustering Technique?

Clustering [Xu 05] is a process of grouping all the data elements in a given data set into subsets, so that the data elements within the same subset are more similar to each other than to the data elements in other subsets. Clustering techniques are widely used for systematically learning from data [Baraldi 99]. Clustering results represent data elements in exclusive or inclusive groups and require further knowledge transformations. Rather than crisp methods, fuzzy representations are normally used to summarize the common characteristics of each cluster that the stability-plasticity dilemma [Grossberg 82] can be better solved and to generate semantically comprehensive representations. This overall knowledge acquisition process determines the performance of self-organizing NFIS models [Kasabov 02, Tung 02b, Wang 04, Angelov 05, Angelov 08, Han 10], because the quality of fuzzy labels and their associated fuzzy sets is directly related to the clustering technique used and the representation method applied. Moreover, poor correspondence of fuzzy sets to the nature of training data degrades the quality of the resulting inference rules and may eventually lead to inaccurate results. Appropriate clustering technique, which describes the nature of training data as precisely as possible, is essential

and critical to a successful self-organizing NFIS.

1.2.3 Why Interpretability Important?

Interpretability [Casillas 03] refers to the ability of a fuzzy model to *concisely* express behaviors of an inference system in a *comprehensible* way. All NFIS architectures aim to produce high level of modeling accuracy, but few [Ang 05, Liu 07] focus on interpretability (in terms of both feature selection and rule reduction) as well. There is always a trade-off between accuracy and interpretability, which is known as the dipole of knowledge fidelity. An accurate system is normally complex and an interpretable system is normally faulty. It is highly desirable to construct an ideal system that achieves a good balance between accuracy and interpretability [Ishibuchi 07]. In this thesis, characteristics of high level of interpretability of the rule base are described as follows:

- The number of features in use is small. This property is also known as applying *feature selection* [Gonzalez 01] during knowledge acquisition. Feature selection not only alleviates the curse of dimensionality problem [Eccles 04] but also makes sense to application domain experts that not all attributes in the data set are important.
- The number of rules in the inference rule base is small. A small rule base size makes the network size of NFIS small and speeds up the inference process. Intuitively, a large number of rules is difficult for comprehension and an over-complex model is not convincing to

application domain experts.

- The rules are simple without redundancies and inconsistencies. Feature selection excludes unimportant features. However, it is not necessarily true that all the selected features are important in every single rule. The removal of redundancies in the antecedent part of each rule further speeds up the inference process and may enhance accuracy since only the most fundamental or core knowledge, which is less sensitive to noisy inputs, is utilized. Rules with similar antecedent parts may have contradictory consequent parts. The removal of inconsistencies in the inference rule base is definitely required to preserve system integrity.
- The number of fuzzy membership functions defined in each dimension is small. The number of MFs is either systematically determined by the representation of clustering results or predefined based on expert knowledge. A large number of fuzzy MFs employed in NFIS violates the spirit of fuzzy logic that computation should be performed with words [Zadeh 96].
- Minimal overlaps between adjacent MFs are required for better linguistic variable assignments. Conventional fuzzy systems used in engineering control domains allow overlap between MFs to an exceedingly high extent [Lovato 10]. The large volume of overlap is to ensure a full coverage of the input space and to minimize chances of system failures. However, in less restrained application domains, less overlap makes linguistic labeling of variables easier and more comprehensible.

According to *Occam's Razor* [Zahalka 10], when all other things being equal, the simplest solution is the best. Therefore, higher level of interpretability with the same level of accuracy constructs better model. This thesis aims to construct an interpretable NFIS without compromising accuracy.

1.3 Objectives

This thesis aims to construct a novel neural fuzzy inference system which employs Takagi-Sugeno-Kang (TSK) type of fuzzy rules [Takagi 85, Sugeno 88]. This framework is named Genetic Algorithm and Rough Set Incorporated Neural Fuzzy Inference System or GARSINFIS in short. Zero-order TSK type of fuzzy rule is equivalent to Mamdani type of fuzzy rule [Mamdani 77] if the consequent part of Mamdani type of fuzzy rule is in singleton form. With employment of zero-order TSK type of fuzzy rules, GARSINFIS achieves high level of interpretability. First-order TSK type of fuzzy rules can be derived after the parameter tuning process (applying the Recursive Least Squares (RLS) algorithm). First-order GARSINFIS achieves higher level of accuracy but sacrifices interpretability because the rules are complicated for comprehension. Selection between zero-order and first-order model depends on user requirement and the complexity of the application domain. Since genetic algorithm is involved, there are plenty of pseudo optimal solutions available. These solutions may not be optimal and inevitably contain modeling biases. A committee [Kuncheva 04] of multiple GARSINFIS networks with the weighted voting scheme alleviates the

individual modeling biases and increases accuracy but increases rule redundancy and degrades interpretability. The committee is either local or global depends on whether the networks are trained on the same set of data or different sets of data. However, this improvement in accuracy is due to the employment of multiple models not the enhancement of GARSINFIS architecture.

To ensure that GARSINFIS employs a compact and interpretable inference rule base without redundancies and inconsistencies, a novel clustering technique is proposed to perform knowledge reduction. The technique is named Genetic Algorithm based Rough Set Clustering or GARSC in short. Based on rough set approximations, the original knowledge base can be greatly reduced without losing any essential information. Rough set theory only applies on categorical values and Genetic Algorithm (GA) is introduced to discretize the input space. Optimization of the discretization is performed by GA and knowledge reduction is performed by rough set approximations. Measurements of both accuracy and interpretability are included in the fitness evaluation function of GA. Therefore, trade-off between accuracy and interpretability can be easily tuned by adjusting the coefficients of different components defined in the fitness function. Moreover, other pursuits of high level of interpretability are also incorporated in the clustering process, e.g. maximum number of MFs allowable in each dimension and minimum distance between two adjacent separation boundaries.

In summary, GARSINFIS architecture possesses following characteristics:

- The size of the network structure and connections between neurons in adjacent layers are self-determined based on the systematically derived fuzzy inference rule base without human manipulation.
- The operations of neurons in each layer are carefully defined to provide correct and efficient fuzzy reasoning and learning foundations.
- Zero-order TSK type of fuzzy rules are extensible to first-order rules. This extension increases accuracy but decreases interpretability. The choice between zero-order and first-order models is based on user requirement or the complexity of the application domain.
- Systematic knowledge acquisition is performed. All redundant and contradictive knowledge is removed from the resultant inference rule base. This knowledge reduction process is automated without human intervention.
- It is easy to tune the trade-off between accuracy and interpretability by adjusting the coefficients of different components in one multi-objective function.
- Numerous satisfactory solutions are produced. A committee of multiple models with a weighted voting scheme is used to improve the overall accuracy.

1.4 Organization of Thesis

The thesis is organized into the following seven chapters.

Chapter 1 begins with a brief introduction on the research background, gives the motivations and objectives of the proposed research work, and ends with the organization of the thesis.

Chapter 2 presents the literature review. It covers the fundamental concepts related to the proposed research work, compares and analyzes the pros and cons of well-established work, and discusses what knowledge is essential to develop the proposed system.

Chapter 3 introduces the clustering technique employed by the proposed system. It illustrates the overall algorithm and presents details of each step.

Chapter 4 covers the architecture of the proposed system and its complete structure together with respective mathematical formulations.

Chapter 5 shows the experimental results of the proposed system on well-known and public downloadable data sets. Results are benchmarked against other well-established techniques as well as neural, neural fuzzy inference systems.

Chapter 6 applies the proposed system to two specific real-world applications, one is a medical diagnosis application and the other is a financial prediction application. Experimental results are not only assessed in terms of accuracy, but also compared to expert knowledge.

Chapter 7 gives conclusion on this thesis and provides some insights to the future improvements and extensions.

Chapter 2

Literature Review

*“If I have seen further than others,
it is by standing upon the shoulders of giants.”*

- Isaac Newton

This chapter presents the literature review. Section 2.1 briefly introduces neural networks and gives the details of two representative models. Section 2.2 describes the basic concepts of fuzzy set theory and fuzzy inference systems. Section 2.3 reviews neural fuzzy inference systems and states the importance of fuzzy knowledge base. Section 2.4 discusses clustering techniques with comparisons. Section 2.5 covers genetic algorithm with different applicable strategies. Section 2.6 presents rough set theory with detailed examples. Lastly, Section 2.7 summaries this chapter.

2.1 Neural Networks

2.1.1 Bio-inspired Synthetic Models

Artificial Neural networks (ANNs) [Bishop 95, Haykin 98] are information processing models inspired by the neuro-physiological fundamentals of human brains [Carpenter 88]. ANNs are essentially massive-connected and parallel-distributed synthetic systems capable of imitating human learning capabilities. The basic processing unit of ANN is termed neuron, which usually employs an activation function to compute the output according to the synaptic inputs received. The output then becomes input to connected neurons in the path until the final result is produced. Connections between neurons are generally assigned with weights, where knowledge is stored. Learning in neural networks normally means adjusting the connection weight matrices.

ANNs are employed in various application domains, such as pattern recognition [Nishikawa 11], function approximation [Gougam 08], objective optimization [Hu 06], etc. They are usually classified according to the following three aspects:

- (1) Fully connected [Wang 98] and partially connected networks [Kang 05] in terms of the connectionist structure.
- (2) Feed-forward [Hornik 89], feed-back [Frasconi 88], and recurrent networks [Ku 95] in terms of the signal flow direction.
- (3) Unsupervised [Barlow 89], reinforcement [Carpenter 91], and super-

vised networks [Hush 93] in terms of the learning paradigm.

Neural networks are powerful learning machines and their strengths come from their adaptive and fault-tolerant natures. Weights between connected neurons can be adjusted according to the learning signals and a neuron can be added to incorporate newly discovered knowledge. Due to the distributed nature of information storage, variation in a single neuron or in one of its connections does not drastically affect the overall performance. However, some neural networks have one or more limitations listed as follows:

- (1) Some are incomprehensible, which is also known as the black-box phenomenon [Bernd 99]. Usually, weights in neural networks are numerals used for computations without semantic meanings. The derived knowledge can be neither extracted nor interpreted by human beings directly without necessary knowledge transformations. How to transform a standard feed-forward neural network into the corresponding fuzzy rule base is shown in [Kolman 05].
- (2) Some cannot utilize *a priori* knowledge. Although knowledge incorporation is possible [Kolman 05, Teng 08], some neural networks still learn from scratch. They do not make use of available knowledge, such as clustering results, to shorten the training time or pre-determine the network structure before learning.
- (3) Some cannot optimize the network structures. Although self-organizing methodology is commonly adopted nowadays [Lu 98, Huang 04, Tan 10],

some neural networks still determine the network structures through trial-and-error or by making heuristic assumptions.

However, self-organizing neural fuzzy inference systems avoid the three drawbacks mentioned above and still enjoy the benefits of the connectionist structure of neural networks, e.g. fast computation property, parallel and distributed processing ability, and ready for tuning nature.

Two representative neural network models are reviewed in Sections 2.1.2 and 2.1.3 respectively. Their performance are benchmarked against the proposed neural fuzzy inference system in Chapters 5 and 6.

2.1.2 Multi-Layer Perceptron Network

Multi-Layer Perceptron (MLP) network is one of the most commonly used neural network architecture. A typical MLP network has three layers, which are the input layer, the hidden layer and the output layer in the sequence of signal flow. MLP network is a fully connected and feed-forward network, whose connection weights are estimated using supervised learning. Its architecture is shown in Figure 2-1.

The most commonly employed activation function in MLP networks is the logistic function $f_{logistic}$, which is defined by Equation 2-1. There are multiple nodes in the output layer of MLP.

$$f_{logistic}(x) = \frac{1}{1 + e^{-x}}, \quad (2-1)$$

Figure 2-1: Multi-layer perceptron network architecture

where x denotes the received synaptic input.

In terms of classification problems, the number of output neurons is set to the number of classes in the data set. The output y is computed by Equation 2-2.

$$y_k = f\left(\sum_j W_{kj}x_j + b_k\right), \quad (2-2)$$

where j denotes the j th input; k denotes the k th output; and b denotes the bias.

The most commonly applied learning algorithm in MLP network is back-propagation, which aims to minimize the total error function ε as defined in Equation 2-3.

$$\varepsilon(n) = \frac{1}{2} \sum_k (d_k(n) - y_k(n))^2, \quad (2-3)$$

where n denotes the n th data point and d denotes the target value.

The learning rule for weight update $w'_{ji}(n)$ is defined by Equation 2-4.

The learning algorithm determines the change in weight $\Delta w_{ji}(n)$, which follows the *gradient descent* algorithm, is defined by Equation 2-5.

$$w'_{ji}(n) = w_{ji}(n) + \Delta w_{ji}(n), \quad (2-4)$$

$$\Delta w_{ji}(n) = -\eta \frac{\partial \varepsilon(n)}{\partial w_{ji}(n)}, \quad (2-5)$$

where $w_{ji}(n)$ denotes the weight associated with the link from the i th neuron to the j th neuron and η denotes the learning rate.

2.1.3 Radial Basis Function Network

Radial Basis Function (RBF) network is another commonly used neural network [Lu 98, Sanner 92] due to its powerful approximation ability. RBF network usually consists of three layers, which are the input layer, the Gaussian layer and the output layer in the sequence of signal flow. A conventional RBF network is fully connected, feed-forward, and connection weights are updated using supervised learning algorithm. Its architecture is shown in Figure 2-2.

RBF network is one of the few neural networks that are able to apply *a priori* knowledge. Neurons in the middle layer employ radial basis functions, such as Gaussian function $f_{Gaussian}$ defined by Equation 2-6. Mean c and variance σ^2 in $f_{Gaussian}$ can be predetermined by the result of a clustering technique, such as K-means [Hartigan 79].

Figure 2-2: Radial basis function network architecture

$$f_{Gaussian}(x) = e^{-\frac{\|x - c_i\|^2}{2\sigma_i^2}}, \quad (2-6)$$

where c_i denotes the center of the i th cluster and σ_i denotes the standard deviation of the i th cluster.

There are many approaches to identify the structure of a RBF network. For example, after c_i and σ_i are determined after applying the K-means clustering algorithm, the gradient descent algorithm can be applied to fine-tune the parameters. Therefore, the output node employs first-order polynomial function f_{poly} as defined by Equation 2-7.

$$f_{poly}(p) = \beta_0 + \beta_1 p_1 + \cdots + \beta_m p_m + \cdots + \beta_M p_M, \quad (2-7)$$

where β_0 is called the intercept; β_m denotes the coefficient of p_m , which is computed by the m th neuron in the Gaussian layer; and M denotes the total number of neurons in the Gaussian layer, whose value is pre-determined before training.

2.2 Fuzzy Systems

2.2.1 Modeling Uncertainties in the World

The concept of fuzzy is first formulated in [Zadeh 65]. Fuzzy set uses membership function to describe the degree of belongingness in non-crisp value. Fuzzy systems [Cox 99] provide a proper way to deal with uncertainties and imprecision in the real world.

Values used in fuzzy and probability [Ross 85] are both in the $[0, 1]$ interval, but they differ in meaning. *Fuzziness* describes event ambiguity and measures the degree to which an event occurs. *Randomness*, on the other hand, describes the uncertainty of event occurrence and people can bet on it [Kosko 92].

Furthermore, fuzzy linguistic labels cannot stand along without references. For example, the concept of “later” for fast-phenomena engineers may be in terms of nanoseconds, but for palaeontologists it may be thousands of years [Kartalopoulos 96]. Therefore, whenever a fuzzy term is used, one should always look at it within the context and find a point of reference as well as a measuring unit.

2.2.2 Fuzzy Sets

Fuzzy set [Zadeh 75] describes the degree to which an element belongs to a set. The characteristic function of a fuzzy set can have any value in the

$[0, 1]$ interval, denoting the degree of membership of an element in a given set. Let X be a collection of elements and x be an element in X . Fuzzy set $\tilde{A}$ in X is defined as a set of ordered pairs, and is illustrated in Equation 2-8.

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in X\}, \quad (2-8)$$

where $\mu_{\tilde{A}}(x)$ denotes the membership function of x in $\tilde{A}$, which maps X to the membership space M .

When M consists of only 0 and 1, $\tilde{A}$ is identical to a crisp set.

Figure 2-3: Example of fuzzy linguistic labels

Figure 2-3 shows a fuzzification example to explain the use of membership functions. In this example, linguistic variable “age” has three linguistic labels named young, middle-aged and old, which are characterized by the trapezoidal membership functions $\mu_{young}(x)$, $\mu_{middle-aged}(x)$ and $\mu_{old}(x)$ respectively.

2.2.3 Fuzzy Rules

Fuzzy IF-THEN rules, also known as fuzzy implications, are used to define fuzzy inference relations and to capture the imprecise modes of rea-

soning in an uncertain environment. There are three types of fuzzy rule models: Mamdani [Mamdani 77], Takagi-Sugeno-Kang (TSK) [Takagi 85, Sugeno 88] and Tsukamoto [Tsukamoto 79]. These three types of rules have the same form in the antecedent parts but they differ in the consequent parts. The antecedent parts of the three models are all fuzzy. The consequent part of Mamdani model is fuzzy. The consequent part of TSK model is a polynomial function of input variables. The consequent part of Tsukamoto model is a monotonically non-increasing function. Tsukamoto model is not widely employed in the literature due to its limited strength constrained by the one-open-ended function defined in the consequent part. In this thesis, only Mamdani and TSK models are discussed. The decomposed Mamdani type of fuzzy rule is given in Equation 2-9. The first-order TSK type of fuzzy rule is given in Equation 2-10.

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{(1,i)} \text{ and } \cdots \text{ and } x_n \text{ is } A_{(n,i)} \text{ and } \cdots \text{ and } x_N \text{ is } A_{(N,i)}, \\ &\text{THEN } y_i \text{ is } B_i; \end{aligned} \quad (2-9)$$

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{(1,i)} \text{ and } \cdots \text{ and } x_n \text{ is } A_{(n,i)} \text{ and } \cdots \text{ and } x_N \text{ is } A_{(N,i)}, \\ &\text{THEN } y_i = \alpha_{(0,i)} + \alpha_{(1,i)}x_1 + \cdots + \alpha_{(n,i)}x_n + \cdots + \alpha_{(N,i)}x_N, \end{aligned} \quad (2-10)$$

where x_n denotes the n th input linguistic variable; $A_{(n,i)}$ denotes the fuzzy label defined in the i th rule on the n th input linguistic variable; N denotes the total number of linguistic variables; y_i denotes the output of the i th rule; B_i denotes the fuzzy label defined in the i th rule on the output linguistic variable; and $\alpha_{(n,i)}$ denotes the coefficient of x_n in the i th rule.

Mamdani type of fuzzy rule which uses singleton for simplicity [Wang 94]

is defined in Equation 2-11. It is functionally equivalent to zero-order TSK type of fuzzy rule as defined in Equation 2-12. These rule representations improves interpretability since they are simple to comprehend. However, first-order TSK type of fuzzy rule improves accuracy. The proposed system in this thesis employs TSK type of fuzzy rules which have three major advantages over Mamdani Model. TSK model is computationally efficient, guarantees continuity in the output space, and generates lesser number of rules than Mamdani model in general. Selection between zero-order and first-order TSK type of fuzzy rules depends on user requirement and the complexity of application domain.

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{(1,i)} \text{ and } \cdots \text{ and } x_n \text{ is } A_{(n,i)} \text{ and } \cdots \text{ and } x_N \text{ is } A_{(N,i)}, \\ &\text{THEN } y_i \text{ is } C_i; \end{aligned} \quad (2-11)$$

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_{(1,i)} \text{ and } \cdots \text{ and } x_n \text{ is } A_{(n,i)} \text{ and } \cdots \text{ and } x_N \text{ is } A_{(N,i)}, \\ &\text{THEN } y_i = \alpha_{(0,i)}, \end{aligned} \quad (2-12)$$

where C_i denotes the singleton defined in the i th rule and $\alpha_{(0,i)}$ denotes the constant defined in the i th rule.

2.2.4 Fuzzy Inference Systems

There are three main types of fuzzy inference schemes, namely Compositional Rule of Inference (CRI) [Zadeh 75], Truth-Value Restriction (TVR) [Baldwin 79, Baldwin 80], and Approximate Analogical Reasoning Schema (AARS) [Turksen 90]. Among the three, CRI method is most commonly

employed because it is conceptually easy to understand and mathematically easy to implement. The most commonly employed implication operator in CRI method is the *min* operator I_{min} [Mamdani 77] as defined in Equation 2-13. I_{min} is simple, but proven to be effective in many models [Ang 03, Tung 04, Wang 04]. An example of the fuzzy reasoning process, which employs CRI inference scheme with I_{min} operator, is illustrated in Figure 2-4. In this example, there are two input linguistic variables with linguistic labels defined by membership functions of triangular shape and two rules of TSK type.

$$I_{min}(\mu_A(x), \mu_B(y)) = \min(\mu_A(x), \mu_B(y)). \quad (2-13)$$

Figure 2-4: Example of fuzzy reasoning process

Fuzzy inference systems [Cherkassky 98] are also known as fuzzy-rule-based systems and fuzzy models. Each system consists of four modules, namely fuzzifier, fuzzy knowledge base, inference engine, and defuzzifier, as shown in Figure 2-5. If the system employs TSK type of fuzzy rules, defuzzifier is not required because the consequent part of TSK type of rule

is already a crisp function. This inference sequence is generic and can be applied to neural fuzzy inference systems.

Figure 2-5: Fuzzy inference system architecture

2.3 Neural Fuzzy Inference Systems

2.3.1 Hybrid Intelligent Systems

The objectives of fuzzy inference system and neural network are different. The former one attempts to capture human thinking and reasoning capabilities at cognitive level and the latter one attempts to mimic the learning mechanisms of human brains at biological level. An integration of the two models is highly desirable because the learning capabilities and the parallel structures of neural networks complement the limitations of fuzzy systems. Neural Fuzzy Inference system (NFIS) [Lin 96a, Abe 97, Nauck 97], also known as Fuzzy Neural Network (FNN), is a powerful model realized by layered neural network with respective fuzzy or non-fuzzy operations. Such a hybrid intelligent system provides strong fuzzy logic foundations and great learning capabilities. NFIS has been proven to be successful and are utilized in various real-world applications such as pattern recognition [Kasabov 99], decision support [Wong 09], value prediction [Kasabov 02],

failure warning [Tung 04], etc. There are special types of NFIS models proposed in the literature such as neural network consisting of fuzzy neurons [Pedrycz 93] and neural network utilizing fuzzy weights for knowledge storage [Hayashi 93]. However, most NFIS architectures are standard and those special models are not discussed in this thesis.

2.3.2 Knowledge Acquisition

Most NFIS models define a systematic way to represent the fuzzy knowledge base illustrated in Figure 2-5. Manipulations based on expert knowledge transformations in conventional fuzzy systems have become obsolete today because applications are more complex and more automated. There are two different ways to systematically formulate fuzzy knowledge. The first way requires *a priori* understanding of the data set, or learns heuristics through trial-and-error. The second way analyzes the data set using clustering techniques and formulates knowledge based on the derived results.

ANFIS network [Jang 93] requires the user to predefine the number of Membership Functions (MFs) in each dimension. Therefore, ANFIS constructs an initial rule base prior to training. The number of MFs does not vary but the parameters of MFs are tuned during training. It is difficult or impossible for ANFIS to define the optimal number of MFs in each dimension, especially when dimensionality of the problem increases. Another intuitive method is to partition the feature space into grids. In [Cordon 00] and [Kasabov 01], each dimension is partitioned into equally distributed intervals. The number of grids is heuristically determined and the intervals

may or may not be merged to formulate associated MFs. The intensions are simple and direct, but the assumption that data elements are equally distributed throughout the feature space is normally inaccurate.

Clustering techniques [Xu 05] are systematic ways to formulate MFs based on the clustering results. There are two different approaches under this framework. The first approach is to perform clustering in the high dimensional space and project the derived clusters into each dimension to form MFs. This approach normally requires normalization in the feature space so that the distance between data elements can be measured without biases. ECM clustering technique [Song 01] employed in DENFIS [Kasabov 02] is a typical example of this approach. Although clusters are well separated in the high dimensional space, their projections in certain dimensions may have a great deal of overlap which degrades the meanings of fuzzy labels. Extra computations [Juang 98] are required to handle this problem. The other approach to MF formulation is to group data elements with respect to each individual dimension. Discrete Incremental Clustering (DIC) technique [Tung 02a] employed in GenSoFNN [Tung 02b] is a typical example of this approach. In this way, correctness of fuzzy labels is ensured and interpretability of fuzzy rules is improved. However, some clustering techniques require *a priori* understandings of the data set. These techniques are not self-organizing when formulating fuzzy knowledge base. E.g. K-means clustering technique [Hartigan 79] requires a predefined number of clusters prior to training. In recent years, there are many research work focus on how to develop self-organizing neural fuzzy networks with sequentially presented data samples. In [Angelov 05, Angelov 08],

whenever a new data sample comes in, its scatter or spread measurement is computed according to each existing cluster. If certain criteria hold, it is added to the winner cluster. Otherwise, a new cluster is created. In [Rong 06, Han 10, Rong 11], although the exact term of cluster is not used, the number of rules increases in the same manner.

2.3.3 Knowledge Reduction

The quality of fuzzy knowledge base directly determines the performance of NFIS. All redundancies and inconsistencies in the knowledge base should be removed or fixed to speed up the inference process and to improve the overall accuracy. There are three main approaches proposed in the literature to refine the knowledge base and each of them is discussed in the following paragraphs respectively.

The increase of dimensionality of the input feature space makes NFIS more difficult to perform accurately. This problem is commonly known as the curse of dimensionality [Eccles 04]. In the pattern recognition research field, high dimensional data elements are often visualized in a low dimensional space. Other than the traditional principal component analysis method, in [Lepinat 07], a variation of multi-dimensional scaling method is proposed, which improves the similarity measurements among data elements with a weighting scheme and a symmetric handling approach. Furthermore, feature selection methods are usually employed to deal with high dimensional applications. Even for a relatively small number of features, reduction of linguistic variables makes the resultant knowledge base more

interpretable and possibly increases accuracy because only the most fundamental features, which are less sensitive to noisy inputs, are selected. Feature selection is either performed as a filter or wrapper mechanism before training or is incorporated within the model during training. RELIEF [Kira 92] and FOCUS [Almuallim 94] algorithms are two typical examples of filter based methods that a subset of input features are selected prior to training. Opposite to the filter mechanisms, a wrapper approach is proposed in [Wang 08], which selects important features through a class-dependent approach. Feature selection algorithms can also be embedded in the neural fuzzy models. The general idea is to remove those features that do not significantly contribute to increase the overall performance. There are many models proposed in the literature, which perform feature selection within the learning cycles [Jin 00, Chakraborty 04].

Fuzzy rules with similar antecedent parts may have different or even contradictory consequent parts. These rules are known as inconsistent rules. They may be created by ill-defined knowledge acquisition process or introduced when knowledge bases from different sources are combined. The existence of inconsistent rules puts NFIS in danger that the performance of NFIS becomes unpredictable and often degraded. To remove or fix rule inconsistencies, various methods are proposed in the literature [Lin 96b, Lin 96c, Lee 02]. Furthermore, redundant rules which do not significantly contribute to increase the overall performance are also removed [Yen 99, Chen 04].

According to psychology experts, human short-term memory has a forward memory span of approximately seven items plus or minus two

[Miller 56]. Furthermore, fewer words can be recalled when the words have longer spoken duration [Baddeley 75] and when their speech sounds are similar to each other [Conrad 64]. Therefore, the number of MFs in each dimension should be kept small so that the linguistic labels are easier to comprehend. Moreover, the number of arguments in the antecedent parts of rules should be kept small, which definitely should not exceed the magical number seven plus or minus two. Feature selection excludes unimportant features. However, it is not necessarily true that all the selected features are important in every single rule. One specific feature may be significantly more important in one rule, but it can be ignored in another rule. There is an increasing number of proposed work, which focus on attribute reduction [Gonzalez 01, Ang 05, Liu 07].

2.3.4 Trade-off Between Accuracy and Interpretability

The trade-off between accuracy and interpretability is illustrated in Figure 2-6. Accurate NFIS is often complicated but has low level of interpretability. On the other hand, interpretable NFIS is often simple but has high level of error. The ideal system is usually not available so that a satisfactory trade-off between the two contradictory objectives is the focus. Intuitively, there are two approaches. One is to improve interpretability without compromising accuracy and the other is to enhance accuracy without degrading interpretability. There are many methods proposed in the first approach, such as redundant rule removal [Setnes 01], rule aggregation

[Setnes 98], and rule transformation [Combs 98]. The second approach often involves fine-tuning of fuzzy rules [Espinosa 00]. Many research work that focus on the trade-off between accuracy and interpretability are collected in [Casillas 03].

Figure 2-6: Trade-off between accuracy and interpretability

2.3.5 Review of Existing Models

NFIS is introduced in the 1990s [Lin 91]. There has been a proliferation of research in this family over these years. Review of some existing models is available in [Mitra 00]. In this thesis, details of existing models are not given but some representative and well-defined ones are compared in Table 2-1.

Although many NFIS models self-determine their network structures and fuzzy rule bases, additional control parameters are still required. ECM [Song 01] clustering technique employed in DENFIS requires heuristic setting of a threshold value to define the maximum distance allowed. A clus-

Table 2-1: Comparisons of some existing neural fuzzy inference systems

Model	Reference	Type	Remarks	SD	AR	RR
ANFIS	[Jang 93]	TSK	structure predefined by user; tunes parameters only	no	no	no
DENFIS	[Kasabov 02]	TSK	forms clusters based on distance; selects best performing rules	yes	no	yes
eclass	[Angelov 08]	TSK	extension of e-TS [Angelov 05]; based on incremental clustering	yes	no	yes
EFuFNN	[Kasabov 01]	Mamdani	evolving network structure; equal partitions in feature space	yes	no	yes
ESAFIS	[Rong 11]	TSK	extension of SAFIS [Rong 06]; specialized in sequential learning	yes	no	yes
FALCON-ART	[Lin 97a]	Mamdani	employs fuzzy-ART clustering; merges MFs by alignment	yes	no	yes
GDFNN	[Wu 01]	TSK	extension of RBF network; requires number of clusters	no	no	yes
GP-FNN	[Han 10]	TSK	extension of RBF network; convergence is guaranteed	yes	no	yes
GenSoFNN	[Tung 02b]	Mamdani	generic self-organizing NFIS; employs crisp-ART clustering	yes	no	yes
RS-HeRR	[Liu 07]	Mamdani	merges MFs if much overlapped; performs knowledge reduction	yes	yes	yes
R-SONFIN	[Juang 99]	Mamdani	recurrent variation of [Juang 98]; TSK types in [Juang 02, Juang 09]	yes	no	no
RS-POPFNN	[Ang 05]	Mamdani	extension of POPFNN [Ang 03]; performs knowledge reduction	yes	yes	yes
SOTFN-SV	[Juang 07]	TSK	structure defined by clustering; rules tuned by SVM algorithm	yes	no	no

SD: self-determines structure and rule base; AR: performs attribute or feature selection; RR: performs rule reduction

ter is prohibited from expanding if its radius is larger than the threshold value. Similar requirements also apply to eclass, ESAFIS, FALCON-ART, GenSoFNN, and GP-FNN to preserve the integrity of the obtained rules. Additional constraints are required to partition the input feature space in EFuFNN, RS-HeRR, R-SONFIN, RS-POPFNN, and SOTFN-SV. However, these parameters are learning criteria defined to perform systematic knowledge acquisition. Models which employ these additional controls are still considered as self-determination of their network structures and fuzzy inference rule bases. Furthermore, it is obvious to observe from Table 2-1 that most NFIS models perform rule reduction or selection, but few perform attribute or feature selection. Due to the limitations of human memories [Miller 56], NFIS models should keep a small number of MFs in each dimension and a small number of arguments in the antecedent part of rule. Otherwise, the resultant system is difficult to comprehend for developers, experts, and users.

2.4 Clustering Techniques

2.4.1 Grouping Similar Ones Together

Clustering [Xu 05] is a process of grouping certain data elements in the given data set into subsets, so that data elements within the same subset are more similar to each other than to other data elements in other subsets. This effort is valid on the assumption, which is normally true [Jain 88], that natural clusters exist in the data set. In self-organizing NFIS models, clustering technique is significantly important because it directly affects the reliability and the expressiveness of the fuzzy knowledge base. Appropriate clustering technique, which describes the nature of the training data as precisely as possible, is essential and critical to a successful self-organizing NFIS. The overall process of a typical clustering technique involves the following five steps:

- (1) Pattern representation. This step refers to the information gathering of the given data set, such as the number of input features, the number of available samples, the number of different output classes, etc. Feature selection is encouraged to remove redundant knowledge. Feature transformation is sometimes performed to derive more salient information. However, it is not desirable because the resultant representation is difficult to comprehend.
- (2) Similarity measurement. This step defines the criteria to form clusters. Usually, distance between data elements is used, such as Euclidean distance and other transformed or normalized distance mea-

surements. However, some clustering techniques employ other deterministic information, such as rough set theory [Vinterbo 97], to determine whether and how the clusters should be formed.

- (3) Grouping process. This step defines how clusters could be formed based on the criteria defined in the previous step. Among various ways to form clusters, some commonly adopted ones are discussed and compared in Section 2.4.2.
- (4) Data abstraction. This step is necessary to summarize the characteristics of clusters. E.g. fuzzy membership functions could be used to describe data distributions.
- (5) Result evaluation. This step examines the validity of resultant clusters, usually by assessments on the experimental results. This process is more significant to those clustering techniques, which apply recursive learning.

2.4.2 Clustering Techniques Differ

Clustering techniques are proposed to solve different problems in various application domains, such as pattern recognition, data classification, information retrieval, function approximation, future trend prediction, system control, etc. Some techniques are similar to each other as they basically follow the same routine, while some of them are different as their fundamental concepts are distinguished. The following subsections review some representative clustering techniques, which differ in the grouping processes.

2.4.2.1 Overlapping of Clusters: Crisp or Fuzzy

A crisp clustering technique allocates each data element to a single cluster. A fuzzy one assigns each data element the degrees of membership belongingness to multiple clusters. A fuzzy clustering technique could be easily converted to a crisp one by assigning each data element to the cluster possesses the largest membership value. K-means clustering technique [Hartigan 79] and its variation, fuzzy C-means [Bezdek 81], are good representatives for crisp and fuzzy type of clustering techniques respectively.

2.4.2.2 Forming Process: Incremental or Partitioning

An incremental clustering technique starts with building the first cluster base on the first data element fed in. Subsequently, when the following data elements come, certain decisions are made either to include them into existing clusters or to create new clusters. On the contrary, a partitioning clustering technique initially treats the whole data set as one cluster and then divides it into several smaller ones. The two different types of techniques distinguish in approach as the first one incrementally forms larger clusters by taking in new data elements and the second one step by step partitions the whole data set into smaller clusters. ECM [Song 01] used in DENFIS [Kasabov 02] is a typical incremental clustering technique and various partitioning clustering techniques are discussed in [Elavarasi 11].

2.4.2.3 Additional Control: Unsupervised, Semi-supervised, or Supervised

In unsupervised clustering, the formation of clusters entirely depends on the natural distribution of the data set, because there is no extra information given. In semi-supervised clustering, a small set of labeled data is used to supervise the clustering process of the remaining data. In supervised clustering, the formation of clusters is instructed by certain given information or criteria. Unsupervised clustering techniques are widely used in various application domains [Meyerbase 07, Kulkarni 08, Bose 09]. Semi-supervised ones are primarily used for image segmentations [Nock 05]. Supervised ones are often used for pattern recognition purpose [Qu 04, Li 06].

2.4.2.4 Number of Clusters: Predefined or Auto-generated

Some clustering techniques require users to predefine the number of clusters before the formation of clusters, while other techniques do not need such kind of information. The former type of techniques, e.g. K-means and fuzzy C-means, often face problems such as pre-given cluster numbers being unable to optimally represent the natural distribution of the data set or selection of a bad initial state that may lead to an incorrect representation. The determination of control parameters for the latter type of techniques (such as ECM) requires extra attention. If not carefully examined, the number of clusters generated may be either too many as over-fitting occurs, or too few as under-fitting occurs. Heuristics are often acquired through trial-and-error.

2.4.3 Review of Existing Clustering Techniques

Clustering techniques are initially proposed for pattern recognition purposes and are now commonly adapted to systematically formulate fuzzy inference knowledge. Review of some existing clustering techniques is available in [Baraldi 99]. In this thesis, details of existing clustering techniques are not given but some representative and well-defined ones are compared in Table 2-2.

Table 2-2: Comparisons of some existing clustering techniques

Clustering Techniques	Reference	OBC	Forming process	Additional control	A prior knowledge	Learning mode	MF shape
DIC	[Tung 02a]	no	INC	thresholds	no	on-line	T
ECM	[Song 01]	no	INC	minimization	no	both	G
FCM	[Bezdek 81]	yes	N.A.	minimization	yes	batch	IRR
Fuzzy ART	[Carpenter 91]	no	INC	vigilances	no	on-line	T
FKP	[Ang 03]	no	INC	minimization	yes	batch	T
K-means	[Hartigan 79]	no	INC	minimization	yes	batch	G
LVQ	[Kohonen 82]	no	INC	unsupervised	yes	on-line	N.A.

OBC: Overlapping Between Clusters; INC: incremental; T: Trapezoidal; G: Gaussian; IRR: Irregular.

According to Table 2-2, most clustering techniques do not allow overlapping between clusters except FCM. Because FCM defines a degree of belongingness of each data element to every cluster, it does not form actual clusters but find centroids of clusters only. DIC and Fuzzy ART require additional control to define whether an existing cluster is allowed to expand and the other clustering techniques except LVQ perform minimization either on the objective functions or the value changes between iterations.

In the literature, there are several clustering methods utilize both Genetic Algorithm (GA) and rough set theory (technical details are introduced in the following two subsections), which are similar to the clustering algorithm proposed in this thesis. An unsupervised clustering method is proposed in [Lingras 01]. Its incorporated GA uses binary encoding to in-

dicating whether every gene belongs to the lower and upper approximations of each class. The fitness function combines the measurements of both precision and within-group-errors. The rough c-means clustering method is proposed in [Mitra 04]. The center of each cluster is computed according to the rough set relations and GA is applied to find optimal values for the coefficient and threshold parameters, which define the relations. The rough k-medoid method is proposed in [Peters 08], which is conceptually similar to the rough c-means method. However, extensions are added to the rough k-medoid method that different fitness functions can be applied for different purposes. The aforementioned three clustering methods only visualize the clustering results or report indexes on the quality measurements without further investigations. However, the clustering method proposed in this thesis is more advanced in terms of (1) the clustering result is used to construct inference rules for a complex system to evaluate the performance; (2) the evaluations are based on different objectives, which allows leveraging the balance between accuracy and interpretability within one fitness function; (3) the number of genes in one chromosome is bounded, but not restricted by a fixed number; (4) the total number of the clusters is also not fixed by a predefined number.

2.5 Genetic Algorithms

2.5.1 Robust Searching Algorithms

Genetic Algorithms (GAs), first formally introduced in [Holland 75], are in the field of evolutionary algorithms [Back 96]. The idea of GA is inspired by the survival of the fittest theory proposed by Charles Darwin. GAs are searching algorithms based on the mechanics of natural selection and natural genetics [Goldberg 89]. GA normally starts with a randomly initialized population, which consists of artificial creatures named chromosomes. Based on their fitness values, some of them are selected in pairs as parents and granted the opportunities to produce offspring by means of crossover operators. Subsequently, some survived chromosomes are randomly selected for mutation, which means their genes are to be varied. The chromosomes in the next generation are expected to perform better and the process goes on iteratively until any termination criterion is met. Although randomized, GA is not a random walk. It efficiently exploits historical information to speculate on new search points with expected improvements [Goldberg 89]. The state diagram in Figure 2-7 demonstrates how GA works in general. Different strategies applicable to GA are discussed in the following sections.

Figure 2-7: State diagram of generic algorithms

2.5.2 Genetic Algorithm Strategies

2.5.2.1 Coding Strategies

In general, there are three types of coding strategies to represent chromosomes, namely binary coding, gray coding, and real number coding strategies. Among all the three, binary coding strategy is the most canonical and widely applied, because it is the simplest one in terms of implementation. It utilizes binary strings of 0s and 1s to represent chromosomes. However, if the application is in the continuous space, this coding strategy may suffer from quantization errors. This type of error can only be ignored if resolution is small enough. Gray coding strategy is similar to binary coding

strategy in that they both use binary representations. The two strategies are interchangeable by applying appropriate transformations. However, gray coding is better in the sense that changing in one quantity of its represented value will result in changing only one bit of the string. It has been proven [Caruana 88] that gray coding strategy performs better in real-valued problems and does not perform worse in all cases with respect to binary coding strategy. Real number coding strategy actually performs no processing on data unless the number of significant digits is specified. In general, real numbers are directly used to represent chromosomes. Real number coding strategy performs best when solving complex real-valued problems because of two reasons that there is no overhead of transformation involved and the strategy helps to reduce the lengths of chromosomes. However, crossover and mutation operators of real number coding strategy are not as easy to be defined as those of binary and gray coding strategies. Many operators designed for real number coding are available in [Michalewicz 96]. However, the clustering method proposed in this thesis allows chromosomes vary in length. Customized operators are designed and introduced in Chapter 3.

2.5.2.2 Fitness Evaluation

Fitness function, which is probably the most important aspect in GA, has to be carefully defined. Because GA is application independent, the performance evaluation solely depends on the quality of the fitness function. In other words, fitness function essentially characterizes the ideal solution which the user attempts to search for. The function can be constructed

in various ways. However, minimizing error and maximizing accuracy are the most commonly used criteria. Furthermore, several objectives can be achieved if they are all defined in the fitness function. These algorithms are termed multi-objective GAs, which are on the center stage of current research [Korkmaz 08, Deep 11, Ko 11].

GA sometimes exhibits undesirable behaviors. E.g. highly fit individuals dominate the selection in the first few generations and this situation can get worse in the subsequent generations. This domination leads to a premature convergence of GA because the rest regions in the feature space are not explored. To solve this problem, fitness function has to be carefully defined in a way that performance of similar chromosomes can be differentiated. Furthermore, there are additional control strategies to ensure a fair selection. Fitness scaling [Kreinovich 93], which re-computes fitness values so that they can fit into a linear line, gives better chances to lesser fit chromosomes because gaps between dominate ones and lesser fit ones has been reduced.

2.5.2.3 Reproduction Strategies

There are four major reproduction strategies proposed in the literature and a detailed comparison is available in [Blickle 95]. Roulette wheel selection strategy mimics the mechanisms of roulette wheels placed in casinos or the Wheel of Fortune. Each chromosome is placed on one sector of the wheel and the area is proportional to its fitness value. This strategy selects n parents by spinning the wheel for n times. Stochastic universal selection

strategy utilizes the same wheel as that of the roulette wheel selection strategy, except that there is n markers placed on the wheel instead of one. So n parents are selected by spinning the wheel once. This strategy guarantees a minimum selection error, which is at most one [Goldberg 89]. Ranking selection strategy is similar to fitness scaling in terms of avoiding premature convergence in GA. In this strategy, ranks of performance are used for selection instead of the actual fitness values. Tournament selection strategy conducts competitions among candidates. When selection starts, at each time, m chromosomes are randomly chosen from the population and the one with the highest fitness value is selected to produce offspring. This process iterates until all parents are selected. Tournament selection strategy is widely employed [Deb 00, Coello 02, Sun 10] because it is relatively easy to implement, it avoids overhead computations, and the selection stress is easily controlled by assigning different values to m .

2.5.2.4 Crossover Operators

Crossover operators define ways to produce offspring when parents are selected. One-point crossover operator randomly selects a crossover site and exchanges the following portions of the parents to produce two offspring. Multi-point crossover operator is similar to one-point crossover operator except that multiple crossover sites are defined instead of one. Uniform crossover operator is considered as a variation of multi-point crossover operator. At each time, a binary string with the same length of parents is randomly constructed. This control string determines that in each bit position, from which parent should the child inherit the gene. Special crossover

operators, namely partially matched, ordered, and cycle crossover operators, are defined to solve schedule optimization applications such as the traveling salesman problem [Oliver 87]. In this application domain, relative order of genes is important and each gene value is constrained to only appear once in any chromosome. Crossover operators for conventional real number coding strategy, such as linear, convex, and direction-based crossover operators, are different from above. Because generally speaking, operators defined for binary strings (normally used to encode a real number) cannot be directly applied to real numbers. One-point, multi-point, and uniform crossover operators can only be applied to real numbers if the numbers are represented in the form of a vector with fixed length.

2.5.2.5 Mutation Operators

Mutation operator simulates the process of natural evolution of genes. The basic idea is to provide new information to the population that crossover cannot produce and to introduce a degree of uncertainty when creating offspring. The general way to perform mutation is to vary the value in each position of the chromosome with certain probability known as mutation rate. Although GA with mutations alone can converge to optimal solutions [Michalewicz 96], mutation rate should still be carefully assigned, because a small rate leads to a slow convergence and a large rate leads to unnecessary oscillations.

2.5.2.6 Population Replacement Strategies

It is not guaranteed that offspring are always better than their parents. Usually, a concept of elitism is used in the organization of new generation $P(t+1) = \mu \cup \lambda$, which consists of highly fit parents μ from last generation $P(t)$ and produced offspring λ .

2.5.2.7 Termination Criteria

GA terminates when any of the termination criteria is matched and reports a serial of feasible solutions. Typical termination criteria are defined as the predefined maximum number of generations is reached, performance does not vary over a predefined number of generations, and the current population of solutions is satisfactory enough to the user.

2.5.2.8 Summary of Genetic Algorithm Strategies

All the aforementioned strategies applicable to genetic algorithms are listed with their implications and influences in Table 2-3.

Table 2-3: Summary of genetic algorithm strategies

Category	Strategy	Remarks
coding	binary	simple; may suffer from quantization errors in continuous space
	gray	proven to be better than binary coding [Caruana 88]
	real number	no overhead of transformation; reduces lengths of chromosomes
fitness evaluation	single objective	either minimize error or maximize accuracy
	multi-objective	different objectives are evaluated at the same time
reproduction (selection)	roulette wheel	parents selected by spinning the roulette wheel for multiple times
	stochastic universal	parents selected by spinning the wheel with multiple markers once
	ranking	ranks are used instead of the actual fitness values
	tournament	relatively easy to implement; able to control the selection stress
crossover	for binary ones	including one-point, multi-point, and uniform operators
	for ordered ones	including partially matched, ordered, and cycle operators
	for real numbers	including linear, convex and direction-based operators
mutation	toggle values	mutation rate defines the probability of actual mutation
replacement	elitism	population consists of highly fit survivors and newly produced ones
termination	maximum limit	GA reaches the predefined maximum number of generations
	convergence	performance of GA reaches convergence
	acceptable result	GA reports satisfactory results

2.5.3 Species and Niches

GA is theoretically and practically proven [Goldberg 89] to possess robust searching capabilities. In each generation, chromosomes are evaluated and those with higher fitness values get better chances to be selected as parents. Therefore, in an iterative way, GA explores the problem space with preferences and produces better solutions. GA processes in a parallel and distributed way because each chromosome is a feasible solution. Unlike some other optimization techniques, which only report one best solution, GA provides a serial of pseudo optimal solutions known as *species* that support several optimal regions known as *niches* [Deb 89]. Species and niches are extremely useful to handle multi-objective optimization problems [Deb 01].

2.5.4 Applications of Genetic Algorithms

GA has been employed in various application domains because it does not require the search space to be continuous, differentiable, unimodal, or of any functional form. GA is more popular or widely known than the other evolutionary algorithms (such as simulated annealing [Kirkpatrick 83], particle swarm [Kennedy 95], ant-colony [Dorigo 96], and tabu search [Glover 97]), probably due to its easy-to-understand nature and simple-to-implement characteristic.

The most straightforward way to apply GA is to directly define the problem in the fitness function for optimization. In this way, each chromo-

some in GA represents a possible solution to the problem, which requires no additional transformations and computations. These applications include function optimization (normally a mathematical optimization problem with multiple solutions), schedule optimization (search for the shortest path to finish the assignment with predefined constraints), and efficiency maximization (utilize the given resources as much as possible with predefined constraints).

GA is often integrated into other models to dynamically control the learning process. Genetic programming (GP) [Koza 92] is a typical example of such usage. Each chromosome in GP is a piece of computer program, which requires specially designed crossover and mutation operators. The ultimate goal of GP is to construct an intelligent machine that evolves by itself. In modern computer game domain, GA is either applied to adjust the behavior of an individual by dynamically altering its actions [Louis 05] or applied to manage the collaboration of individuals in group units [Stanley 05, Lucas 06].

GA is a powerful tool to search for optimal parameter values used in other models, such as the parameters of a flower image classification model [Cho 05], parameters of support vector machine [Wu 07], parameters of a software evaluation system [Andrews 11], etc. GA is also employed in NFIS architectures for various purposes, such as evaluation of qualitative effects on stock market to formulate the knowledge base [Kuo 01], parameter tuning of membership functions and connection weights [Wang 03], identification of the least important rules for pruning [Leng 06], etc.

2.6 Rough Sets

2.6.1 Modeling Vagueness in the World

Rough sets, first proposed in [Pawlak 82], are always compared to fuzzy sets since existence. These two theories are easily confused by people who fail to recognize the real differences between them [Lin 97b]. To describe belongingness, fuzzy sets use straightforwardly defined membership functions, while rough sets use relative relations named the lower and upper approximations. Both theories aim to achieve the same type of goal [Yao 98]. However, it is always better to have them both in one system to take advantages of their complements [Pawlak 91].

2.6.2 Knowledge Representation System

Rough set is initiated to model relations in a given data set or knowledge base [Pawlak 82]. It is a formally defined methodology that can be employed to reduce the dimensionality of given data set [Shen 01], proceeding to train an inference system.

Before performing knowledge reduction using rough set theory, first of all, the knowledge base should be formalized in the form of Knowledge Representation System (KRS). To better express and handle the relations in KRS, decision logic language (consisting of carefully defined mathematical representations) is introduced [Pawlak 82]. KRS is represented as $S = (U, A)$, where U is a nonempty and finite set denotes the universe of

discourse and A is a nonempty and finite set of primitive attributes. Table 2-4 shows an example of a KRS, where U has seven elements and A has five primitive attributes. Each number (0, 1, and 2) shown in Table 2-4 represents the attribute (column) value of its corresponding element (row).

Table 2-4: Example of knowledge representation system

U	a	b	c	d	e
1	1	0	0	1	1
2	1	0	0	0	1
3	0	0	0	0	0
4	1	1	0	1	0
5	1	1	0	2	2
6	2	1	0	2	2
7	2	2	2	2	2

2.6.3 Indiscernible Relation and Approximations

Indiscernible relation over knowledge K , denoted as $IND(K)$, is defined in Equation 2-14. The family of all equivalence classes of the equivalence relation $IND(K)$ is denoted as $U/IND(K)$. E.g. referring to Table 2-4, $U/IND(a) = \{\{3\}, \{1, 2, 4, 5\}, \{6, 7\}\}$.

$$IND(K) = \{(x, y) \in U^2 \mid \forall r \in K, r(x) = r(y)\}. \quad (2-14)$$

Rough set theory approximates knowledge using a pair of relational approximations. The lower and upper approximations of a set, when given an equivalence relation $IND(K)$, are defined in Equations 2-15 and 2-16 respectively. By presenting knowledge K , the lower approximation $\underline{K}Y$ is the set of elements which can be certainly classified by K and Y and the

upper approximation $\overline{K}Y$ is the set of elements which can be possibly classified by K and Y . E.g. referring to Table 2-4 and assume $Y = \{1, 2, 6, 7\}$, then $\underline{a}Y = \{6, 7\}$ and $\overline{a}Y = \{1, 2, 4, 5\} \cup \{6, 7\} = \{1, 2, 4, 5, 6, 7\}$.

$$\underline{K}Y = \bigcup \{X : X \in U/IND(K), X \subseteq Y, Y \subseteq U\}, \quad (2-15)$$

$$\overline{K}Y = \bigcup \{X : X \in U/IND(K), X \cap Y \neq \phi, X \subseteq Y, Y \subseteq U\}. \quad (2-16)$$

2.6.4 Decision Table

Decision tables can be defined in terms of KRS. Let $S = (U, A)$ be a KRS and $C, D \subset A$, where C denotes the condition attributes and D denotes the decision attributes. KRS with distinguished condition and decision attributes is termed a decision table, which is denoted as $T = \{U, C, D\}$ or CD-based decision table. E.g. referring to Table 2-4, if a, b, c , and d are the condition attributes and e is the decision attribute, a decision table is constructed and shown in Table 2-5. Because there are no conflict inference rules, the example decision table is consistent.

Table 2-5: Example of decision table

U	a	b	c	d		e
1	1	0	0	1	$\rightarrow$	1
2	1	0	0	0	$\rightarrow$	1
3	0	0	0	0	$\rightarrow$	0
4	1	1	0	1	$\rightarrow$	0
5	1	1	0	2	$\rightarrow$	2
6	2	1	0	2	$\rightarrow$	2
7	2	2	2	2	$\rightarrow$	2

2.6.5 Attribute Reduction

Rough set theory performs knowledge reduction with two fundamental concepts named reduct and core. Intuitively, a reduct of knowledge is an essential subset of knowledge which suffices to define all basic relations, whereas a core is the most fundamental subset of knowledge that consists of the common attributes of all reducts.

Indispensable relation is defined as follows. Given a decision table $T = \{U, C, D\}$, an attribute a is dispensable if and only if $IND(C) = IND(C - \{a\})$. Otherwise, a is indispensable. The family C is independent if $\forall a \in C$ is indispensable in C .

Based on indispensable relations, reduct and core are defined as follows. $Q \subseteq P$ is a reduct of P , if Q is independent and $IND(P) = IND(Q)$. The core of P is defined as the common parts of all reducts, i.e. $CORE(P) = \bigcap REDUCT(P)$. It is easy to infer that P can have many reducts, but only has one core.

A simplified decision table, which is generated by removing all dispensable attributes from Table 2-5, is shown in Table 2-6, where “x” denotes a do-not-care value. Please note that Table 2-6 is not the only feasible simplified table, because different choice of reduct constructs different decision table. It is obvious to observe from Table 2-6 that attribute c column is removed because the column consists of do-not-care values only. This removal of the whole column is often referred as feature selection.

Table 2-6: Example of decision table with attributes reduct

U	a	b	d		e
1	1	0	x	$\rightarrow$	1
2	1	0	x	$\rightarrow$	1
3	0	x	x	$\rightarrow$	0
4	x	1	1	$\rightarrow$	0
5	x	x	2	$\rightarrow$	2
6	x	x	2	$\rightarrow$	2
7	x	x	2	$\rightarrow$	2

2.6.6 Decision Rule Reduction

Each row in a decision table is essentially a decision rule. Every set of duplicate rules should be merged. A simplified decision table, which is generated by removing all duplicate rules from Table 2-6, is shown in Table 2-7. The finalized decision table consists of only four decision rules and rule number one is translated as IF $a = 1$ and $b = 0$ THEN $e = 1$.

Table 2-7: Example of decision table with attributes and rules reduct

U	a	b	d		e
1	1	0	x	$\rightarrow$	1
2	0	x	x	$\rightarrow$	0
3	x	1	1	$\rightarrow$	0
4	x	x	2	$\rightarrow$	2

2.6.7 Applications of Rough Set Theory

Rough set theory is a simple and efficient method to reduce unnecessary knowledge and to produce simplified decision rules. It has been applied in many real-world applications [Slowinski 92] such as financial analyses [Tay 02], medical diagnoses [Tsumoto 98], text categorizations [Chouchoulas 01], etc. Rough set theory is also integrated with other mod-

els, such as NFISs, to improve the overall performance.

One of the NFISs which incorporates rough set theory is proposed in [Banerjee 98]. The hybrid model is termed rough-fuzzy MLP. When learning starts, crude data sets are processed to construct decision tables. Subsequently, simple rules with good performance are selected. These crisp rules are further transferred into fuzzy rules and mapped into the hidden layer of MLP. This architecture is extended [Mitra 01, Pal 03] to integrate GA to tune the network structures and fuzzy membership functions.

Another NFIS approach which employs rough set theory is proposed in [Shen 02]. The proposed model employs rough set based attribute reduction algorithm to simplify fuzzy rules, which are derived based on the rule induction algorithm proposed in [Lozowski 96]. Further improvement is made in [Shen 04], where fuzzy-rough set [Pal 99] is introduced to improve the attribute reduction process.

2.6.8 Chronology of Rough Set Theory

Milestones of rough set theory research work are list in Figure 2-8 in chronological order.

Figure 2-8: Milestones of rough set theory

RS-POPFNN [Ang 05] is a hybrid intelligent system, which employs rough set theory to simplify fuzzy rules. It is an extension of POPFNN [Ang 03], whose fuzzy rule base is generated by the pseudo outer product rule identification algorithm [Quek 99]. By introducing rough set theory, network size of RS-POPFNN is significantly reduced but performance is improved. A similar architecture, named RS-HeRR, is proposed in [Liu 07]. The major difference is that RS-HeRR employs Hebbian-based rule reduction algorithm to merge fuzzy MFs and remove inconsistent rules prior to attribute reduction. The performance of the proposed NFIS is benchmarked against with RS-POPFNN and RS-HeRR in Chapter 5.

2.7 Summary

This chapter represents the literature review. Introductions of the reviewed knowledge, which are the foundations of this doctoral research work, are essential to understand the following chapters of this thesis. Neural networks and fuzzy systems are introduced prior to their hybrid models known as neural fuzzy inference systems. The hybrid model overcomes the disadvantages of the original models such as the opaqueness of neural networks and the lack of systematic way to determine the knowledge base in conventional fuzzy systems. More importantly, the hybrid model maintains the advantages of the original models in the way that carefully defined fuzzy reasoning and learning processes are realized in the layered neural network to perform fast and accurate inferences. Clustering techniques are usually required to construct the knowledge base for self-organizing NFISs. Ways

to form clusters differ but they all aim to describe the training data set as precisely as possible. Genetic algorithms possess robust searching capabilities. Unlike some other optimization algorithms which produce only one solution, genetic algorithms provide a serial of pseudo optimal solutions. Multiple solutions are especially useful to handle multi-objective optimization problems. Rough set theory is widely studied in the literature as a powerful tool to reduce the size of the knowledge base without losing essential information. This doctoral research work is proposed on the bases of all the aforementioned theories and techniques. The proposed system and its experimental results are reported in the following chapters.

Chapter 3

GARSC: Systematic Clustering Technique

“It’s not that I’m so smart, it’s just that I stay with problems longer.”

- Albert Einstein

This chapter introduces the proposed novel clustering technique termed Genetic Algorithm based Rough Set Clustering or GARSC in short. This novel clustering technique partitions data elements in each dimension to form clusters and subsequently derive associated fuzzy membership functions and rules. Genetic algorithm is incorporated to search for optimal or suboptimal separation boundaries and rough set theory is applied to simplify the decision knowledge base. The performance of the clustering results can be evaluated in terms of both accuracy and interpretability.

3.1 Technique Overview

It has been reviewed in Chapter 2 of this thesis that neural fuzzy inference systems utilize fuzzy rules to learn knowledge and to perform reasoning. Systems which self-organize their network structures normally employ clustering techniques to systematically form clusters on training data sets and to derive fuzzy rules based on the clustering results.

The proposed self-organizing neural fuzzy inference system, which is introduced in Chapter 4 of this thesis, applies the novel Genetic Algorithm based Rough Set Clustering (GARSC) technique. GARSC partitions data elements in each dimension to form clusters and subsequently derives associated fuzzy membership functions of Gaussian type. Because clustering is performed in each individual feature, no transformation and normalization is required and more importantly, semantic meanings of the assigned linguistic labels are preserved. As the name of GARSC reveals, it is a clustering technique which integrates genetic algorithm and rough set theory together. Genetic algorithm is applied to determine optimal or at least satisfactory suboptimal solutions. Rough set theory is incorporated to alleviate the curse of dimensionality problem [Eccles 04], which leads to the unnecessarily large network sizes of many established neural fuzzy inference systems. By applying rough set approximations, the original knowledge base is greatly reduced without losing essential information. This wonderful characteristic of rough set theory is extremely helpful to reduce the number of features used for reasoning, the number of rules in the rule base, and the number of arguments stated in each inference rule.

Therefore, the overall proposed system achieves a high level of interpretability without sacrificing accuracy. Nevertheless, rough set knowledge reduction methods are applicable to categorical values only. Continuous features have to be discretized before the derivation of the knowledge base. Genetic algorithm is applied to search for optimal or satisfactory suboptimal separation boundaries in each input feature. Therefore, when the iterative clustering process ends, all the derived membership functions are optimized according to the characteristics pre-determined by the users.

3.2 A Systematic Knowledge Reduction Clustering Technique

3.2.1 Overall Clustering Process

The GARSC process can be summarized as employing GA to determine optimal or satisfactory suboptimal sets of separation boundaries which discretize every continuous input dimension and applying rough set approximations to identify the minimum reduct of the derived knowledge base.

As long as the separation boundaries are determined on each feature, continuous data can be labeled with categorical values. After the removal of duplicate rows, conflicting rows, and dispensable conditional attributes, a reduct-decision table is eventually formed which contains only the most essential information to represent the given data set. Each set of different

separation boundaries derives a different decision table. Different decision tables may have the same form of representation such as the same number of rules and the same values used in each rule. However, these tables are still different because the same categorical value used in decision rules actually represents different sets of data elements which are determined by different sets of separation boundaries. This interpretation also applies to fuzzy inference rule base wherein fuzzy rules in the same form are represented with different membership functions. Each row in the decision table is an inference rule which represents the characteristics of the associated cluster. All data elements of that cluster fall into the category values defined in the rule in each individual dimension.

If each conditional attribute in the reduct-decision table is assigned to respective linguistic labels, fuzzy membership functions can be formulated based on the clustering result. Moreover, decision attribute used in the decision table can be directly assigned to the consequent part of the zero-order TSK type of fuzzy rule. In this way, crisp decision rules are easily translated into fuzzy inference rules. This concept is illustrated in Figure 3-1. The reason why Gaussian type fuzzy membership functions are utilized in our proposed system rather than other commonly adopted ones (such as triangular and trapezoidal) is that Gaussian function is continuous, which better represents the density of the cluster, and it only has two parameters (mean and standard deviation), which are easier to derive. Assume in dimension x , $n - 1$ separation boundaries are defined in the corresponding chromosome, then x is discretized into n regions $\{x_1, \dots, x_i, \dots, x_n\}$. Therefore, determining Gaussian membership function $f_{G_i}(x) = e^{-\frac{\|x - c_i\|^2}{2\sigma_i^2}}$

is actually computing the center c_i and the standard deviation σ_i of all the data points in x_i .

Figure 3-1: Illustration of knowledge transfer from crisp to fuzzy

This particular process of knowledge transfer from crisp to fuzzy cannot be omitted because the crispness of separation adopted in rough set theory does not tolerate overlapping. Fuzzy membership functions are employed to represent the clusters derived to deal with inexact information and unforeseen circumstances. This kind of knowledge transfer has a great advantage because it naturally prevents the fuzzy membership functions from overlapping or separating too much with adjacent ones, which is another important aspect of interpretability in fuzzy modeling.

Different fuzzy inference rule bases can be generated in the above described way according to different candidates produced in the current generation of the genetic algorithm. These rule bases are subsequently applied to the proposed neural fuzzy inference system for performance evaluations. Based on the computed fitness values, genetic algorithm iteratively creates new generations of candidates until termination. Figure 3-2 shows the flowchart of the overall clustering process.

Figure 3-2: Flowchart of genetic algorithm based rough set clustering process

3.2.2 Constraints on Discretization

In GARSC, an important constraint must be defined, which is the maximum number of separation boundaries allowable in each dimension. This constraint straightforwardly determines the maximum number of fuzzy membership functions allowable in each dimension. However, the actual number of fuzzy membership functions formulated in each dimension is also determined by the knowledge reduction process. The smallest number of separation boundaries actually in use is zero, which means information of that particular dimension is not included in the reduct-knowledge base. This constraint should not be set to a large value because employing a huge number of fuzzy membership functions degrades interpretability.

There is another constraint on the minimum distance between any pair

of adjacent separation boundaries in the same dimension, which is termed *mindis*. This constraint is imposed to make sure that the actual constructed fuzzy membership functions have high level of generalization such that any adjacent pair of them should not be merged into one. The minimum distance constraint *mindis* is defined by Equation 3-1. The *max* function in the denominator defines the level of generalization.

$$mindis_i = \frac{ub_i - lb_i}{\max(nop_i, M)}, \forall i \in C, \quad (3-1)$$

where i denotes the i th dimension; ub_i and lb_i denote the upper boundary and lower boundary of values in the i th dimension respectively; nop_i denotes the total number of possible separation boundaries in the i th dimension, which if not specifically stated otherwise, is assigned to the total number of different values in the i th dimension where corresponding values in the decision attribute are changed across; M denotes the user specified minimum number of separation boundaries in every dimension, which has a default value of 10; and C denotes the set of conditional attributes.

3.2.3 Decision Table Reduction

It has been discussed in Section 2.6.5 on Page 50 that any conditional attribute can be removed from the decision table if the attribute is dispensable. Given a decision table $T = \{U, C, D\}$, an attribute a is dispensable if and only if $IND(C) = IND(C - \{a\})$, where the indiscernible relation $IND(K)$ is defined in Equation 2-14 on Page 48. Otherwise, a is indispensable. After the removal of all dispensable attributes, the finalized

decision table is independent, which is actually one of the many possible reducts of the original decision table. Therefore, attribute reduction is performed during the process of constructing the independent decision table with minimum cardinality.

If for all the rules in the reduct-decision table, an attribute is always dispensable, then that attribute can be removed from the given data set. This process of excluding attributes which do not contribute to define the essential knowledge base is referred as feature selection. It is extremely helpful to solve problems in high dimensional domains.

Decision rule reduction is conceptually similar to attribute reduction. Other than merging each set of duplicate rules into one single rule, a decision rule in the rule set is dispensable if and only if the performance of the rule base does not decrease with the removal of that rule. This is also referred as the pruning process of decision rules.

3.2.4 Removal of Inconsistent Rules

How to remove inconsistent rules from the reduct-decision table is not covered in Section 2.6, because there are too many ways to define the selection criteria. The selection criteria adopted in GARSC are defined in this section.

Inconsistent rules, which are the rules with the same conditional attribute values but different decision attribute values, do often exist in real-world applications. To maintain the integrity of the inference knowledge

base, only one rule from each inconsistent rule set should be preserved. To determine which rule(s) should be preserved in the reduct-decision table, one way of computing the confidence of the k th rule is proposed in Equation 3-2. The *min* function is applied to penalize information incompatibility.

$$conf(k) = \min \left(\frac{card(U_i(k_i) \cap d_k)}{card(U_i(k_i))} \right), \forall i \in C, \quad (3-2)$$

where *card* denotes the function to compute cardinality; U_i denotes the function to generate a union of decision attributes of every rule in the decision table which shares the same value on the i th attribute; k_i denotes the value of the i th attribute of the k th rule; and d_k denotes the decision attribute of the k th rule.

Based on the introduction of the confidence evaluation function, the criteria to remove inconsistency rules from the reduct-decision table are defined as follows.

- (1) The rule with the maximum confidence value will be preserved in the decision table while all the other rules from the same inconsistent rule set will be removed.
- (2) If multiple rules are tied at the maximum confidence value in the same inconsistent rule set, the rule which covers the most number of data elements is preserved. This particular criterion is feasible because each rule actually represents a cluster of data elements. It is assumed that the largest cluster always wins because the majority party normally possesses the highest level of confidence.

- (3) If multiple rules still tie in this situation, a random rule will be selected as the winner with equal probability.

3.2.5 General Parameters and Strategies of Genetic Algorithm

In genetic algorithm, the population size defines the total number of chromosomes existing in each generation. Increasing the population size is equivalent to increasing the number of possible solutions. Therefore, more candidates will be examined, and a wider range of area in the universe of discourse will be explored.

In GARSC, real number coding strategy is employed to construct chromosomes. Each gene used in the chromosome represents a separation boundary in its respective dimension. Because GARSC technique only constrains the maximum number of partitions allowable in each dimension, hence, the actual number of partitions varies. Therefore, chromosomes in GARSC have different lengths. To deal with chromosomes in different lengths, a modified uniform crossover operator is proposed in Section 3.2.8.

In GARSC, elitism replacement strategy is applied. The elitism ratio $\mu \in [0, 1)$, defines how many percentage of highly fit individuals in the current generation $P(t)$ will be directly brought into the next generation $P(t + 1)$. Normally μ is assigned to a relatively small number to prevent the domination of species on local optimums.

The overall clustering process stops when the genetic algorithm reaches

the predefined number of generations, which should sufficiently ensure the convergence of the genetic algorithm. The total number of generations and the population size approximately define the total number of solutions for the genetic algorithm to evaluate. To minimize the number of revisits, other control parameters should be carefully assigned.

3.2.6 Fitness Evaluation Function

Fitness function is defined to evaluate the quality of each chromosome. It is probably the most important component in GA, because it directly evaluates the performance of each chromosome and intimately characterizes the ideal solution which the user attempts to search for. Because the intension of the proposed soft computing model is to produce highly interpretable rules without sacrificing accuracy, fitness function f is defined in Equation 3-3 where performance evaluations of both accuracy and interpretability are incorporated. This fitness function is to be minimized by GA.

$$f(x) = \underbrace{\tau_1(1-a)\frac{NOD}{K}}_1 + \underbrace{\tau_2\frac{nof}{NOF}}_2 + \underbrace{\tau_3\frac{nor}{NOD}}_3 + \underbrace{\tau_4\frac{noa}{NOF \cdot NOD}}_4 + \underbrace{\tau_5\frac{mse}{NOF}}_5, \quad (3-3)$$

where x denotes the chromosome to be evaluated; $\tau_1, \dots, \tau_5$ denote associated coefficient parameters for the user to specify; a denotes the accuracy of applying data set to the model derived using x ; NOD is the total number of data elements in the data set; K denotes a constant to systematically leverage between accuracy and interpretability which is defined by Equation 3-4; nof denotes the number of features actually used to construct the

model; NOF denotes the total number of input features in the data set; nor denotes the number of rules actually used to construct the model; noa denotes the total number of arguments in the antecedent part of all rules in the rule set; and mse denotes mean squared error computation which is defined by Equation 3-5.

$$K = \begin{cases} NOC, & \text{if } \ln(NOF) \leq NOC, \\ NOF, & \text{otherwise,} \end{cases} \quad (3-4)$$

where NOC denotes the number of different values used for the decision attribute.

Therefore K is a constant defined by NOF and NOC which is used as a scaling factor to leverage the importance between accuracy and interpretability.

$$mse = \frac{1}{NOD} \sum_{i=1}^{NOD} (y_i - \hat{y}_i)^2, \quad (3-5)$$

where y_i denotes the predicted value and $\hat{y}_i$ denotes the expected value.

In Equation 3-3, all constants are represented in capital letters and all variables are represented in small letters. The first and fifth terms of Equation 3-3 represent the correctness of the model because accuracy and mean squared error are concerned respectively. While, the rest of the three terms represent the interpretability of the model, corresponding to the actual number of features used, actual number of rules used, and the total number of arguments in the antecedent part of all rules in the rule set respectively. Equation 3-3 focuses on both accuracy and interpretability, but it can be easily transformed into the function which only focuses on

accuracy by simply assigning τ_2, τ_3 and τ_4 all to zero. The transformed fitness function is represented in Equation 3-6.

$$f(x) = \tau_1(1 - a)\frac{NOD}{K} + \tau_5\frac{mse}{NOF}. \quad (3-6)$$

Equation 3-6 only addresses the accuracy of the model but it does not mean interpretability is totally ignored. No matter how the fitness function is defined, the inference knowledge base has already been simplified by applying rough set approximations. In other words, although only accuracy is emphasized in Equation 3-6, the model under evaluation has already been simplified using rough set approximations. In complicated and accuracy-driven application domains, Equation 3-6 is preferred, but in simple and interpretability-driven application domains, Equation 3-3 is preferred to put greater emphasis on the interpretability of the constructed model. However, users are always welcomed to experiment on different values to coefficients $\tau_1, \dots, \tau_5$ to control the balance between accuracy and interpretability.

According to Equation 3-3, if the constructed model is more accurate and utilizes lesser amount of knowledge, then it has a lower fitness value and the chromosome used to construct the model performs better. The five terms in Equation 3-3 are placed in descending order of importance, assuming $\tau_1, \dots, \tau_5$ all are assigned equally.

The first term of Equation 3-3 can be expanded into Equation 3-7.

$$(1 - a)\frac{NOD}{K} = (1 - \frac{noc}{NOD})\frac{NOD}{K} = \frac{NOD - noc}{K}, \quad (3-7)$$

where noc denotes the number of correctly classified data elements.

If the number of correctly classified data elements decreases by one, i.e. $noc \rightarrow noc - 1$, then Equation 3-7 becomes Equation 3-8.

$$\frac{NOD - (noc - 1)}{K} = \frac{NOD - noc + 1}{K} = \frac{NOD - noc}{K} + \frac{1}{K}. \quad (3-8)$$

Comparing Equations 3-7 to 3-8, if all the other terms in Equation 3-3 remain unchanged, then fitness value is increased by $\frac{1}{K}$ which means a lesser fit of the chromosome.

If the second term of Equation 3-3 changes in the slightest amount, i.e. $nof \rightarrow nof + 1$, which means one more feature is used to construct the model, then the fitness value is increased by $\frac{1}{NOF}$, which also means a lesser fit of the chromosome. The derivation steps are similar to the transformation from Equations 3-7 to 3-8. Therefore, the derivation steps will not be repeated hereinafter.

From Equation 3-4, it is easy to imply that K is always smaller or equal to NOF . Therefore, $\frac{1}{K}$ is always greater or equal to $\frac{1}{NOF}$. In other words, in Equation 3-3, the effect of the slightest change in the first term is always greater or equal to the effect of the slightest change in the second term. Moreover, if $\ln(NO F) \leq NOC$, then $\frac{1}{K} = \frac{1}{NOC}$. This implies that the effect of the first term which represents the correctness of the model is more sensitive than the effect of the second term which represents interpretability of the model in terms of feature selection. When $\ln(NO F) > NOC$, the number of features in the given data set is much greater than the total

number of different decision attribute values, which is the number of classes for classification applications. In such situation, $\frac{1}{K} = \frac{1}{NOF}$. This implies that the two terms have the same amount of effect for slightest changes. In other words, the amount of effect on accuracy is greatly reduced to the same level of feature selection and the constructed model is no longer expected to fit perfectly on the given data set. Instead, more generalized models which utilize lesser amount of knowledge are preferred.

If one more rule is employed in the inference knowledge base, i.e. $nor \rightarrow nor + 1$, then the fitness value is increased by $\frac{1}{NOD}$. The number of dimensions in the given data set is normally smaller than the number of data elements, i.e. $NOF < NOD \Rightarrow \frac{1}{NOF} > \frac{1}{NOD}$, which implies that the effect of the slightest change in the second term of Equation 3-3 is greater than that of the third term of the same equation.

If one more argument is employed in the rule set, i.e. $noa \rightarrow noa + 1$, then the fitness value is increased by $\frac{1}{NOF \cdot NOD}$. Because $NOF > 1$ in all real-world application domains, the effect of the slightest change in the third term of Equation 3-3 is greater than that of the fourth term of the same equation.

If all the first four terms of Equation 3-3 remain the same, then the classification accuracy and interpretability of the constructed model do not vary. If there is a small change in the fifth term of the same equation, then the model is slightly lesser fit for the given data set, which leads to an increase in the mean squared error. By substituting Equation 3-5 into

the fifth term of Equation 3-3, Equation 3-9 is derived.

$$\frac{mse}{NOF} = \sum \frac{(y_i - \hat{y}_i)^2}{NOF \cdot NOD}. \quad (3-9)$$

According to Equation 3-9, the change in the sum of squared error has the same level of effect as the change in the number of arguments employed in the rule set because they have the same denominator. The slightest change in sum of squared error is assumed to be smaller than one, if only one predicted value among all the data elements is computed differently without changing the value of the first term of Equation 3-3. Therefore, the fourth term takes on a greater effect than that of the fifth term in a rather stable solution where predictions do not vary drastically.

Thus, the complete proof has been given that the amount of effect decreases along the five terms of Equation 3-3. The first term, which concerns about accuracy, guarantees the correctness of the constructed model. The second, third and fourth terms, which concern about interpretability, encourage the constructed model to achieve the same level of accuracy with lesser amount of information. According to Occam's razor that all other things being equal, the simplest solution is the best [Zahalka 10], a higher level of interpretability with the same level of accuracy constructs better model in terms of better generalization which prevents the model from overfitting. The fifth term, which concerns about the mean squared error, is employed to refine the constructed model to achieve a better fitting on the given data set as long as accuracy and the amount of information employed are kept unchanged. The statement that the five terms in Equation 3-3

are placed in descending order of importance is based on the assumption of $\tau_1, \dots, \tau_5$ all being assigned equally. The trade-off between accuracy and interpretability should vary for different application domains and different user requirements. Actually, although all the terms in Equation 3-3 are linearly combined, their coefficients can be assigned to any real numbers, even negative values. In the thesis, only 0 and 1 are used to illustrate how different values of the coefficients can affect the final results in general. However, any user can actually choose any values for each coefficient by heuristics or trial-and-error. If the user does not want to spend time on a relatively time-consuming sweep of parameter values, one can always choose 0 and 1, maybe -1 also, for simplicity, to control the objective of the fitness function towards the user's intension. Basically, based on user's initial setting of the parameters, if the obtained result is not satisfactory enough, one can always go further to emphasize certain terms by increasing their coefficient values or to penalize certain terms by decreasing their coefficient values.

3.2.7 Tournament Selection Strategy

To produce new chromosomes, parents with high fitness values are selected from the current generation to produce offspring in the next generation. Tournament selection strategy [Miller 96] is employed in GARSC for the following reasons: easy to implement; avoids overhead computations of fitness scaling; and the selection stress can be easily controlled by adjusting only the tournament size m and the selection probability p .

For each chromosome to be selected for crossover, m candidates are randomly selected from the current generation for consideration and they are sorted in descending order based on their fitness values. The selection process starts with the first candidate and the probability of selecting the n th candidate $p(n)$ is defined in Equation 3-10.

$$p(n) = p(1 - p)^{n-1}, \quad 0.5 < p \leq 1. \quad (3-10)$$

If the tournament size m is large, it is more stressful for lesser fit candidates to be selected. On the other hand, a small tournament size increases the probability of lesser fit candidates being selected as they are competing with a lesser number of the others. In general, a large value of m is used in simple and unimodal application domains to accelerate the converging process and a small value of m is used in complex and multimodal application domains to better explore the universe of discourse [Miller 96].

In the early generations of genetic algorithm, p should be set to a small value to give more chances to those lesser fit candidates to get selected. In this way, the search is prevented from premature convergence because more possible candidates are considered even if they have smaller fitness values. However, in the late generations, p should set to a large value, because only those highly fit candidates are expected to lead the selection towards the best solution in more indepth exploitation. This transition from exploration to exploitation is conceptually similar to the action selection policy defined in reinforcement learning models [Wang 09]. Tournament selection probability p in the proposed clustering technique GARSC is defined by

Equation 3-11.

$$p = 0.5 \left(1 + \frac{cgi}{NOG} \right), \quad (3-11)$$

where cgi denotes the current generation index and NOG denotes the maximum number of generations.

Because cgi is in the $[1, NOG]$ interval, p is equally distributed in the $[0.5 + \frac{0.5}{NOG}, 1]$ interval, which exactly fulfils the constraint defined in Equation 3-10.

3.2.8 Modified Uniform Crossover Operator

Because elitism replacement strategy is applied in the proposed Genetic Algorithm based Rough Set Clustering (GARSC) technique, the number of parents, who are selected to produce offspring in the next generation of the genetic algorithm, depends on the population size S and the elitism ratio μ which has been introduced in Section 3.2.5 on Page 64. The population in each new generation consists of μS number of elites surviving from the last generation and $S - \mu S$ number of newly produced offspring. Please note that a highly fit chromosome may be selected as a parent for multiple times. However, the domination is prevented by applying crossover and mutation operators.

When sufficient number of parents has been selected, the crossover operator is applied to each pair of parents to produce offspring. The probability that a pair of selected parents eventually mate is defined as the crossover rate. However, because elitism replacement strategy is applied in GARSC,

the crossover rate is set to one, i.e. every pair of parents is crossed over to produce offspring. Because chromosomes in GARSC consist of real number coded genes to represent sets of separation boundaries which are different in length, there is no simple crossover operator feasible to perform the proposed information exchange between parents.

A modified uniform crossover operator is proposed to deal with chromosomes in different lengths. Similar to conventional uniform crossover operator, a binary string is randomly created to control in each position, from which parent the child should inherit the gene. The length of the control string is assigned to the number of conditional attributes in the given data set. Therefore, there will be no risk taken on the misunderstanding of the dimensionality during the creation process of offspring. The modified uniform crossover operator is illustrated in Figure 3-3.

Figure 3-3: Illustration of modified uniform crossover operator

Because feature selection is applied in GARSC, many chromosomes have no gene in certain input dimension as indicated with the empty brackets “[]” in Figure 3-3. Therefore, when a pair of chromosomes is uniformly crossed over, there is a chance that one of the offspring has no gene in every input dimension. In such case, the “empty” chromosome will be replaced with a non-empty gene which has been randomly initialized.

3.2.9 Modified Mutation Operators

Unlike conventional mutation operators which simply vary the values of the selected genes, a set of three different mutation operators is proposed in GARSC. Please note that genes in GARSC have variable lengths because each gene represents the set of selected separation boundaries in the corresponding dimension (see Figure 3-3). Whenever a gene is selected to be mutated, one of the following operators is applied with equal probability:

- (1) Add one separation boundary to the gene. Assume a random feasible separation boundary is selected, if the length of the gene (after expansion) in the corresponding dimension is still within the maximum number of separation boundaries allowable and the new one fulfil the minimum distance requirement for all the others, then the new one is added to the gene. Otherwise, mutation is not applied.
- (2) Remove one separation boundary from the gene. If the gene is non-empty, a random separation boundary is selected among all the existing ones with equal probability and removed from the gene. Otherwise, mutation is not applied.
- (3) Vary the value of a randomly selected (with equal probability) separation boundary in the gene. The new separation boundary should fulfil the minimum distance requirement for all the others. Otherwise, mutation is not applied. If the gene is mutated, a re-ordering of the separation boundaries is performed to ensure the correctness of the gene representation.

Similar to tournament selection probability p , the mutation rate $mrte$, which defines the probability for each gene to be mutated, should increase from a small value in the early generations to a large value in the late generations. Based on this policy, $mrte$ is defined by Equation 3-12.

$$mrte = \frac{1}{NOF} + \frac{(NOF - 1) \cdot cgi}{NOF \cdot NOG}. \quad (3-12)$$

According to Equation 3-12, $mrte$ is equally distributed in the $(\frac{1}{NOF}, 1]$ interval. Therefore, each chromosome is approximately mutated once in one input dimension in the first generation of GA and each chromosome is approximately mutated on every gene in the last generation. This mutation rate seems to be much larger than those conventionally adopted values. However, because mutation of genes often fails due to various constraints, the actual mutation rate is not significantly large.

3.3 Summary

In this Chapter, a novel clustering technique named Genetic Algorithm based Rough Set Clustering or GARSC in short is proposed. Details of the technique are presented with sufficient illustrations and ample proofs. Special attentions have been taken to carefully design the fitness evaluation function to achieve a high level of both accuracy and interpretability. Parameters used in the genetic algorithm are designed to explore more feasible areas in the feature space during the early generations and exploit the almost converged areas in the late generations. GARSC is employed to

form clusters on the training data set. Clustering results are represented in a set of decision rules with minimum amount of essential information. These inference rules are transformed into fuzzy rules to be employed by the proposed neural fuzzy inference system. In summary, GARSC combines the strengths of genetic algorithm and rough set theory to define a systematic way to form clusters. Genetic algorithm is incorporated to search for optimal or suboptimal separation boundaries and rough set theory is applied to simplify the decision knowledge base. The correctness and expressiveness of GARSC affects much on the performance of the overall system. The overall GARSC algorithm is summarized in Table 3-1.

Table 3-1: The Proposed GARSC Algorithm

Step 1	Based on the given training data set, find all possible separation boundaries in all input dimensions; initialize the look-up table of fuzzy membership functions T_{mf} ; initialize the first population of GA.
Step 2	Based on the chromosome being analyzed x , construct the corresponding decision table DT_x (see Section 3.2.1); merge duplicate rows so that each row represents a cluster of data.
Step 3	Find the reduct of DT_x to perform feature selection and attribute reduction (see Section 2.6); merge again if duplicate rows exist.
Step 4	If conflict exists, remove inconsistent rules (see Section 3.2.4) and go back to Step 3.
Step 5	Convert the crisp decision rules into fuzzy rules; if the corresponding fuzzy membership functions are not available in T_{mf} , compute the membership functions and update T_{mf} (see Section 3.2.1).
Step 6	Based on the fuzzy rules, construct the NFIS and apply the training data set to compute the fitness value $f(x)$ (see Section 3.2.6).
Step 7	Go back to Step 2 until all chromosomes are evaluated.
Step 8	If the maximum number of generations is reached, terminate the clustering algorithm and output solutions.
Step 9	Select elites from the current generation and directly put them into the next generation without any changes (see Section 3.2.5).
Step 10	Select parents and subsequently apply the crossover and mutation operators on them (See Sections 3.2.8 and 3.2.9).
Step 11	Go back to Step 2 with the new population of chromosomes.

Chapter 4

GARSINFIS: Self-organizing Neural Fuzzy Inference System

“Less is more.”

- Ludwig Mies van der Rohe

This chapter introduces the proposed soft computing model termed Genetic Algorithm and Rough Set Incorporated Neural Fuzzy Inference System or GARSINFIS in short. It employs TSK type of fuzzy rules and has six layers wherein each layer performs respective fuzzy or non-fuzzy operations. Its complete architecture is presented with carefully defined mathematical formulations and its parameter tuning process is also discussed. GARSINFIS utilizes the fuzzy knowledge base systematically derived by the proposed genetic algorithm based rough set clustering technique to

self-organize its network structure.

4.1 GARSINFIS Architecture

GARSINFIS has a six-layered, feed-forward, and partially connected architecture as shown in Figure 4-1. The six layers are characterized by the fuzzy or non-fuzzy operations they perform. From left to right, the six layers are named Layer I - Input Layer, Layer II - Condition Layer, Layer III - Rule-base Layer, Layer IV - Normalization Layer, Layer V - Consequence Layer, and Layer VI - Output Layer. Referring to Figure 4-1, the rectangular boxes used to represent neurons in Layers II and V denote the antecedent and consequent parts of the employed fuzzy rules respectively. Values used for these two layers are adjusted during learning, which is often referred as the parameter tuning processes of fuzzy rules. The previous version of the network structure (fully connected) is proposed in [Wang 04], wherein a clustering method without knowledge reduction (also performs feature normalizations, which degrades interpretability) is integrated.

Each layer in GARSINFIS has a number of neurons denoted as N_d , where d denotes the layer index. In each layer, neurons are not connected to each other but only connected to neurons in adjacent layer(s). Because GARSINFIS employs GARSC to systematically derive reduct-fuzzy inference rules, its network size is relatively smaller than that of the other similar models such as the original ANFIS architecture [Jang 93]. Unlike ANFIS and many other models, GARSINFIS is not fully connected. Be-

Figure 4-1: GARSINFIS Architecture

cause feature selection is performed, not all input dimensions in the given data set are utilized. Because the maximum number of fuzzy membership functions in each dimension is constrained by the clustering technique, the number of neurons employed to represent the derived membership functions is also constrained. Because attribute selection in the antecedent part of fuzzy rules is performed, hence the derived fuzzy rules may not necessarily employ all the fuzzy membership functions in every selected dimension. Because rule pruning is performed during the clustering process, the number of neurons used to represent the fuzzy rule set is also minimized. In summary, GARSINFIS possesses all the aforementioned interpretability-driven characteristics, which makes its network structure relatively smaller and its inference rules easier to comprehend. Each layer of GARSINFIS is introduced with detailed descriptions and carefully defined with mathematical equations.

4.2 Input Layer

Input to the overall system is a data tuple represented in the vector form. The function of the input layer is to receive and translate the input vector into fuzzy singletons in each dimension and subsequently output the singletons to all the connected neurons in the next layer. Neurons in the input layer are denoted as linguistic neurons, because they represent input linguistic variables such as height, weight, eye sight, etc, when the input data is the physical conditions of a person.

Each linguistic neuron receives only one input which is the value of the data tuple in the corresponding dimension. Since feature selection is performed by the proposed clustering technique, hence not all input dimensions in the given data set are utilized, i.e. $N_1 \leq NOF$, where NOF is the total number of conditional attributes in the given data set. Each linguistic neuron outputs to different number of neurons in the next layer, which depends on the number of membership functions formulated in the corresponding dimension.

The input and output functions of the linguistic neurons are defined in Equations 4-1a and 4-1b respectively.

$$f_i^I = x_i, \quad (4-1a)$$

$$o_i^I = f_i^I, \quad (4-1b)$$

where f_i^I denotes the input of the i th linguistic neuron in Layer I; o_i^I denotes the output of the i th linguistic neuron in Layer I; x_i denotes the i th element

of input vector X ; and $i \in \{1, \dots, N_1\}$, $N_1 \leq NOF$.

4.3 Condition Layer

Inputs to the neurons in the condition layer are fuzzy singletons. The function of the condition layer is to compute the activation values of fuzzy membership functions embedded in neurons according to the fuzzy singleton inputs and subsequently output the activation values to the connected neurons in the next layer. Neurons in the condition layer are denoted as input label neurons, because they represent linguistic labels such as “short”, “medium”, and “high” of the corresponding input linguistic variable “height”.

Each input label neuron receives only one input which is the derived fuzzy singleton. The inputs received by input label neurons in the same dimension are all equivalent. The number of input label neurons in each dimension is equal to the number of fuzzy membership functions formulated in the same dimension, which is derived by GARSC. The value of this number is at least two because if only zero or one membership function is formulated in any dimension then this linguistic variable will be excluded from the knowledge base after the feature selection process completes. Furthermore, the maximum number of fuzzy membership functions in each dimension is constrained by the maximum number of partitions allowable in GARSC. Each input label neuron outputs to different number of neurons in the next layer. The number of connected neurons in the next

layer depends on how many rules the particular input linguistic label is associated with. This number is often smaller than the total number of fuzzy rules, because attribute reduction in the antecedent parts of inference rules are performed to remove redundancies.

In GARSINFIS, fuzzy membership functions of Gaussian type are employed. The input and output functions of the input label neurons are defined in Equations 4-2a and 4-2b respectively.

$$f_{ij}^H = -\frac{(o_i^I - c_{ij}^H)^2}{2\sigma_{ij}^{H2}}, \quad (4-2a)$$

$$o_{ij}^H = \exp(f_{ij}^H), \quad (4-2b)$$

where f_{ij}^H denotes the input of the j th input label neuron in the i th dimension in Layer II; o_{ij}^H denotes the output of the j th input label neuron in the i th dimension in Layer II; c_{ij}^H denotes the centroid of the j th cluster in the i th dimension; σ_{ij}^H denotes the standard deviation of the j th cluster in the i th dimension; $i \in \{1, \dots, N_1\}$, $j \in \{1, \dots, T_{N_2}\}$, $N_1 = N_2$; and T_i denotes the total number of fuzzy membership functions in the i th dimension.

The two parameters required to formulate a fuzzy Membership Function (MF) of Gaussian type are c_{ij}^H and σ_{ij}^H , which are defined in Equation 4-2a. Because the fuzzy knowledge base employed in GARSINFIS is systematically derived by GARSC, these two parameters for each fuzzy MF are also automatically determined (see Sections 3.2.1 and 3.3). Although GARSINFIS does not employ gradient-descent-based methods such as the back propagation algorithm [Rubio 10] or the least mean square algorithm

[Liu 08] to tune the MFs, it employs GA to perform MF optimizations. Numerous MFs are evaluated during the clustering process. Hence, when GA terminates, the best performing sets of MFs are provided. These MFs have been empirically proven to be accurate and precise along the fitness function optimization process.

4.4 Rule-base Layer

Inputs to the neurons in the rule-base layer are activated membership values. The function of the rule-base layer is to perform fuzzy reasoning based on the activated values received from different dimensions and subsequently output the activation value of the corresponding fuzzy rule to all the neurons in the next layer. Neurons in the rule-base layer are denoted as rule-base neurons, because they represent the antecedent parts of fuzzy rules. The number of rule-base neurons is equal to the number of fuzzy rules in the inference knowledge base, i.e. $N_3 = nor$, where *nor* is the actual number of fuzzy rules employed (or the number of clusters derived by GARSC).

Each rule-base neuron receives a number of inputs from the input label neurons. The actual number is equal to the number of arguments employed in the corresponding fuzzy rule. Because attribute reduction is performed on the antecedent parts of fuzzy rules, the rule-base neuron does not necessarily connect to the input label neurons in every dimension. Each rule-base neuron connects to all the neurons in the next layer to perform

normalization on the rule firing strengths.

GARSINFIS applies Computational Rule of Inference (CRI) approach [Zadeh 75] for the reasoning process with the *min* implication operator. The input and output functions of the rule-base neurons are defined in Equations 4-3a and 4-3b respectively.

$$f_k^{III} = \min(o_{ij}^{II}), \quad (4-3a)$$

$$o_k^{III} = f_k^{III}, \quad (4-3b)$$

where f_k^{III} denotes the input of the k th rule-base neuron in Layer III; o_k^{III} denotes the output of the k th rule-base neuron in Layer III; and $i \in \{1, \dots, N_1\}$, $j \in \{1, \dots, T_{N_2}\}$, $k \in \{1, \dots, N_3\}$, $N_3 = nor$.

4.5 Normalization Layer

Inputs to the neurons in the normalization layer are the rule firing strengths of all fuzzy rules. The function of the normalization layer is to scale the activation values of all fuzzy rules to the same level of reference and subsequently output the normalized values to the connected neuron in the next layer. Neurons in the normalization layer are denoted as normalization neurons. The number of normalization neurons is equal to the number of rule-base neurons, i.e. $N_3 = N_4$.

Each normalization neuron receives N_3 number of inputs. This normalization on rule firing strengths is necessary for correct fuzzy reasoning

process and its importance is analyzed in [Azeem 00]. Although ANFIS [Jang 93] and DENFIS [Kasabov 02] do not explicitly define this layer of neurons, they do perform similar operations to ensure the correctness of the inference process. Each normalization neuron connects to only one neuron in the next layer which represents the consequent part of the corresponding rule.

The input and output functions of the normalization the neurons are defined in Equations 4-4a and 4-4b respectively.

$$f_l^{IV} = \frac{o_l^{III}}{\sum_{k=1}^{N_3} o_k^{III}}, \quad (4-4a)$$

$$o_l^{IV} = f_l^{IV}, \quad (4-4b)$$

where f_l^{IV} denotes the input of the l th normalization neuron in Layer IV; o_l^{IV} denotes the output of the l th normalization neuron in Layer IV; and $k \in \{1, \dots, N_3\}$, $l \in \{1, \dots, N_4\}$, $N_3 = N_4$.

4.6 Consequence Layer

Inputs to the neurons in the consequence layer are the normalized rule firing strengths. The function of the consequence layer is to compute the prediction of each rule based on the normalized rule firing strength and subsequently output it to the only neuron in the next layer. Each neuron in the consequence layer is denoted as a consequence neuron because it represents the consequence part of the corresponding fuzzy rule. The

number of consequence neurons is equal to the number of normalization neurons, i.e. $N_5 = N_4$.

Each consequence neuron receives inputs from two sources, one from the normalization layer and N_1 number of system inputs in singleton form. Each consequence neuron connects to the only neuron in the next layer which aggregates the inference result.

The input and output functions of the consequence neurons are defined in Equations 4-5a and 4-5b respectively.

$$f_m^V = c_0 + c_1x_1 + \cdots + c_ix_i + \cdots + c_{N_1}x_{N_1}, \quad (4-5a)$$

$$o_m^V = o_m^{IV} f_m^V, \quad (4-5b)$$

where f_m^V denotes the input of the m th consequence neuron in Layer V; o_m^V denotes the output of the m th consequence neuron in Layer V; c_0 is a constant value; c_i denotes the coefficient of the i th linguistic variable; and $m \in \{1, \dots, N_5\}$.

Equation 4-5a actually represents the consequent part of a first-order TSK type of fuzzy rule. Furthermore, zero-order rule can be easily generated by assigning $c_i \leftarrow 0, \forall i \in \{1, \dots, N_1\}$. In GARSC, values used for the decision attribute are crisp. These crisp values are directly assigned to c_0 to formulated zero-order TSK type of fuzzy rules. Zero-order rules possess great level of interpretability. However, if the proposed system is used in a complex problem domain or user specifically requires, then zero-order TSK type of fuzzy rules can be extended to more accurate first-order rules. Al-

though semantic meanings significantly degrade, the results are expected to increase. The parameter tuning technique employed in GARSINFIS is the Recursive Least Squares (RLS) algorithm, which is introduced as follows.

The consequent part of a first-order TSK type of fuzzy rule is defined in Equation 4-5a. Suppose M fuzzy rules are employed by the inference system, then W , the coefficient matrix to be optimized, is defined in Equation 4-6.

$$W = [c_{10} \cdots c_{m0} \ c_{11} \cdots c_{m1} \ \cdots \ c_{1I} \cdots c_{MI}]^T. \quad (4-6)$$

where I denotes the number of features selected in the given data set, $I = N_1$.

Suppose the system is presented with P number of input vectors, then based on Equation 4-4a, the normalized activation value matrix $\bar{\alpha}$ is defined by Equation 4-7.

$$\bar{\alpha} = \begin{bmatrix} \bar{\alpha}_{11} & \cdots & \bar{\alpha}_{M1} \\ \vdots & \cdots & \vdots \\ \bar{\alpha}_{1P} & \cdots & \bar{\alpha}_{MP} \end{bmatrix}. \quad (4-7)$$

Based on Equation 4-7 and the same P number of input vectors, the weighted input matrix A is defined in Equation 4-8.

$$A = \begin{bmatrix} \bar{\alpha}_{11} & \cdots & \bar{\alpha}_{M1} & \bar{\alpha}_{11}x_{11} & \cdots & \bar{\alpha}_{M1}x_{11} & \cdots & \bar{\alpha}_{11}x_{I1} & \cdots & \bar{\alpha}_{M1}x_{I1} \\ \vdots & \cdots & \vdots & \vdots & \cdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ \bar{\alpha}_{1P} & \cdots & \bar{\alpha}_{MP} & \bar{\alpha}_{1P}x_{1P} & \cdots & \bar{\alpha}_{MP}x_{1P} & \cdots & \bar{\alpha}_{1P}x_{IP} & \cdots & \bar{\alpha}_{MP}x_{IP} \end{bmatrix}. \quad (4-8)$$

Based on Equations 4-6 and 4-8, the system output Y of P input vectors is defined in Equation 4-9. Dimensions of matrices Y , A , and W are $P \times 1$, $P \times M(I + 1)$, and $M(I + 1) \times 1$ respectively.

$$Y = AW. \quad (4-9)$$

If the input vectors are training samples, then their desired value matrix D is also provided. The optimization problem becomes the determination of the optimal coefficient matrix W^* which fulfills Equation 4-10.

$$AW^* = D. \quad (4-10)$$

One of the most concise ways to solve Equation 4-10 is to apply the pseudoinverse technique. The estimation of W^* is defined by Equation 4-11.

$$W^* = (A^T A)^{-1} A^T D, \quad (4-11)$$

where $(A^T A)^{-1} A^T$ is the pseudoinverse of A , if $A^T A$ is nonsingular.

In Equation 4-11, the determination of W^* requires the presence of P training samples. To optimize the coefficient matrix W in an adaptive way is to employ the Recursive Least Squares (RLS) algorithm [Angelov 05, Angelov 08, Han 10, Rong 11]. Let a_p denotes the p th row of matrix A ,

then W^* can be recursively estimated using Equation Set 4-12.

$$W_p = W_{p-1} + S_p a_p^T \overbrace{(D_p - a_p W_{p-1})}^{\text{prediction error}}, \quad (4-12a)$$

$$S_p = S_{p-1} - \frac{S_{p-1} a_p^T a_p S_{p-1}}{1 + a_p S_{p-1} a_p^T}, \quad p = 1, \dots, P, \quad (4-12b)$$

$$W_0 = [c_{10} \cdots c_{m0} \underbrace{0 \cdots 0}_{MI}]^T, \quad (4-12c)$$

$$S_0 = \gamma I, \quad (4-12d)$$

$$W^* = W_P, \quad (4-12e)$$

where γ is a large positive number and I is the identity matrix of dimension $M(I+1) \times M(I+1)$.

S_p in Equation 4-12b denotes the error covariance matrix for the p th input vector. The detailed derivation steps are available in [Lin 96a]. Unlike the conventional assignment $W_0 = 0$, in GARSINFIS parameter tuning, W_0 is initialized and defined in Equation 4-12c. Values of $c_{10}, \dots, c_{m0}$ are systematically derived by GARSC which formulates zero-order TSK type of fuzzy rules. RLS algorithm is selected to perform parameter tuning because of its online learning capability and fast convergence speed.

4.7 Output Layer

This layer consists of only one neuron which is denoted as the output neuron. It aggregates all the inputs received from consequence neurons and present the result as the system output. Because TSK type of fuzzy

rules are employed in GARSINFIS, the system output is a crisp value which means defuzzification is not required. Since GARSINFIS provides only one output, it is classified as a multiple-input-single-output system.

The input and output functions of the output neuron are defined in Equations 4-13a and 4-13b respectively.

$$f^{VI} = \sum_{m=1}^{N_5} o_m^V, \quad (4-13a)$$

$$o^{VI} = f^{VI}, \quad (4-13b)$$

where f^{VI} denotes the input of the output neuron and o^{VI} denotes the output of the output neuron.

4.8 Summary

This chapter introduces the proposed Genetic Algorithm and Rough Set Incorporated Neural Fuzzy Inference System or GARSINFIS in short. It realizes the fuzzy inference process through a six-layered network architecture. The six layers are described in details and carefully defined with mathematical equations of how to perform respective fuzzy or non-fuzzy operations. GARSINFIS utilizes the fuzzy knowledge base systematically derived by GARSC and performs rule tuning. Different parameters of the fuzzy membership functions are evaluated by GARSC. Therefore, when GA terminates, the best sets of MFs after optimization are derived. For the consequent parts of rules, initial rules are of zero-order TSK type,

which possess high level of interpretability. However, if the user wants to improve the accuracy of the model in complex application domains, zero-order fuzzy rules are extended to first-order rules by applying the RLS algorithm. GARSINFIS has a relatively small network size because it employs a reduct-inference knowledge base. Each layer of GARINFIS is explicitly distinguished and introduced to illustrate how the reduct-knowledge base generates the relatively small network in size. The general function of mapping input $X = (x_1, \dots, x_i, \dots, x_I)$ into output y using M number of first-order TSK type of fuzzy rules is defined in Equation 4-14.

$$y = \frac{\sum_{m=1}^M \alpha_m (c_{m0} + c_{m1}x_1 + \dots + c_{mi}x_i + \dots + c_{mI}x_I)}{\sum_{m=1}^M \alpha_m}, \quad (4-14)$$

where α_m is the firing strength of the m th rule and M denotes the total number of derived fuzzy rules, $M = N_4$.

When the implication operator is selected as the *min* operator, then α_m is defined by Equation 4-15.

$$\alpha_m = \min(\mu_{A_{m1}}(x_1), \dots, \mu_{A_{mi}}(x_i), \dots, \mu_{A_{mI}}(x_I)), \quad (4-15)$$

where $\mu_{A_{mi}}(x_i)$ denotes the membership value of x_i according to the membership function A_{mi} of the m th rule in the i th dimension.

Chapter 5

Experimental Results and Analyses

“Practice alone constitutes the criterion of truth.”

- Karl Marx

This chapter evaluates the performance of different configurations of GARSINFIS, such as the necessity of translation from crisp rules to fuzzy ones, accuracy improvement by transformation of the fitness function into an accuracy-driven one, further accuracy improvement by employment of less interpretable fuzzy rules, and formation of a committee to increase accuracy without jeopardizing interpretability. These different configurations are assessed with well-known and public downloadable data sets. Extensive experimental results show that GARSINFIS produces satisfactory accuracy

with relatively less amount of inference knowledge compared to the other models.

5.1 Miscellaneous Experimental Setups

GARSINFIS is a carefully constructed neural fuzzy inference system that employs TSK type of fuzzy rules, which are systematically derived from the clustering results produced by GARSC. The objective of the experiments is to evaluate the performance of GARSINFIS under different configurations for leveraging between accuracy and interpretability.

5.1.1 Design of Experiments

There are many ways to configure GARSINFIS networks. If the clustering results of GARSC are represented in crisp sets, then GARSINFIS employs crisp rules rather than fuzzy rules. The trade-off between interpretability and accuracy can be easily tuned by adjusting the coefficients of different components defined in the fitness function. Please note that the coefficients can be assigned to any real numbers based on user requirements and the complexity of the application domain. The accuracy of GARSINFIS can also be improved by either converting zero-order TSK type of fuzzy rules to first-order ones or forming a committee of multiple GARSINFIS networks. These different configurations of GARSINFIS are evaluated in different sections and are summarized in Table 5-1.

Table 5-1: Summary of different configurations of GARSINFIS

Index	Configuration	Remarks
1	GARSINFIS-crisp	performance evaluation is based on the employed crisp rules; selected boundaries are different with those of fuzzy rules.
2	GARSINFIS-a&i	fitness function focuses on both accuracy and interpretability; $\tau_{1,\dots,5} = 1$: $f(x) = (1 - a) \frac{NOD}{K} + \frac{nof}{NOF} + \frac{nor}{NOD} + \frac{noa}{NOF \cdot NOD} + \frac{mse}{NOF}$
3	GARSINFIS-a	rules are simplified but fitness function focuses on accuracy only; $\tau_{1,5} = 1, \tau_{2,3,4} = 0$: $f(x) = (1 - a) \frac{NOD}{K} + \frac{mse}{NOF}$
4	GARSINFIS-1	zero-order TSK type of fuzzy rules derived by GARSINFIS-a are converted to first-order ones by applying the KF algorithm.
5	GARSINFIS-c	a committee consists of ten GARSINFIS-a networks; whether the committee is local or global depends on the training data sets.

Two different configurations selected from Table 5-1 are compared in terms of performance for each data set. Experiments are designed to show the necessity of translation from crisp rules to fuzzy ones by comparing GARSINFIS-crisp against GARSINFIS-a&i, to show accuracy improvement by transformation of the fitness function into an accuracy-driven one by comparing GARSINFIS-a&i against GARSINFIS-a, to show further accuracy improvement by employment of less interpretable fuzzy rules by comparing GARSINFIS-a against GARSINFIS-1, and to show increase of accuracy without jeopardizing interpretability by comparing GARSINFIS-a against GARSINFIS-c.

Unless it is explicitly stated, each GARSINFIS experiment uses two thirds of randomly selected data samples for training and the remaining one third for testing. Furthermore, all the other benchmarking models apply the exact same pair of training and testing data sets for each experiment. Therefore, differences in performance of all models are compared on equal bases. Each application is simulated for ten times, i.e. each data set is evaluated as a three-fold cross validation problem and is simulated for ten times to average the performance.

All experiments are run on different computers with the same specifications to record consistent computation times. All source codes are written in MATLAB language and the computers used for experiments are installed with Intel(R) Core(TM)2 DUO CPU at 2.53GHz each and 3G physical RAM.

5.1.2 Control Parameters

Some control parameters of GARSINFIS, which do not vary for all application domains, are given as follows.

- The maximum number of separation boundaries allowable in any input dimension is set to two. Therefore, the maximum number of fuzzy membership functions in any input dimension is three. This guarantees a great level of interpretability because there are at most three linguistic labels, namely small, medium, and large, assigned in any input dimension.
- Elitism ratio is set to 0.1, i.e. the top 10% elites in the current generation are directly brought into the next generation. Survival of highly fit elites guarantees exploitation and preserves better performing genes.
- Crossover rate is set to 1, i.e. every pair of selected parents eventually mates. High crossover rate guarantees exploration.
- Tournament size is set to two, which is meant to ease the stress during selection. However, the selection of parents eventually becomes more

stressful with the increase of the tournament selection probability p , which is defined by Equation 3-11 on Page 73.

5.1.3 Evaluation of Results

When benchmarking the experimental results, some well-established models such as C4.5 decision tree [Quinlan 93], Naive Bayes classifier [Ross 85], Support Vector Machine (SVM) [Cortes 95], Multi-Layer Perceptron (MLP) network [Haykin 98], Radial Basis Function (RBF) network [Haykin 98], ANFIS [Jang 93], DENFIS [Kasabov 02], RS-POPFNN [Ang 05], and RS-HeRR [Liu 07] are used for comparisons. RS-HeRR is implemented in C++ and all the other NFIS models are implemented in MATLAB. The rest benchmarking models are implemented in WEKA [Hall 09]. Ample evidence shows that GARSINFIS produces satisfactory accuracy with relatively less amount of inference knowledge compared to the other models.

Other than accuracy on different data sets and computation times, number of employed features, rules, and arguments are also recorded for comparisons. The number of features utilized by GARSINFIS is the actual number of features selected from the given data set to perform inferences. The number of rules employed by GARSINFIS is the actual number of rules in the reduct-fuzzy rule set after removals of redundancies and inconsistencies. The number of arguments is defined by the total number of arguments employed in all the employed rules after attribute reduction.

In terms of comparisons on the selected features, GARSINFIS, C4.5,

RS-POPFNN, and RS-HeRR are discussed, because the other models utilize all given features without preferences. Even if the associated weights of certain input dimensions can be significantly small in some models, it is still not considered as feature selection.

In terms of the size of the employed rule set, the actual number of employed rules is recorded for the five neural fuzzy inference systems, namely GARSINFIS, ANFIS, DENFIS, RS-POPFNN, and RS-HeRR. For C4.5, the number of tree leaves is recorded because it is equivalent to the number of crisp decision rules. For Naive Bayes classifier and SVM, the number of rules is not applicable (although extracting rules from SVM is possible [Barakat 10], in this thesis, SVM constructs multiple classifiers to perform binary classifications for each and every pair of output classes). For MLP network, the number of hidden neurons, which is systematically determined by the total number of input dimensions and the number of output classes, is recorded. For RBF, the number of radial basis neurons, which is pre-determined by the number of clusters in every output class, is recorded.

In the comparison tables, the winner of any particular column is highlighted in bold. The winner is either the largest accuracy value or the least amount of information employed. Because multiple experiments are conducted on each model for each data set, stability of every model is also recorded and compared. Stability is examined in terms of the standard deviations of multiple experiments. The smallest value in terms of stability is the winner whose value is also highlighted in bold. Other than mean and standard deviation, geometric mean ($G\text{-Mean}_{test}$) of the accuracy on the testing data sets are also reported as another statistical measurement.

In this thesis, ANFIS utilizes clustering results derived using the employed Fuzzy C-Means (FCM) clustering technique. In this way, ANFIS is more comparable to GARSINFIS because the number of rules employed by ANFIS is greatly reduced. However, extra efforts are taken to carefully select the optimal number of pre-defined clusters through trial-and-error. This is a great disadvantage of ANFIS, because its network structure is not self-organized but has to be pre-defined by employing additional knowledge. Although GARSINFIS employs GA which is time-consuming, the search is guided by the fitness function and bounded with carefully defined constraints to obtain the desirable results. On the other hand, ANFIS with FCM runs randomly which requires a sweep of parameters for multiple times or heuristic expert knowledge.

The clustering technique ECM used in DENFIS constructs clusters based on distances between data samples in high-dimensional space. ECM often derives a large number of clusters and as a result DENFIS employs a large number of rules. This situation definitely becomes worse when the dimensionality of the given data set increases and utilizing a greater number of rules does not necessarily guarantee more accurate results. This is the main drawback of DENFIS in which feature selection is not performed. Generally speaking, DENFIS often fails to alleviate the curse of dimensionality problem.

Although RS-POPFNN [Ang 05] and RS-HeRR [Liu 07] also employ rough set approximations to perform inference knowledge reduction, they are different from GARSINFIS in two major aspects. (1) RS-POPFNN and RS-HeRR employ Mamdani type of fuzzy rules (see Section 2.3.5 on

Page 30). Generally speaking, the number of derived TSK type of fuzzy rules is smaller than that number of Mamdani type. (2) RS-POPFNN employs the Pseudo-Outer-Product (POP) rule identification algorithm and RS-HeRR employs the Hebbian-based rule generation algorithm to automatically derive the initial rule sets. Because the two rule identification algorithms normally produce unnecessary large numbers of rules, rough set approximations are applied to improve the interpretability. On the other hand, the rules employed by GARSINFIS are optimized through the iterative clustering process wherein rough set approximations are incorporated with GA. Because GARSINFIS leverages between accuracy and interpretability by adopting different configurations, it offers more flexibility than the other two models.

5.2 Necessity of Translation from Crisp to Fuzzy

In this section, the necessity of translation from crisp decision rules to fuzzy inference rules is discussed. Respective comparisons are made by applying GARSINFIS to two well-known and public downloadable data sets, namely iris classification and wine recognition data sets. Experimental results show that by the transformation of crisp rules into fuzzy ones, performance of GARSINFIS improves because fuzzy rules allow tolerance around separation boundaries and handle unforeseen circumstances well. Please note that the performance comparisons between crisp rules and fuzzy ones are

not based on direct transformations. Two respective GARSINFIS models are optimized separately. The actual mutation rate in GARSC is also discussed in this section. It is introduced in Section 3.2.9 on Page 75 that the mutation rate defined in GARSC is relatively higher than the rates defined in other models which also employ genetic algorithms. However, experimental results show that the actual mutation rate is not unnecessarily high due to various constraints on different mutation operators.

5.2.1 Iris Classification Data Set

Iris classification data set is widely used as a baseline application in discrimination and clustering analyses. This data set is downloadable from the UCI Machine Learning Repository [(UCI) 14]. There are three species of iris flowers gathered in this data set, namely Setosa, Versicolor, and Virginica, and each class consists of 50 data samples. There are four input features, namely sepal length, sepal width, petal length, and petal width. This data set is known to be distinguishable in the last two features (petal length and petal width) as illustrated in Figure 5-1.

The population size of GARSC is set to 100 and the number of generations is set to 30. Experimental results of applying GARSINFIS which employs fuzzy rules on iris classification data set are summarized in Table 5-2. The employed fuzzy rules are zero-order TSK type which are evaluated by the fitness function defined in Equation 3-3 on Page 65 with all coefficients set to 1.

Figure 5-1: Plot of iris classification data set on two distinguishable features

Table 5-2: Summary of applying GARSINFIS on iris classification data set

Experiment Index	Training (%)	Testing (%)	Whole (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	99.00	96.00	98.00	2	3	4	46.63
2	100.0	96.00	98.67	2	4	5	51.30
3	98.00	98.00	98.00	3	4	5	49.14
4	99.00	96.00	98.00	2	3	4	60.39
5	99.00	96.00	98.00	2	3	4	56.90
6	99.00	96.00	98.00	3	4	5	45.11
7	98.00	98.00	98.00	2	3	4	42.91
8	99.00	98.00	98.67	3	4	6	52.75
9	99.00	96.00	98.00	2	3	4	57.03
10	99.00	96.00	98.00	2	3	4	41.88
mean	98.90	96.60	98.13	2.3	3.4	4.5	50.40
std	0.57	0.97	0.28	0.48	0.52	0.71	6.37

Among all the ten experiments, three of them employ three input features and the rest of the experiments employ only two features. Please note that because all components to derive a high level of interpretability are defined in the fitness function, hence, feature selection is considered to be necessary for these highly accurate classification results. Although only experiment-2 achieves 100% accuracy on the training data set, all experiments utilize less number of input features to achieve an average accuracy of 98.9% on the training data set. On average, GARSINFIS utilizes 2.3 input features to construct a model to achieve 98.13% accuracy on the whole data set. This result is not surprising because iris classification data set is known to be well distinguishable by only utilizing petal length and petal width which is illustrated in Figure 5-1. However, since both prior knowledge and human intervention are not applied, the fact that GARSINFIS utilizes only the most essential information to achieve an average accuracy of 96.6% on the testing data set is really encouraging.

Among all the ten experiments, four of them employ four rules to produce an average accuracy of 99% on the training data set, 97% on the testing data set, and 98.34% on the whole data set. The other experiments employ three rules to achieve an average accuracy of 98.83%, 96.33%, and 98% respectively. This discovery shows that in the iris classification application, employment of more rules constructs a larger network with compensation of accuracy improvement. On average, there are 3.4 rules employed by GARSINFIS and each rule consists of $4.5/3.4 = 1.32$ arguments in the antecedent part. Figure 5-2 illustrates the fuzzy membership functions derived in experiment-7 because in this experiment, GARSINFIS employs

the least amount of information to achieve a balanced accuracy. Table 5-3 shows the fuzzy rules employed in the same experiment.

Table 5-3: Derived fuzzy rules of iris classification data set

IF petal length is small	THEN class is Setosa
IF petal length is medium and petal width is medium	THEN class is Versicolor
IF petal length is large	THEN class is Virginica

Figure 5-2: Derived fuzzy membership functions by GARSC

To demonstrate the superior advantage in terms of interpretability of the MFs derived by the proposed GARSC technique, the MFs derived by ECM clustering technique, which is used in DENFIS, are illustrated in Figure 5-3. For comparison purpose, only MFs in petal length and petal width dimensions are shown and ECM performs clustering on the same data set as GARSC does.

Comparing Figures 5-3 to 5-2, the MFs derived by GARSC are clearly assigned to comprehensive linguistic labels and adjacent MFs do not over-

Figure 5-3: Derived fuzzy membership functions by ECM

lap much between each other to avoid ambiguities. Furthermore, ECM performs data normalization in every input dimension that the semantic meanings of the derived MFs have no references to the original values (backward transformations are required).

To demonstrate how rough set theory greatly reduces the inference rule base, the fuzzy rules, which are optimized by GA but not simplified by rough set approximations, are listed in Table 5-4.

Table 5-4: Derived fuzzy rules if rough set theory is excluded

IF	sepal width is large and petal length is small and petal width is small	THEN class is Setosa
IF	sepal width is medium and petal length is medium and petal width is medium	THEN class is Versicolor
IF	sepal width is small and petal length is large and petal width is large	THEN class is Virginica
IF	sepal width is medium and petal length is large and petal width is large	THEN class is Virginica

Comparing Tables 5-4 to 5-3, on the same data set, the fuzzy rules derived without applying rough set approximations are more complicated than those simplified with rough set approximations. Please note that the computation time to derive the rules shown in Table 5-4 is 175.33 seconds because most evaluated GARSINFIS networks employ more number of rules and more number of arguments in the antecedent part of each rule. MFs derived in the petal length and petal width dimensions are almost identical to those illustrated in Figure 5-2. Therefore the derived MFs are not shown in this thesis. It is encouraging to learn that although rough set approximations are not applied, the rule set presented in Table 5-4 does not select sepal length feature. This is because the most optimal solution does not select any separation boundary in that dimension and the fitness function focuses on both accuracy and interpretability. The rules set presented in Table 5-4 has four rules and each rule has three arguments in the antecedent part, while the rule set presented in Table 5-3 only has three rules and each rule has 1.33 arguments on average. Although the rule set presented in Table 5-4 achieves 100% accuracy on the training data set, both rule sets achieve 98% accuracy on the testing data set. Therefore, it is shown in this particular experiment that reduct-inference rule set achieves higher level of interpretability and at the same time does not sacrifice accuracy.

The actual mutation rate of GARSC over all generations are plotted in Figure 5-4. Although the pre-defined mutation rate in the last generation reaches 1, the actual mutation rate is less than 0.57. Figure 5-4 shows that the assignment of a high mutation rate is necessary because there are

many constraints on the defined mutation operators which are discussed in Section 3.2.9 on Page 75.

Figure 5-4: Theoretical and actual mutation rates of iris classification data set

To show the necessity of translation from crisp decision rules to fuzzy inference rules and to benchmark the performance of GARSINFIS, respective experiments are conducted and results are presented in Table 5-5. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Comparing GARSINFIS-crisp which utilizes crisp decision rules to GARSINFIS-a&i whose performance is summarized in Table 5-2, they both utilize relatively small number of features and rules. However, the accuracy of GARSINFIS-a&i is better than that of GARSINFIS-crisp on every data set. This shows that fuzzy rules perform better than crisp rules because

Table 5-5: Benchmarks on iris classification data set

Model		Training (%)	Testing (%)	Whole (%)	Features	Rules	G-Meant _{test} (%)
GARSINFIS-a&i	mean	98.90	96.60	98.13	2.3	3.4	96.60
	std	0.57	0.97	0.28	0.48	0.52	
GARSINFIS-crisp	mean	96.10	95.20	95.80	2.2	3.7	95.19
	std	0.57	1.03	0.45	0.42	0.48	
C4.5	mean	94.90	94.60	94.80	2.1	4.1	94.58
	std	1.20	1.90	0.76	0.32	0.57	
Naive Bayes	mean	95.50	96.20	95.73	4	N.A.	96.19
	std	0.97	1.48	0.72			
SVM	mean	96.10	97.20	96.47	4	N.A.	97.19
	std	0.88	1.69	0.45			
MLP	mean	97.00	95.60	96.53	4	6	96.59
	std	1.16	1.58	0.53			
RBF	mean	98.70	94.80	97.40	4	6	94.79
	std	1.16	1.40	0.49			
ANFIS	mean	99.70	96.40	98.60	4	4.6	96.39
	std	0.48	1.27	0.49		0.52	
DENFIS	mean	100.0	95.20	98.40	4	13.4	95.19
	std	0.00	1.69	0.56		0.52	
RS-POPFNN	mean	97.90	94.80	96.87	3.9	13.4	94.78
	std	0.57	2.15	0.63	0.32	2.21	
RS-HeRR	mean	97.90	95.60	97.13	2.3	8.5	95.60
	std	0.99	0.84	0.77	0.48	3.27	

fuzzy membership functions allow tolerance beyond separation boundaries and handle unforeseen circumstances better. Furthermore, GARSINFIS-crisp performs better than C4.5 because the crisp decision rules employed by GARSINFIS-crisp are optimized by GA. The crisp rule set derived by GARSINFIS-crisp in experiment-7 is presented in Table 5-6.

Table 5-6: Derived crisp rules of iris classification data set

IF petal width ≤ 0.6	THEN class is Setosa
IF petal length ≤ 4.9 and $0.6 < \text{petal width} \leq 1.7$	THEN class is Versicolor
IF petal length > 4.9 and $0.6 < \text{petal width} \leq 1.7$	THEN class is Virginica
IF petal width > 1.7	THEN class is Virginica

In terms of accuracy on the training data set, DENFIS performs best because it achieves 100% accuracy in all the ten experiments. However, when comparing against GARSINFIS-a&i, the difference is $100\% - 98.9\% = 1.1\%$, which is approximately 1.1 data samples. For such a small difference, GARSINFIS-a&i utilizes 2.3 input features and 3.4 rules on average. DENFIS on the other hand, utilizes all 4 input features and 13.4 rules. Furthermore, DENFIS is actually over-fitted to the training data set which leads to its relatively low accuracy on the testing data set.

In terms of accuracy on the testing data set, SVM performs best. Actually, for SVM, the accuracy on the training data set is higher than that on the testing data set (same as Naive Bayes). This suggests that SVM has higher level of generalization, which prevents it from over-fitting on the training data set. In this measurement, GARSINFIS-a&i performs second best (best among all NFISs). This implies that GARSINFIS-a&i maintains a good level of generalization by the employment of a relatively small inference knowledge base.

In terms of accuracy on the whole data set, ANFIS performs best. However, the difference is $98.6\% - 98.13\% = 0.47\%$, which is approximately 0.7 data sample, when comparing to GARSINFIS-a&i. This small amount of difference is basically contributed by the high accuracy of ANFIS on the training data set. Furthermore, ANFIS employs more number of features and more number of rules than GARSINFIS-a&i does.

In terms of interpretability, C4.5 utilizes the smallest number of features. GARSINFIS-a&i employs the smallest number of rules. Although C4.5 utilizes $2.3 - 2.1 = 0.2$ less number of features than GARSINFIS-a&i does, its accuracy on the whole data set is $98.13\% - 94.8\% = 3.33\%$, which is approximately 5 data samples, lower than that of GARSINFIS-a&i.

Because for the iris classification data set, all models perform well and stably on the testing accuracy, the geometric means listed in the last column of Table 5-5 are almost identical to the arithmetic mean. It is shown that GARSINFIS-a&i achieves the highest testing accuracy among all NFISs.

5.2.2 Wine Recognition Data Set

Wine recognition data set, which is downloadable from the UCI Machine Learning Repository [(UCI) 14], is widely applied by many models for performance evaluations. This data set consists of chemical analyses of wines produced in the same region of Italy but derived from three different cultivars. The input features are the recording of thirteen different constituents found in each type of wine which are listed in Table 5-7. Altogether there are 178 data samples for three different kinds of wine. Class I has 59 samples. Class II has 71 samples. Class III has 48 samples.

Table 5-7: Input features of wine recognition data set

1	Alcohol	6	Total phenols	10	Color intensity
2	Malic acid	7	Flavanoids	11	Hue
3	Ash	8	Nonflavanoid phenols	12	OD280/OD315 of diluted wines
4	Alcalinity of ash	9	Proanthocyanins	13	Proline
5	Magnesium				

The population size of GARSC is set to 100 and the number of generations is set to 60. Experimental results of applying GARSINFIS which employs fuzzy rules on wine recognition data set is summarized in Table 5-8.

Table 5-8: Summary of applying GARSINFIS on wine recognition data set

Experiment Index	Training (%)	Testing (%)	Whole (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	100.0	93.33	97.75	3	6	12	1826.84
2	100.0	93.32	97.75	4	6	13	1572.57
3	100.0	94.92	98.31	3	5	10	1391.07
4	100.0	95.00	98.31	4	5	9	1289.97
5	100.0	93.33	97.75	4	7	15	1603.72
6	100.0	91.67	97.19	3	5	10	1387.12
7	100.0	98.31	99.44	5	7	14	1226.74
8	100.0	96.61	98.88	3	6	12	1478.27
9	100.0	91.67	97.19	3	4	8	1309.01
10	100.0	95.00	98.31	3	5	9	1256.39
mean	100.0	94.31	98.09	3.5	5.6	11.2	1434.17
std	0	2.1	0.71	0.71	0.97	2.35	188.33

Experiment-7 utilizes the largest number of input features which is five. However, this is a rather small number comparing to the total number of thirteen. By employing more features, the accuracy of experiment-7 is significantly increased. However, in wine recognition application, employment of more rules does not necessarily guarantee an improvement in accuracy. All the ten experiments achieve 100% accuracy on the training data set and the lowest accuracy on the testing data set is 91.67%. On average, GARSINFIS utilizes 3.5 input features and 5.6 rules that each rule consists of $11.2/5.6 = 2$ arguments in the antecedent part to achieve 98.09% accuracy on the whole data set. The fact that GARSINFIS utilizes at most five input features to achieve highly accurate inferences is encouraging.

The theoretical and actual mutation rates of GARSC are plotted in Figure 5-5. Applying GARSINFIS on wine recognition data set once again shows the necessity of defining a high theoretical mutation rate to better explore the universe of discourse.

To show the necessity of translation from crisp decision rules to fuzzy inference rules and to benchmark the performance of GARSINFIS, respective experiments are conducted and results are presented in Table 5-9. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Comparing GARSINFIS-crisp which utilizes crisp decision rules to GARSINFIS-a&i whose performance is summarized in Table 5-8, GARSINFIS-a&i not only achieves higher accuracy on every data set but also employs lesser amount of inference knowledge. This again shows the necessity of

Figure 5-5: Theoretical and actual mutation rates of wine recognition data set

Table 5-9: Benchmarks on wine recognition data set

Model		Training (%)	Testing (%)	Whole (%)	Features	Rules	G-Mean _{test}
GARSINFIS-a&i	mean	100.0	94.31	98.09	3.5	5.6	94.29
	std	0.00	2.10	0.71	0.71	0.97	
GARSINFIS-crisp	mean	96.13	93.66	95.30	3.8	5.9	93.63
	std	1.97	2.19	1.53	0.63	0.88	
C4.5	mean	90.88	94.13	91.97	4.0	5.6	94.06
	std	2.29	3.49	1.79	1.25	0.84	
Naive Bayes	mean	97.13	97.48	97.25	13	N.A.	97.47
	std	1.14	1.80	0.89			
SVM	mean	98.82	97.82	98.49	13	N.A.	97.81
	std	0.91	1.37	0.75			
MLP	mean	97.21	97.48	97.30	13	11	97.48
	std	1.06	1.63	0.78			
RBF	mean	100.0	97.82	99.27	13	6	97.81
	std	0.00	1.96	0.65			
ANFIS	mean	100.0	96.98	98.99	13	3.7	96.97
	std	0.00	2.05	0.69			
DENFIS	mean	100.0	96.98	98.99	13	42.8	96.97
	std	0.00	1.73	0.58			
RS-POPFNN	mean	99.83	92.12	97.26	12.5	90.5	92.04
	std	0.35	4.02	1.34	0.71	8.40	
RS-HeRR	mean	100	94.63	98.21	4.8	73.0	94.58
	std	0.00	2.15	0.63	0.32	2.21	

transformation from crisp rules to fuzzy ones. Furthermore, GARSINFIS-crisp again performs better than C4.5, which is sensitive to small changes in the given data set, in terms of accuracy on the whole data set.

In terms of accuracy on the training data set, most NFISs (except RS-POPFNN) and RBF which employs Gaussian radial basis functions achieve 100% accuracy for all the ten experiments. This indicates that data samples in wine recognition data set are normally distributed in the respective input features. Therefore, generally speaking, fuzzy and statistical models perform better than crisp models in this application.

In terms of accuracy on the testing data set, both SVM and RBF perform best with the difference of $97.82\% - 94.31\% = 3.51\%$, which is approximately 2.1 data samples, when comparing against GARSINFIS-a&i. However, SVM performs binary classifications without explicit rules and the weights associated with the radial basis neurons in RBF are difficult for human beings to comprehend.

In terms of accuracy on the whole data set, RBF performs best. The difference between the highest value and that of GARSINFIS-a&i is $99.27\% - 98.09\% = 1.18\%$, which is approximately 2.1 number of data samples.

In terms of interpretability, GARSINFIS-a&i utilizes the least number of input features and ANFIS utilizes the least number of rules. The credit of only 3.7 rules employed by ANFIS should be given to its Fuzzy C-Means (FCM) clustering technique. However, the number of clusters required to perform FCM is determined through trial-and-error, which involves more efforts than the other self-organizing NFISs (GARSINFIS, DENFIS, RS-

POPFNN, and RS-HeRR). It is also interesting to note that the number of rules employed by DENFIS is exceedingly large, where on average 42.8 rules are employed to represent 118.4 data samples. This is because that if dimensionality of the application domain increases, ECM clustering technique employed by DENFIS may form an unnecessarily large number of clusters to cover data samples diversely distributed in the high-dimensional space. Subsequently, DENFIS employs an unnecessarily large number of rules to construct an unnecessarily large network. Although RS-POPFNN and RS-HeRR employ Mamdani type of fuzzy rules, they suffer from similar problems. Employment of unnecessarily large number of rules to alleviate the curse of dimensionality problem is not encouraged because it greatly degrades the level of interpretability.

There is no significant difference between the geometric mean and arithmetic mean for each model, which suggests that all models are stable in this application. It is shown in Table 5-9 that GARSINFIS-a&i achieves the fourth highest testing accuracy among all NFISs. Single factor ANOVA is applied to the testing accuracy of GARSINFIS-a&i and DENFIS, which has the highest values among all NFISs (more stable than ANFIS). P -value is computed as $0.006 < 0.01$ and $F = 9.69 > F_{crit} = 4.41$. Both measurements indicate there is a difference between the performance of the two models. The difference ($96.97\% - 94.29\% = 2.68\%$) is approximately 1.59 data samples.

5.3 Accuracy Increase without Sacrificing Interpretability

In the previous section, zero-order TSK type of fuzzy rules are shown to perform better than crisp decision rules if they both are employed by GARSINFIS to perform inferences. The derived fuzzy rules possess a high level of interpretability because various terms to evaluate different aspects of interpretability are defined in the fitness function. In this section, it is shown that by employment of the fitness function defined in Equation 3-6 on Page 67, GARSINFIS increases accuracy but still maintain a relatively high level of interpretability. The coefficients defined in Equation 3-6 are both assigned to 1.

5.3.1 Thyroid Gland Disease Diagnosis Data Set

Thyroid gland disease diagnosis data set or thyroid diagnosis data set in short is downloadable from the UCI Machine Learning Repository [(UCI) 14]. The data set used in this thesis consists of 215 thyroid gland disease samples collected in Australia. All data samples are categorized into three different classes, namely normal, hyper-functioning, and hypo-functioning, which consist of 150, 35, and 30 data samples respectively. Each data sample consists of five different measurements, namely T3-resin uptake test, total serum thyroxine, total serum triiodothyronine, basal thyroid-stimulating hormone (TSH), and maximal absolute difference of the TSH value.

Because thyroid diagnosis data set is more complex than the two data sets introduced in the previous section, the population size of GARSC is set to 200 to examine more number of possible solutions and the number of generations is set to 80 to ensure convergence. Experimental results of applying GARSINFIS with accuracy driven fitness function on thyroid diagnosis data set is summarized in Table 5-10.

Table 5-10: Summary of applying GARSINFIS on thyroid diagnosis data set

Experiment Index	Training (%)	Testing (%)	Whole (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	99.30	94.44	97.68	3	4	9	551.40
2	99.30	94.44	97.68	2	4	7	636.95
3	98.60	97.18	98.13	3	5	8	572.16
4	97.92	97.22	97.69	3	4	7	537.76
5	97.22	94.44	96.29	4	5	9	616.89
6	97.90	95.83	97.21	4	5	10	589.05
7	98.61	94.44	97.21	4	4	8	563.83
8	98.61	91.55	96.26	3	5	9	676.72
9	98.60	98.59	98.60	3	3	6	593.22
10	97.92	95.83	97.22	3	4	9	658.17
mean	98.40	95.40	97.40	3.2	4.3	8.2	599.62
std	0.66	1.99	0.73	0.63	0.67	1.23	46.43

Experiment-2 utilizes the least number of input features which is two to achieve a non-lowest accuracy. In the other nine experiments, GARSC selects three or four features. Although the fitness function is accuracy driven only, the inference knowledge base derived by GARSC is still significantly simplified. On average, GARSINFIS utilizes 3.2 input features to achieve 95.4% accuracy on the testing data set and 97.4% accuracy on the whole data set.

Experiment-6 employs the largest number of rules and the largest number of arguments in the antecedent part of the rules. Although it does not achieve the highest accuracy, the constructed network is still relatively small in size. On average, GARSINFIS employs 4.3 rules and each rule

consists of $8.2/4.3 = 1.91$ arguments in the antecedent part.

To show that by the employment of fitness function which focuses on accuracy only, GARSINFIS-a increases accuracy when comparing to GARSINFIS-a&i, respective experiments are conducted and results are compared in Table 5-11 which also presents the benchmarks of the other models. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Table 5-11: Benchmarks on thyroid diagnosis data set

Model		Training (%)	Testing (%)	Whole (%)	Features	Rules	G-Mean _{test}
GARSINFIS-a	mean	98.40	95.40	97.40	3.2	4.3	95.38
	std	0.66	1.99	0.73	0.63	1.23	
GARSINFIS-a&i	mean	97.77	94.83	96.79	2.8	4.0	94.79
	std	0.97	2.21	0.56	0.79	0.67	
C4.5	mean	91.70	93.49	92.28	3.2	5.9	93.42
	std	2.60	2.09	1.41	1.03	0.99	
Naive Bayes	mean	96.30	97.91	96.84	5	N.A.	97.90
	std	0.75	0.98	0.48			
SVM	mean	87.72	89.96	88.47	5	N.A.	89.94
	std	0.92	1.94	0.57			
MLP	mean	95.32	96.79	95.81	5	7	96.78
	std	1.14	1.86	0.79			
RBF	mean	98.60	97.07	98.09	5	6	97.05
	std	1.47	1.39	0.64			
ANFIS	mean	97.84	91.21	95.63	5	11.6	91.18
	std	1.33	2.69	1.12		1.17	
DENFIS	mean	100.0	95.96	98.65	5	13.2	95.95
	std	0.00	1.52	0.51		0.79	
RS-POPFNN	mean	93.18	91.67	93.10	5	25.6	91.59
	std	1.56	4.16	1.63	0.00	2.91	
RS-HeRR	mean	95.53	92.88	94.65	3.8	29.5	92.86
	std	1.51	2.23	1.62	0.42	5.48	

According to Table 5-11, although GARSINFIS-a utilizes more number of features and rules than GARSINFIS-a&i does, GARSINFIS-a achieves higher accuracy on every data set. This shows that by transformation of the fitness function into an accuracy driven one, the derived knowledge base performs more accurate inferences. Furthermore, GARSINFIS-a only employs $3.2 - 2.8 = 0.4$ more number of input features and $4.3 - 4 = 0.3$ more number of fuzzy rules on average. This shows that although the fitness function is accuracy driven only, the size of the derived inference knowledge base is still significantly simplified. Moreover, GARSINFIS-a

employs less amount of knowledge than C4.5, RS-POPFNN, and RS-HeRR do and achieves better accuracy on every data set.

In terms of accuracy on the training data set, DENFIS performs best with the difference of $100\% - 98.4\% = 1.6\%$, which is approximately 2.3 data samples, when comparing to GARSINFIS-a. This difference is relatively small if the amount of the employed knowledge is taken into consideration.

In terms of accuracy on the testing data set, Naive Bayes classifier performs surprisingly well because it achieves the highest accuracy with the smallest stability. The difference between Naive Bayes classifier and GARSINFIS-a is $97.91\% - 95.4\% = 2.51\%$, which is approximately 1.8 data samples.

In terms of accuracy on the whole data set, DENFIS performs best due to its high accuracy on the training data set. The difference is $98.65\% - 97.4\% = 1.25\%$, which is approximately 2.7 data samples, when comparing to GARSINFIS-a.

In terms of interpretability, GARSINFIS-a&i performs better than all the other models because its fitness function evaluates both interpretability and accuracy. GARSINFIS-a employs a slightly larger knowledge base (same number of selected features as C4.5 but less number of rules) to achieve better accuracy on every data set in this application.

The geometric mean listed in the last column of Table 5-11 shows that GARSINFIS-a achieves the second highest testing accuracy among

all NFISs. Single factor ANOVA is applied to the testing accuracy of GARSINFIS-a and DENFIS, which has the highest values among all NFISs. P -value is computed as $0.486 > 0.05$ and $F = 0.505 < F_{crit} = 4.41$. Both measurements indicate there is no significant difference between the performance of the two models.

5.3.2 Ionosphere Detection Data set

Ionosphere detection data set is downloadable from the UCI Machine Learning Repository [(UCI) 14]. This data set consists of 351 radar signal arrays to identify whether there are certain structures in the ionosphere. This binary classification problem has 225 samples of Class I whose answers are yes and 126 samples of Class II whose answers are no. The data set used in this thesis has 32 input features that the first two features in the original data set are removed because they contain redundant information. Furthermore, to simplify the computations and to accelerate the speed, all data samples in the original data set are truncated to two significant figures.

The population size of GARSC is set to 200 to examine more number of possible solutions and the number of generations is set to 10. A larger number of generations should be assigned to ensure convergence but it requires longer time to complete. To maintain a reasonable computation time in this high-dimensional application, the number of generations is set to a small value. Experimental results of applying GARSINFIS which employs accuracy driven fitness function on ionosphere detection data set

is summarized in Table 5-12.

Table 5-12: Summary of applying GARSINFIS on ionosphere detection data set

Experiment Index	Training (%)	Testing (%)	Whole (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	92.74	88.03	91.17	7	16	43	5007.97
2	94.87	89.74	93.16	8	16	45	4698.63
3	97.44	92.31	95.73	10	20	73	5750.02
4	94.44	91.45	93.44	9	20	60	5415.83
5	93.59	90.60	92.59	9	22	63	5653.31
6	93.16	89.74	92.02	7	21	61	4308.19
7	92.31	87.18	90.60	6	15	42	5254.76
8	97.86	94.02	96.58	9	24	65	4792.27
9	94.02	93.16	93.73	9	19	54	5374.77
10	96.58	91.45	94.87	9	19	56	4983.40
mean	94.70	90.77	93.39	8.3	19.2	56.2	5123.92
std	1.96	2.16	1.92	1.25	2.86	10.27	450.55

Experiment-7 utilizes the least number of features and unfortunately achieves the lowest accuracy. On the other hand, experiment-3 utilizes the most number of features but it does not achieve the highest accuracy on any data set. On average, GARSINFIS utilizes 8.3 input features to achieve 90.77% accuracy on the testing data set and 93.39% accuracy on the whole data set.

Experiment-8 employs the largest number of rules and second largest number of arguments in the antecedent part of rules to achieve the highest accuracy on every data set. On average, GARSINFIS employs 19.2 rules and each rule consists of $56.2/19.2 = 2.93$ arguments in the antecedent part.

To show that by the employment of fitness function which focuses on accuracy only, GARSINFIS-a increases accuracy when comparing to GARSINFIS-a&i, respective experiments are conducted and results are compared in Table 5-13 which also presents the benchmarks of the other models. All models utilize the same pairs of training and testing data sets

for consistent comparisons.

Table 5-13: Benchmarks on ionosphere detection data set

Model		Training (%)	Testing (%)	Whole (%)	Features	Rules	G-Mean _{test}
GARSINFIS-a	mean	94.70	90.77	93.39	8.3	19.2	90.74
	std	1.96	2.16	1.92	1.25	2.86	
GARSINFIS-a&i	mean	93.59	90.51	92.56	7.9	18.9	90.49
	std	0.92	1.87	1.05	1.37	2.08	
C4.5	mean	97.99	90.60	95.53	7.6	11.5	90.54
	std	0.78	3.32	1.15	1.17	2.12	
Naive Bayes	mean	82.88	83.68	83.14	32	N.A.	83.62
	std	1.52	3.28	1.67			
SVM	mean	90.34	85.56	88.75	32	N.A.	85.50
	std	0.88	1.69	0.45			
MLP	mean	99.36	88.80	95.84	32	19	88.76
	std	0.22	2.92	0.93			
RBF	mean	93.63	90.60	92.62	32	6	90.56
	std	1.09	2.67	0.79			
ANFIS	mean	100.0	84.24	94.75	32	24.6	84.22
	std	0.00	1.85	0.62		3.84	
DENFIS	mean	99.91	80.17	93.33	32	93.6	80.11
	std	0.18	3.32	1.08		2.17	
RS-POPFNN	mean	98.21	82.39	92.94	27.1	151.3	82.36
	std	0.53	2.62	0.99	1.20	5.50	
RS-HeRR	mean	100.0	87.26	95.75	6.7	146.2	87.23
	std	0.00	2.56	0.85	2.16	15.0	

According to Table 5-13, GARSINFIS-a employs more amount of knowledge than GARSINFIS-a&i to achieve better accuracy. However, GARSINFIS-a is less stable than GARSINFIS-a&i except on the number of selected features. This is because GARSINFIS-a&i optimizes both accuracy and minimal structure. In this high-dimensional space, a more compact and more accurate model produces better stability. The difference in accuracy between GARSINFIS-a and GARSINFIS-a&i is $94.7\% - 93.59\% = 1.11\%$ on the training data set, which is larger than the difference of $90.77\% - 90.51\% = 0.26\%$ on the testing data set. This shows that although GARSINFIS-a achieves better accuracy, it still suffers from the over-fitting problem.

Except Naive Bayes classifier, all the other models suffer from the over-fitting problem in this application. The largest difference between accuracy on the training data set and the testing data set is $99.91\% - 80.17\% = 19.74\%$ produced by DENFIS. Although Naive Bayes classifier does not

over-fit on the training data set, it produces the lowest accuracy on the whole data set, i.e. Naive Bayes classifier maintains a good balance of generalization with underachieved accuracy.

In terms of accuracy on the training data set, ANFIS and RS-HeRR perform best that all training data samples are correctly classified. However, they achieve lower accuracy on the testing data set comparing to either GARSINFIS model. Both models suffer from the over-fitting problem, which is generally manifested as a relatively large decrease of the testing data set accuracy comparing to the training data set accuracy.

In terms of accuracy on the testing data set, GARSINFIS-a performs best. It appears that all models do not perform well to predict unseen data samples in this application. One of the reasons is due to the high dimensionality, which makes the models easier to be over-fitted. The importance of incorporating feature selection into the inference scheme is well demonstrated in Table 5-13 wherein all models with feature selection capability generally suffer less from the curse of dimensionality problem. Among all NFISs, it is shown that GARSINFIS and RS-HeRR employ less number of features (8.3 and 6.7 respectively) than ANFIS, DENFIS, and RS-POPFNN (32, 32, and 27.1 respectively) while performing better on the testing accuracy (90.77% and 87.26% against 84.24%, 80.17%, and 82.39% respectively).

In terms of accuracy on the whole data set, MLP network performs best which nearly produces perfect result on the training data set and achieves a high accuracy on the testing data set. The difference is 95.84%—

$93.39\% = 2.45\%$, which is approximately 8.6 data samples, when comparing to GARSINFIS-a.

In terms of interpretability, C4.5 utilizes lesser features and employs lesser rules than GARSINFIS does. Furthermore, C4.5 performs better in terms of accuracy on the whole data set. However, GARSINFIS-a produces a better level of generalization because its difference in accuracy between training and testing data sets ($94.7\% - 90.77\% = 3.93\%$) is smaller than that of C4.5 ($97.99\% - 90.6\% = 7.39\%$). Although RS-HeRR utilized the least number of features, it employs a significant large number of rules (146.2 comparing to 19.2 in GARSINFISa and 11.5 in C4.5). Moreover, RS-HeRR suffers more from the over-fitting problem than C4.5.

The geometric mean listed in the last column of Table 5-13 shows that GARSINFIS-a achieves the highest testing accuracy among all models.

5.4 Accuracy Increase in Complex Application Domains

In the previous section, GARSINFIS which employs accuracy driven only fitness function is shown to be capable of increasing accuracy by the employment of a slightly larger amount of inference knowledge. However, the amount of increase in accuracy is limited. In this section, experiments in more complex application domains are conducted to show that by the employment of first-order TSK type of fuzzy rules which are derived from

zero-order ones, GARSINFIS increases accuracy in compensation to the degradation of interpretability.

5.4.1 Glass Identification Data Set

Glass identification data set, which consists of the laboratory analyses of 214 different pieces of glass to identify their associated types, is downloadable from the UCI Machine Learning Repository [(UCI) 14]. There are nine values measured on each piece of glass, namely refractive index and chemical compositions of sodium, magnesium, aluminum, silicon, potassium, calcium, barium, and iron. The distribution of different types of glass is presented in Table 5-14. Because there is no data sample in “vehicle windows (non-float processed)” class, there are actually six different types of glass in this data set. Glass identification data set is not easy to be processed because some pieces of glass have similar measured values in the given features but they are used for different purposes.

Table 5-14: Distribution of data samples in glass identification data set

Type of glass	No. of samples	Type of glass	No. of samples
building windows (float processed)	70	containers	13
building windows (non-float processed)	76	tableware	9
vehicle windows (float processed)	17	headlamps	29
vehicle windows (non-float processed)	0		

The population size of GARSC is set to 200 and the number of generations is set to 20. Experimental results of applying GARSINFIS which employs first-order TSK type of fuzzy rules on glass identification data set is summarized in Table 5-15. The zero-order TSK type of fuzzy rules are converted to first-order ones by applying the Recursive Least Squares (RLS) algorithm which is defined by Equation set 4-12 on Page 90. The

conversion is performed after GARSC derives the best set of zero-order rules, i.e. RLS algorithm is applied to fine tune the rules optimized by GA. During parameter tuning, all training data samples are fed to GARSINFIS in sequence for one time only.

Table 5-15: Summary of applying GARSINFIS on glass identification data set

Experiment Index	Training (%)	Testing (%)	Whole (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	79.72	64.79	74.74	7	21	57	5783.47
2	86.71	74.65	82.69	8	19	54	5253.60
3	82.39	71.83	78.87	7	18	55	5289.85
4	79.02	66.67	74.90	7	20	61	5064.71
5	81.69	69.01	77.46	7	22	67	5751.72
6	77.62	63.38	72.87	8	20	62	5514.94
7	80.28	66.20	75.59	7	19	58	5636.51
8	75.35	62.50	71.07	6	17	51	5259.77
9	81.82	70.42	78.02	7	18	63	5623.52
10	80.42	70.83	77.22	7	21	65	5219.49
mean	80.50	68.03	76.34	7.1	19.5	59.3	5439.76
std	3.04	3.95	3.28	0.57	1.58	5.14	252.17

Experiment-8 utilizes the least number of features which is six. However, it achieves the lowest accuracy on all data sets. This indicates that it is dangerous to select less number of input features in this complex application domain. Between the two experiments which utilize the most number of features which is eight, experiment-2, which achieves the highest accuracy on all data sets, is more successful than experiment-6. On average, GARSINFIS utilizes 7.1 input features to achieve 68.03% accuracy on the testing data set and 76.34% accuracy on the whole data set.

Experiment-8 which employs the least number of rules achieves the lowest accuracy on all data sets. On the other hand, experiment-5 which employs the largest number of rules does not achieve the highest accuracy on all data sets. On average, GARSINFIS employs 19.5 first-order TSK type of fuzzy rules and each rule consists of $59.3/19.5 = 3.04$ arguments in

the antecedent part.

To show that by the employment of first-order TSK type of fuzzy rules, GARSINFIS-1 increases accuracy when comparing to GARSINFIS-a which employs zero-order rules, respective experiments are conducted and results are compared in Table 5-16 which also presents the benchmarks of the other models. All models utilize the same pairs of training and the testing data sets for consistent comparisons.

Table 5-16: Benchmarks on glass identification data set

Model		Training (%)	Testing (%)	Whole (%)	Features	Rules	G-Mean _{test}
GARSINFIS-1	mean	80.50	68.03	76.34	7.1	19.5	67.92
	std	3.04	3.95	3.28	0.57	1.58	
GARSINFIS-a	mean	78.29	66.63	74.40	7.1	19.5	66.53
	std	2.36	3.86	1.32	0.57	1.58	
C4.5	mean	62.44	66.48	63.79	8.4	20.4	66.58
	std	4.10	4.88	3.09	0.70	3.63	
Naive Bayes	mean	46.32	49.09	47.24	9	N.A.	48.92
	std	4.22	4.29	3.58			
SVM	mean	61.53	57.93	60.33	9	N.A.	57.60
	std	4.08	6.32	4.63			
MLP	mean	65.62	67.18	66.07	9	13	67.07
	std	3.20	4.27	2.15			
RBF	mean	80.24	66.75	75.74	9	12	66.59
	std	3.43	4.68	2.26			
ANFIS	mean	83.04	60.17	75.42	9	7.1	59.84
	std	3.08	6.23	2.90		0.88	
DENFIS	mean	84.37	63.53	77.43	9	21.6	63.38
	std	2.43	4.76	2.21		0.97	
RS-POPFNN	mean	78.68	62.03	73.13	9	102	61.87
	std	5.70	4.77	4.80	0.00	5.87	
RS-HeRR	mean	93.56	64.80	83.97	7.1	93.7	64.69
	std	3.13	4.02	2.57	1.10	3.13	

According to Table 5-16, GARSINFIS-1 performs better than GARSINFIS-a in terms of accuracy. Because GARSINFIS-1 is extended from GARSINFIS-a, they have the same number of selected features and rules. The difference only lies in the consequent parts of rules where GARSINFIS-1 increases accuracy by employing less comprehensible rules.

In terms of accuracy on the training data set, RS-HeRR performs best in accuracy. However, it suffers severely from the over-fitting problem that the difference between training and testing data sets is $93.56\% - 64.80\% = 28.76\%$. That difference of GARSINFIS-1 is $80.5\% - 68.03\% = 12.47\%$.

In terms of accuracy on the testing data set, GARSINFIS-1 performs best.

In terms of accuracy on the whole data set, RS-HeRR performs best. However, it is mainly due to its high accuracy on the training data set.

In terms of interpretability, both GARSINFIS and RS-HeRR utilize the least number of input features. However, RS-HeRR employs a significantly larger number of rules (93.7) than GARSINFIS does (19.5). Although ANFIS employs the least number of rules, extra amount of efforts are made to determine the number of clusters, which is required to run the FCM clustering technique employed by ANFIS, through trial-and-error (not automatically determined but requires multiple trials because FCM is random).

The geometric mean listed in the last column of Table 5-16 shows that GARSINFIS-1 achieves the highest testing accuracy among all models.

5.4.2 Material Discrimination Data Set

Material discrimination data set is downloadable from the UCI Machine Learning Repository [(UCI) 14]. This data set consists of sonar signals bounced off a metal cylinder and a rock cylinder with rough surface. There are 111 and 97 samples collected on the two different cylinders respectively at various angles and under various conditions. Each sample consists of 60 measurements valued in the $[0\ 1]$ interval where each represents the energy within a particular frequency band and integrated over a certain period of

time.

The population size of GARSC is set to 200 and the number of generations is set to 20. Experimental results of applying GARSINFIS which employs first-order TSK type of fuzzy rules on material discrimination data set is summarized in Table 5-17.

Table 5-17: Summary of applying GARSINFIS on material discrimination data set

Experiment Index	Training (%)	Testing (%)	Whole (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	92.09	76.81	87.00	9	23	74	6817.46
2	94.96	75.36	88.43	8	22	65	7295.84
3	92.75	72.86	86.12	10	25	82	6942.71
4	90.58	67.14	82.77	8	19	60	7163.60
5	91.37	75.36	86.03	9	25	77	7206.93
6	94.24	71.01	86.50	9	17	53	6742.47
7	92.81	72.46	86.03	8	22	63	7085.61
8	93.53	69.57	85.54	9	17	46	6694.49
9	92.03	71.43	85.16	8	19	55	6973.33
10	92.81	73.91	86.51	8	22	68	6650.52
mean	92.72	72.59	86.01	8.6	21.1	64.3	6957.30
std	1.30	2.94	1.44	0.70	2.96	11.32	227.38

Experiment-3 utilizes the most number of features which is ten. However, it does not achieve the highest accuracy on all data sets. On average, GARSINFIS utilizes 8.6 input features (out of 60) to achieve 72.59% accuracy on the testing data set and 86.01% accuracy on the whole data set. Moreover, on average, it employs 21.1 first-order TSK type of fuzzy rules and each rule consists of $64.3/21.1 = 3.05$ arguments.

To show that by the employment of first-order TSK type of fuzzy rules, GARSINFIS-1 increases accuracy when comparing against GARSINFIS-a which employs zero-order rules, respective experiments are conducted and results are compared in Table 5-18 which also presents the benchmarks of the other models. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Table 5-18: Benchmarks on material discrimination data set

Model		Training (%)	Testing (%)	Whole (%)	Features	Rules	G-Mean _{test}
GARSINFIS-1	mean	92.72	72.59	86.01	8.6	21.1	72.54
	std	1.30	2.94	1.44	0.70	2.96	
GARSINFIS-a	mean	90.70	69.38	83.53	8.6	21.1	69.19
	std	1.39	5.36	2.08	0.70	2.96	
C4.5	mean	97.84	70.82	88.83	11.6	13.6	70.58
	std	0.59	6.25	2.23	1.58	1.17	
Naive Bayes	mean	73.78	70.26	72.60	60	N.A.	70.01
	std	2.88	6.27	2.60			
SVM	mean	88.40	77.31	84.70	60	N.A.	77.21
	std	2.47	4.11	1.54			
MLP	mean	99.21	81.21	93.21	60	33	81.14
	std	3.20	4.27	2.15			
RBF	mean	96.47	81.20	91.38	60	10	81.14
	std	2.24	3.13	2.09			
ANFIS	mean	100.0	73.84	91.28	60	4.4	73.75
	std	0.00	3.73	1.24		0.88	
DENFIS	mean	98.92	77.03	91.62	60	71.7	76.90
	std	0.91	4.43	1.43		1.64	
RS-POPFNN	mean	100.0	70.01	90.00	24.2	137.3	69.78
	std	0.00	6.01	2.00	6.43	2.41	
RS-HeRR	mean	100.0	72.85	90.95	6.1	120.7	72.60
	std	0.00	6.24	2.08	1.20	5.10	

According to Table 5-18, GARSINFIS-1 performs better than GARSINFIS-a in terms of accuracy and stability. It should be noted that on average, they only select 8.6 input features (out of 60) to achieve a medium level of testing accuracy among all models.

In terms of accuracy on the training data set, ANFIS, RS-POPFNN, and RS-HeRR perform best that all training data samples are correctly classified. However, they suffer more severely from the over-fitting problem than GARSINFIS does.

In terms of accuracy on both the testing and the whole data sets, MLP performs best. However, MLP utilizes all the 60 input features and employs 33 hidden neurons where the associated weight matrices cannot be semantically interpreted.

In terms of interpretability, RS-HeRR utilizes the least number of input features but again employs a significantly large number of rules. On the other hand, ANFIS employs the least number of rules. However, extra amount of efforts are made to determine the number of clusters, which is

required to run the FCM clustering technique employed by ANFIS, through trial-and-error. In this high dimensionality application domain, the number of clusters assigned to FCM is capped at five or six. Otherwise, ANFIS will fail due to the insufficient number of training samples.

The geometric mean listed in the last column of Table 5-18 shows that GARSINFIS-1 achieves the fourth highest testing accuracy among all NFISs. Single factor ANOVA is applied to the testing accuracy of GARSINFIS-1 and DENFIS, which has the highest values among all NFISs. P -value is computed as $0.017 > 0.01$ and $F = 6.95 > F_{crit} = 4.41$. Both measurements suggest there is a difference between the performance of the two models. The difference ($76.90\% - 72.54\% = 4.36\%$) is approximately 3.02 data samples.

5.5 Accuracy Increase by Formation of a Committee

The employment of multiple models to form a committee is one way to adaptively increase the overall performance because the individual modeling bias is reduced. Normally, these models are of the same type. The committee is either local if all models are trained with the same data set or global if the models are trained with different data subsets. The increase in accuracy by the formation of a committee is due to the weighted voting scheme, not because the individual models are improved.

Suppose a committee consists of N models, then initially every model is equally weighted, i.e. $W_i = \frac{1}{N}, i = 1, \dots, N$. When a training sample and its desired value d_i are given, the weight of each model in the committee is updated according to Equation 5-1 and the prediction of committee y is defined in Equation 5-2.

$$w_i^{t+1} = \begin{cases} w_i^t(1 + \beta), & \text{if } y_i = d_i, \\ w_i^t(1 - \beta), & \text{otherwise,} \end{cases} \quad (5-1)$$

$$y = \frac{\sum_{i=1}^N w_i y_i}{\sum_{i=1}^N w_i}, \quad (5-2)$$

where β denotes the learning rate and y_i denotes the prediction value of the i th model.

5.5.1 Diabetes Diagnosis Data Set

Diabetes diagnosis data set, which consists of data samples collected from Pima Indian female patients who are at least 21 years old, is downloadable from the UCI Machine Learning Repository [(UCI) 14]. Each collected sample has eight input features which are listed in Table 5-19. There are altogether 768 data samples in the original data set but some samples are suspected to have missing values. According to Table 5-19, all features should not have zero values except the first one which is the number of pregnant times. All data samples which have zero values on any feature except the first one are removed from the original data set. The data set used in this thesis consists of 392 data samples that 262 of them are

diagnosed as “negative” and the other 130 are “positive”.

Table 5-19: Input features of diabetes diagnosis data set

1	Pregnant times	5	Serum insulin
2	Plasma glucose concentration	6	Body mass index
3	Diastolic blood pressure	7	Diabetes pedigree function
4	Triceps skin fold thickness	8	Age

The population size of GARSC is set to 200 and the number of generations is set to 20. Experimental results of applying GARSINFIS committee which consists of ten best performing networks on diabetes diagnosis data set is summarized in Table 5-20. Because GA provides multiple solutions instead of a single one, there is no extra effort involved in training GARSINFIS for multiple times. The ten networks are selected from the last generation when GA terminates. Because elitism strategy is applied, these ten selected networks are the best performing ones among all the evaluated solutions. The ten networks are all trained with the same set of data set. Therefore, the committee is local. The committee is trained once by sequentially applying all data samples in the training data set with $\beta = 0.05$. Unlike the previously shown summaries of performance, the number of features presented in Table 5-20 is the total number of features utilized in the committee. However, the number of rules and arguments presented in Table 5-20 are the average values of the individual committees.

Although many individual GARSINFIS networks utilize less number of input features, all GARSINFIS committees utilize all eight input features. Furthermore, the numbers of rules and arguments employed in the committees are stable because they are taken from the average of ten individual networks. On average, there are $98.98/32.91 = 3.01$ arguments employed

Table 5-20: Summary of applying GARSINFIS committee on diabetes diagnosis data set

Experiment Index	Training (%)	Testing (%)	Whole (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	87.36	79.39	84.70	8	31.2	103.9	5911.73
2	90.08	79.23	86.46	8	33.7	99.8	5895.74
3	91.57	78.63	87.26	8	32.3	101.9	6073.69
4	90.84	79.23	86.97	8	33.4	97.6	6391.81
5	91.22	82.31	88.25	8	34.4	95.2	5712.08
6	88.51	77.86	84.96	8	31.8	97.5	6267.15
7	91.60	83.85	89.02	8	33.2	101.8	5835.23
8	92.33	82.44	89.03	8	34.2	99.6	6346.54
9	91.98	82.31	88.76	8	31.6	97.1	6118.03
10	90.84	77.69	86.46	8	33.3	95.4	6005.85
mean	90.63	80.29	87.19	8	32.91	98.98	6055.79
std	1.58	2.21	1.58	0.00	1.12	2.92	226.56

in the antecedent part of each rule.

To show that by the formation of a local committee, GARSINFIS-c increases accuracy when comparing to GARSINFIS-a which employs zero-order TSK type of rules and accuracy driven only fitness function, respective experiments are conducted and results are compared in Table 5-21 which also presents the benchmarks of the other models. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Table 5-21: Benchmarks on diabetes diagnosis data set

Model		Training (%)	Testing (%)	Whole (%)	Features	Rules	G-Mean _{test}
GARSINFIS-c	mean	90.63	80.29	87.19	8	32.91	80.27
	std	1.58	2.21	1.58	0.00	1.12	
GARSINFIS-a	mean	89.03	78.84	85.63	6.6	32.2	78.81
	std	1.46	2.10	1.40	0.70	6.30	
C4.5	mean	89.81	75.48	85.03	6.8	16.6	75.41
	std	4.06	3.40	2.72	1.14	4.50	
Naive Bayes	mean	80.28	77.31	79.29	8	N.A.	77.26
	std	6.11	2.86	3.59			
SVM	mean	79.47	78.46	79.13	8	N.A.	78.43
	std	1.31	2.28	0.46			
MLP	mean	89.05	77.00	85.03	8	7	76.97
	std	2.17	2.50	1.12			
RBF	mean	79.93	77.39	79.08	8	6	77.35
	std	2.36	2.57	1.22			
ANFIS	mean	92.65	70.67	85.32	8	16.8	70.62
	std	2.14	2.80	1.53		1.87	
DENFIS	mean	97.43	72.42	89.10	8	55.5	72.39
	std	1.88	2.28	1.41		2.76	
RS-POPFNN	mean	90.62	71.05	84.10	8	69.3	71.04
	std	1.53	1.18	1.21	0.00	1.64	
RS-HeRR	mean	100.0	75.82	91.94	5.0	248.4	75.66
	std	0.00	5.09	1.70	1.25	6.31	

According to Table 5-21, GARSINFIS-c performs better than GARSIN-

FIS-a in terms of accuracy on all data sets because individual modeling bias is reduced by the formation of a voting committee. Furthermore, it is encouraging to learn that even GARSINFIS-a performs better than all the other models in terms of accuracy on the testing data set, which shows the integrities of the derived fuzzy rule sets.

In terms of accuracy on both the training and the whole data sets, RS-HeRR performs best. However, it severely suffers from the over-fitting problem.

In terms of interpretability, RS-HeRR utilizes the least number of features but again employs a significant large number of rules. C4.5 employs the least number of rules. Although GARSINFIS-c employs more number of features than GARSINFIS-a, accuracy on every data set is improved. GARSINFIS employs more number of rules than C4.5 and ANFIS which employs the FCM clustering technique, but GARSINFIS achieves a better level of inference accuracy and generalization. The employment of more rules is essential to achieve a high level of accuracy in this application.

The geometric mean listed in the last column of Table 5-11 shows that GARSINFIS-a achieves the highest testing accuracy among all models.

5.5.2 Colon Cancer Diagnosis Data Set

Colon cancer diagnosis data set is downloadable from the Kent Ridge Biomedical Data Set Repository [(KRBD) 14]. There are 62 data samples collected from colon cancer patients. Particularly, 40 samples are collected

from tumor biopsies and the other 22 are from normal ones. Each data sample is a gene expression array which has 2000 features. The capability of GARSINFIS in terms of feature selection is ultimately assessed.

The population size of GARSC is set to 300 and the number of generations is set to 30. Experimental results of applying GARSINFIS committee which consists of ten best performing networks on the diabetes diagnosis data set is summarized in Table 5-22. Unlike the experiment in the previous subsection, the committee in this section is global which means GARSINFIS networks are trained on different data sets. The committee is trained once by sequentially applying the union of data samples in all the training data sets which is practically the whole data set with $\beta = 0.1$. The ten GARSINFIS networks listed in Table 5-22 form a global committee whose performance are presented in the last two rows. The last column presents the voting weights associated with each individual GARSINFIS network in the committee.

Table 5-22: Summary of applying GARSINFIS committee on colon cancer diagnosis data set

Experiment Index	Training (%)	Testing (%)	Whole (%)	Number of features	Number of rules	Number of arguments	Time (seconds)	Voting weight
1	97.62	80.00	91.75	5	8	17	3552.40	13.51
2	97.56	76.19	90.44	4	7	19	3118.17	11.05
3	95.12	71.43	87.22	6	9	21	3456.45	7.40
4	97.56	76.19	90.44	5	8	15	3472.04	11.05
5	92.86	70.00	85.24	5	8	17	3239.13	6.05
6	92.68	76.19	87.13	4	7	13	3703.42	7.40
7	97.56	71.43	88.85	6	8	20	3269.39	9.04
8	95.24	75.00	88.49	5	9	23	3117.55	9.04
9	92.86	75.00	86.91	4	7	18	3527.67	7.40
10	95.12	80.95	90.40	5	8	17	3495.23	11.05
mean	95.42	75.24	88.69	4.9	7.9	18	3395.15	9.30
std	2.09	3.57	2.05	0.74	0.74	2.91	197.47	2.32
committee	96.16	76.71	89.68	35	-	-	-	-
std	1.65	2.88	1.38	-	-	-	-	-

According to Table 5-22, the amount of increase in accuracy by the formation of a global committee is smaller than expectation. The reason is that the data set has small number of samples and all individual mod-

els have similar performance in accuracy. Please note that there are 35 features actually employed by the global committee which is smaller than the total number of 49 because some features are employed in more than one models. In practice, global committee works well in a time-varying application domain because model which is trained on a new set of data samples can replace the worst performing model in the committee. In such a way, the global committee is periodically updated with the time-varying knowledge and always maintains a high level of accuracy.

Experiments are also conducted on other benchmarking models and results are compared in Table 5-23. All models utilize the same pairs of training and testing data sets for consistent comparisons. Please note that some models which were previously benchmarked are not included in Table 5-23, because they cannot handle the high dimensionality of this colon cancer diagnosis application. Experiments on MLP network run out of system memory. ANFIS and DENFIS cannot converge because there is not sufficient amount of training samples in this high-dimensionality application.

Table 5-23: Benchmarks on colon cancer diagnosis data set

Model		Training (%)	Testing (%)	Whole (%)	Features	Rules	G-Mean _{test}
GARSINFIS-c	mean	96.16	76.71	89.68	4.9	7.9	76.67
	std	1.65	2.88	1.38	0.74	0.74	
GARSINFIS-a	mean	95.42	75.24	88.69	4.9	7.9	75.16
	std	2.09	3.57	2.05	0.74	0.74	
C4.5	mean	98.79	72.62	90.06	2.6	3.6	72.30
	std	1.72	7.05	3.07	0.52	0.52	
Naive Bayes	mean	73.25	57.79	68.09	2000	N.A.	57.30
	std	4.81	7.91	4.61			
SVM	mean	100.0	83.64	94.55	2000	N.A.	83.57
	std	0.00	4.65	1.55			
RBF	mean	99.03	75.98	91.35	2000	6	75.60
	std	1.70	7.79	2.27			
RS-POPFNN	mean	98.78	70.07	89.21	22.4	40.2	69.20
	std	2.37	11.00	3.38	10.89	1.55	
RS-HeRR	mean	100.0	71.64	90.55	3.3	24.9	71.07
	std	0.00	9.34	3.12	0.95	4.33	

It is worth mentioning that SVM performs best in terms of accuracy

on all the data sets. This finding is not surprising that first of all, in this binary classification application, the number of input features (2000) is significantly larger than the total number of data samples (62). SVM used in this thesis performs binary classifications. Therefore, it is relatively easier for SVM to correctly classify all the training data samples in the high dimensional space. Moreover, SVM is known to possess a high level of generalization. Therefore, it achieves the highest testing accuracy among all the models. In consequence, its accuracy on the whole data set is also the highest.

In terms of interpretability, C4.5 performs surprisingly well because it only utilizes 2.6 input features and employs 3.6 rules on average. Each C4.5 utilizes two or three number of features and the total number of features is 17. In comparison, GARSINFIS-c utilizes averagely $4.9 - 2.6 = 2.3$ more features and $7.9 - 3.6 = 4.3$ more rules to achieve $76.21\% - 72.62\% = 3.59\%$ better accuracy on the testing data set which is approximately 0.7 data sample. Nevertheless, it is encouraging that GARSINFIS-c only utilizes 4.9 input features out of the total number of 2000 to achieve a relatively high accuracy.

The geometric mean listed in the last column of Table 5-23 shows that GARSINFIS-c achieves the highest testing accuracy among all NFISs.

5.6 Results Analyses on all Data Sets

In the previous four sections, eight public downloadable data sets are applied to GARSINFIS and other benchmarking models. In Section 5.2, iris classification and wine recognition data sets are applied to show the necessity of translation from crisp decision rules to fuzzy inference rules in GARSINFIS. In Section 5.3, thyroid gland disease diagnosis and ionosphere detection data sets are applied to show the accuracy increase of GARSINFIS when fitness function is modified to be accuracy driven only. In Section 5.4, glass identification and material discrimination data sets are applied to show the accuracy increase of GARSINFIS when zero-order TSK type of fuzzy rules are extended to first-order ones. In Section 5.5, diabetes diagnosis and colon cancer diagnosis data sets are applied to show the accuracy increase of GARSINFIS by formation of either a local or a global committee. These data sets vary in size and GARSINFIS employs different amounts of essential knowledge to perform inferences on these data sets. The summary of applying GARSINFIS to various data sets is presented in Table 5-24.

Table 5-24: Summary of applying GARSINFIS on various data sets

Data set	No. of input features	No. of data samples	Population size	No. of iterations	No. of selected features	No. of employed rules	Accuracy on the whole data set (%)	Computation time (seconds)
iris classification	4	150	100	30	2.3	3.4	98.13	50.40
wine recognition	13	178	100	60	3.5	5.6	98.09	1434.17
thyroid diagnosis	5	215	200	80	3.2	4.3	97.40	599.62
ionosphere detection	32	351	200	10	8.3	19.2	93.39	5123.92
glass identification	9	214	200	20	7.1	19.5	76.34	5439.76
material discrimination	60	208	200	20	8.6	21.1	86.01	6957.30
diabetes diagnosis	8	392	200	20	8	32.91	87.19	6055.79
colon cancer diagnosis	2000	62	300	30	35	7.9	89.68	3395.15

According to Table 5-24, each GARSINFIS experiment completes within two hours in general (specification of the computers to conduct all experi-

ments are given in the last paragraph of Section 5.1.1). To better illustrate the performance of GARINFIS on various data sets, the following figures visualize the benchmarks in various aspects. The benchmarks of all models on accuracy is visualized in Figure 5-6 which illustrates the relative levels of generalization according to Equation 5-3a, Figure 5-7 which illustrates the relative accuracy on the whole data set according to Equation 5-3b, and Figure 5-8 which illustrates the relative accuracy on the testing data set according to Equation 5-3c. The benchmarks of all models on interpretability is visualized in Figure 5-9 which illustrates the relative number of employed rules according to Equation 5-3d and Figure 5-10 which illustrates the relative number of selected features according to Equation 5-3e. Please note that respective comparisons on the colon cancer diagnosis data set is not included in certain figures because not all models can be applied to this high dimensional application (see Section 5.5.2 for all necessary details).

$$g_{\text{relative}}^i = (a_{\text{train}}^i - a_{\text{test}}^i) / (a_{\text{train}}^G - a_{\text{test}}^G), \quad (5-3a)$$

$$a_{\text{relative}}^i = a_{\text{whole}}^i / a_{\text{whole}}^G, \quad (5-3b)$$

$$t_{\text{relative}}^i = t^i / t^G, \quad (5-3c)$$

$$r_{\text{relative}}^i = r^i / r^G, \quad (5-3d)$$

$$f_{\text{relative}}^i = f^i / f^G, \quad (5-3e)$$

where i denotes the i th model; a denotes accuracy; $t = a_{\text{test}}$; and G denotes GARSINFIS.

Generally speaking, NFISs are powerful learning machines which nor-

Figure 5-6: Visualization of relative level of generalization

mally suffer from the over-fitting problem, which is manifested as a significant gap between the accuracy on the training and testing data sets. According to Figure 5-6, GARSINFIS has a middle level of generalization among all models (points above the baseline of GARSINFIS denote inferior generalizations and vice versa). It is noticeable that across all data sets and among all models, GARSINFIS performs relatively better for ionosphere (a high dimensional application) and inferior for wine (reduction of inference knowledge leads to incorrect classifications on the testing data set, which is a problem also faced by RS-POPFNN and RS-HeRR). Because reduction is performed to remove all unnecessary inference knowledge, GARSINFIS employs a relatively compact inference rule base (see Figure 5-9) that are less likely to be over-fitting when compared to the other benchmarking NFISs. As expected, Naive Bayes classifier performs best in terms of gen-

eralization. However, also as expected, it does not perform well in terms of accuracy on the whole data set as shown in Figure 5-7.

Figure 5-7: Visualization of relative accuracy on the whole data set

The accuracy on the whole data set roughly examines the correctness and completeness of the constructed models to represent the whole data set. It is shown in Figure 5-7 that GARSINFIS is always one of the best performing models for the first three data sets and in the middle level for the rest. Among all the NFISs for the last five data sets, RS-HeRR performs best in terms of accuracy on the whole data set. However, according to Figure 5-6, RS-HeRR always performs inferior in terms of the level of generalization when comparing to GARSINFIS. This finding infers that the high level of accuracy on the whole data set achieved by RS-HeRR is mainly contributed by its high level of accuracy on the training data set. To clearly visualize how explicit rule-based models perform in terms of ac-

curacy on the testing data set, Figure 5-8 is used to compare the inference capabilities of all NFISs models and C4.5.

Figure 5-8: Visualization of relative accuracy on the testing data set

It is clearly shown in Figure 5-8 that GARSINFIS performs best among all NFISs for most data sets except inferior than ANFIS and DENFIS for wine recognition and material discrimination (ANOVA verifies that there is no significant difference for thyroid gland disease diagnosis, see Section 5.3.1). However, although all employ TSK type of fuzzy rules, ANFIS does not self-organize its network structure and DENFIS employs more rules (also more difficult to comprehend, see Figures 5-2 and 5-3 on Page 104) than GARSINFIS does (see Figure 5-9). Moreover, both ANFIS and DENFIS do not perform feature selection and they cannot handle the high dimensional data set such as colon cancer diagnosis. It is also encouraging to find out that among all the three NFISs that perform inference

Figure 5-9: Visualization of relative number of employed rules

knowledge reduction, GARSINFIS always produce similar or better testing accuracy than RS-POPFNN and RS-HeRR.

It is clearly shown in Figure 5-9, C4.5, GARSINFIS, and ANFIS utilize less rules than the other models. However, C4.5 utilize crisp rules (generally sensitive to small changes in the training data set and cannot handle unseen data samples well) which leads to its inferior testing accuracy (see Figure 5-8) and ANFIS (major credit of the employment of the relatively less number of rules should be given to FCM) require extra efforts to carefully select the optimal number of pre-defined clusters through trial-and-error. Moreover, it is clearly shown in Figure 5-9 that all NFISs that employ TSK type of fuzzy rules (GARSINFIS, ANFIS, and DENFIS) produce lesser rules than those employ Mamdani rules (RS-POPFNN and RS-HeRR).

Figure 5-10: Visualization of relative number of selected features

Among all benchmarking models, only four of them (shown in Figure 5-10) perform feature selection. Please note that the colon cancer diagnosis data set has 2000 input features which is not used as the value for “the other models” in Figure 5-10. Otherwise, the relative relation for the other data sets will not be illustrated distinguishably. Although RS-HeRR selects similar or slightly smaller number of features than GARSINFIS does, RS-HeRR employs significantly more number of rules (see Figure 5-9).

5.7 Summary

In this Chapter, results of extensive experiments are reported. There are eight public downloadable data sets applied to all benchmarking models for

performance comparisons. The performance of GARSINFIS is encouraging when compared to the other benchmarking models because it handles all data sets efficiently and accurately (see Figure 5-7) without the distinction between low and high dimensional applications. Because GARSINFIS only employs the most essential inference knowledge base, it achieves a relatively better level of generalization between the training and testing data sets than the other NFISs (see Figure 5-6). Although GARSINFIS does not achieve the highest testing accuracy in all application domains (highest in six out of the total number of eight, see Figure 5-8), it selects lesser input features (see Figure 5-10) and employs lesser rules (see Figure 5-9) with strong semantic meanings (see Figure 5-2 on Page 104) to achieve a high level of accuracy. GARSINFIS is empirically proven to produce satisfactory accuracy with relatively less amount of inference knowledge compared to the other models. Although GARSINFIS does not achieve the highest testing accuracy among all the benchmarking models in every experiment, in most of the experiments (all but wine recognition and material discrimination), it achieves either the highest testing accuracy or its testing accuracy is not different from the winner (verified by ANOVA for thyroid gland disease diagnosis) among all NFISs. More encouragingly, it performs better than RS-POPFNN and RS-HeRR, which also incorporate rough set approximations to perform inference knowledge reduction, in terms of both accuracy and interpretability.

Although genetic algorithm is considered to be time consuming, the recorded computation times in this chapter show that GARSINFIS handles all application domains efficiently and accurately within two hours

(specification of the computers to conduct all experiments are given in the last paragraph of Section 5.1.1). This implies that GARSINFIS achieves highly competitive results among all NFISs in terms of both accuracy and interpretability within reasonable time.

It may seem that many parameters are required in GARSINFIS. Actually, all the conducted experiments share a large number of common parameters. The only parameters that differ in applications are the population size, the number of generations, and the coefficients of the fitness evaluation function. These parameters decide the approximate number of solutions to be evaluated and the objectives that user wants to achieve which have to be differentiated in applications. Therefore, GARSINFIS requires limited parameter settings to perform systematic optimizations.

GARSINFIS performs feature selection and attribute reduction. Therefore, it utilizes a relatively small number of input features and employs a relatively small number of arguments in the antecedent part of each rule. Furthermore, a relatively small number of fuzzy membership functions, which is at most three in all applications, is systematically formulated in each selected input feature. Because a small number of membership functions are formulated in the original input features without normalizations, semantic meanings of the associated fuzzy labels are well preserved. Because inconsistent and redundant rules are removed, GARSC derives a relatively small number of interpretable fuzzy rules. The derived compact inference knowledge base is preferred by both experts and users because they can understand how GARSINFIS performs accurate inferences.

Users are able to achieve different multi-objectives by the assignments of different values to coefficients of the fitness function defined in GARSC. Furthermore, users are able to increase accuracy of GARINFIS in their own preferred ways. Among all models presented in this thesis (although there are clustering methods incorporate both GA and rough set approximations in the literature, they do not allow the leverage between accuracy and interpretability, see the last paragraph of Section 2.4.3), GARSINFIS is the only one which allows these kinds of flexibilities.

Chapter 6

Real-World Applications

“The best theory is inspired by practice.

The best practice is guided by theory.”

- Donald Knuth

This chapter further evaluates the performance of GARSINFIS by conducting experiments on two self-collected real-world applications. One is in the category of medical diagnosis and the other is in the category of financial analysis. These two applications are used to demonstrate the capability of GARSINFIS to maintain a good balance between accuracy and interpretability. More importantly, the derived inference knowledge bases are compared against the expert knowledge.

6.1 Ovarian Cancer Diagnosis System

6.1.1 Overview of Ovarian Cancer Diagnosis Application

Ovarian cancer grows when a tumor forms in one or both ovaries of a woman. It accounts for 4% of all cancers among women and ranks fourth as a cause of their deaths from cancer [(ACS) 14]. Ovarian cancer is completely curable if it is discovered early. Unfortunately, 70% are late, which increases the high death rate to some extent. Statistics from [(eMCH) 14] shows that the five-year survival rate is greater than 75% if diagnosis occurs before the cancer cells have spread to other organs which is referred as stage I. However, the survival rate drops to 20% when the cancer cells have spread to upper abdomen which is referred as stage III. Therefore, it is crucial to detect ovarian cancer as early as possible and to correctly identify the stage of ovarian cancer to prevent any further delay of appropriate treatment [(NLM) 14].

The ovarian cancer diagnosis data set used in this thesis is collected from Department of Oncology and Gynecology, National University Hospital, Singapore. This data set consists of 171 patient samples and adopts the staging system defined by International Federation of Gynecology and Obstetrics [(FIGO) 14]. All samples are categorized into one of the following five groups: “normal” which consists of 22 instances, “benign” which is short for benign cyst and consists of 78 instances, “borderline” which is short for proliferating tumors and consists of 10 instances, “stage I &

II ” which consists 19 instances, and “stage III & IV” which consists 42 instances. The seriousness of a patient is determined by 28 features which consist of the age of the patient and certain blood test results. The list of all features is given in Table 6-1.

Table 6-1: Input features of ovarian cancer diagnosis data set

1	Age	15	Antithrombin III Antigen (ATAG)
2	Blood Platelet	16	Plasminogen
3	Packed Cell Volume (PCV)	17	D-Dimer
4	Hemoglobin (Hgb)	18	Tissue Plasminogen Activator (tPA) Activity
5	β -Thromboglobulin (BTG)	19	Tissue Plasminogen Activator (tPA) Antigen
6	Reaction Time of Thrombelastography (TEG-R)	20	Urokinase-like Tissue Plasminogen Activator (uPA) Activity
7	Coagulation Time of Thrombelastography (TEG-K)	21	Urokinase-like Tissue Plasminogen Activator (uPA) Antigen
8	Maximum Amplitude of Thrombelastography (TEG-MA)	22	Urokinase-like Tissue Plasminogen Activator (uPA) Receptor
9	Fibrinogen	23	Plasminogen Activator Inhibitor (PAI) Activity
10	Factor VII	24	Plasminogen Activator Inhibitor (PAI) Antigen
11	Von Willebrand-Factor (VWF)	25	Protein C Antigen (PCAg)
12	Thrombin-Antithrombin Complex (T/AT)	26	Tissue Polypeptide Specific (TPS) Antigen
13	Prothrombin Fragment 1 & 2 (F1+2)	27	Tissue Factor Pathway Inhibitor (TFPI)
14	Antithrombin III Activity (ATAC)	28	Cancer Antigen-125 (CA-125)

6.1.2 Experimental Setups

The collected ovarian cancer diagnosis data set is applied to GARSINFIS and other benchmarking models for comparisons. Experiments are conducted to evaluate the capability of GARSINFIS to automatically derive human-like reasoning rules and subsequently perform accurate inferences. To resemble the diagnosis processes of doctors, a systematic ovarian cancer diagnosis system is constructed and illustrated in Figure 6-1.

When a new instance comes, it is sequentially processed until the diagnosis system reports the final result. In such a way, at each stage, the two important measurements in medical diagnosis, namely sensitivity and specificity, are reported. Sensitivity measures the degree of true positive and is defined in Equation 6-1a. Specificity measures the degree of true

Figure 6-1: Ovarian cancer diagnosis system

negative and is defined in Equation 6-1b. In ovarian cancer diagnosis, sensitivity is critical because early treatment is beneficial to patients and specificity is critical because treatment is harmful to patients mentally and physically.

$$\text{Sensitivity} = \frac{\text{Number of positive instances correctly diagnosed}}{\text{Total number of positive instances}} \quad (6-1a)$$

$$\text{Specificity} = \frac{\text{Number of negative instances correctly diagnosed}}{\text{Total number of negative instances}} \quad (6-1b)$$

The over-fitting problem of all models and their capabilities to perform accurate inferences on the whole data sets are intensively evaluated and the results are illustrated in Figure 5-6 and Figure 5-7 respectively. In this medical diagnosis application, only accuracy on the testing data set is reported with the associated sensitivities and specificities, because they are the major concerns to both doctors and patients.

In Diagnosis-1, GARSINFIS employs zero-order TSK type of fuzzy rules derived by GARSC which employs the fitness function to maximize both accuracy and interpretability. In all the other diagnosis stages, GARSINFIS employ first-order TSK type of fuzzy rules extended by applying the RLS algorithm and GARSC employs the fitness function to maximize accuracy only. The population size and the number of generations used in GARSC are given in each respective section and all the other parameters remain unchanged as given in Section 5.1 on Page 94.

6.1.3 Diagnosis-1: Normal vs. Abnormal

In diagnosis-1, 22 samples are normal and 149 are abnormal. The population size of GARSC is set to 100 and the number of generations is set to 10. Experimental results of applying GARSINFIS on ovarian cancer diagnosis-1 data set is summarized in Table 6-2.

Table 6-2: Summary of applying GARSINFIS on ovarian cancer diagnosis-1 data set

Experiment Index	Accuracy (%)	Sensitivity (%)	Specificity (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	100.0	100.0	100.0	2	2	3	62.76
2	100.0	100.0	100.0	2	2	3	84.58
3	100.0	100.0	100.0	2	2	3	71.50
4	100.0	100.0	100.0	2	2	3	66.75
5	94.74	94.00	100.0	1	2	2	75.05
6	100.0	100.0	100.0	2	2	3	74.36
7	98.25	97.96	100.0	1	2	2	68.28
8	100.0	100.0	100.0	2	2	3	75.92
9	100.0	100.0	100.0	2	2	3	72.54
10	96.49	95.92	100.0	1	2	2	80.93
mean	98.95	98.79	100.0	1.7	2	2.7	73.27
std	1.89	2.16	0.00	0.48	0.00	0.48	6.49

Table 6-2 presents the astonishing performance of GARSINFIS on ovarian cancer diagnosis-1 data set. There are only two rules employed by each GARSINFIS network and each rule consists of $2.7/2 = 1.35$ arguments in the antecedent part on average. Accuracy on the testing data set is 98.95%

and sensitivity is 98.79%. Furthermore, GARSINFIS achieves 100% specificity which means all normal patients are correctly diagnosed. Figure 6-2 illustrates the fuzzy membership functions derived in experiment-1 and Table 6-3 shows the fuzzy rules employed in the same experiment.

Figure 6-2: Derived fuzzy membership functions of ovarian cancer diagnosis-1 data set

Table 6-3: Derived fuzzy rules of ovarian cancer diagnosis-1 data set

IF TPS antigen is low and CA-125 is low	THEN the sample is normal
IF TPS antigen is high	THEN the sample is abnormal

There are three experiments, namely 5, 7 and 10, utilize only one input feature out of the total number of 28. The other seven experiments utilize two features to diagnosis all samples in the testing data set correctly. It is more encouraging to learn that all GARSINFIS networks employ no other features but the two given in Table 6-3. TPS antigen is employed in all networks and CA-125 is employed in seven of them.

The most simple diagnosis criterion adopted by doctors is that ovarian cancer usually happens in women over age 50, but it is not reliable because more deterministic tests are required to make a definite diagnosis [(NLM) 14]. GARSINFIS automatically discovers this knowledge because the input feature “age” is not utilized at all.

“A normal CA-125 level is less than 35. However, this test is not always an accurate way to tell whether a woman has ovarian cancer” [(AAFP) 14] and “the combination of TPS and CA-125 will increase the accuracy” [Sliutz 95]. These statements validate the correctness of GARSINFIS in terms of feature selection because only the two expert suggested features are utilized. Furthermore, in [Sliutz 95], the cut-off values for TPS antigen and CA-125 are given as 97U/l and 35mU/ml respectively. These diagnosis criteria are fulfilled in Figure 6-2. Therefore, the rules derived by GARSINFIS are consistent with those expert adopted ones. This significant finding best illustrates the superior property of GARSINFIS which systematically discovers intrinsic knowledge embedded in the data set without any prior knowledge or human intervention.

Experiments of applying ovarian cancer diagnosis-1 data set to other benchmarking models are conducted and results are compared in Table 6-4. All models utilize the same pairs of training and testing data sets for consistent comparisons.

In terms of accuracy, MLP, RBF, and DENFIS perform best with the difference of $99.82\% - 98.95\% = 0.87\%$, which is approximately 0.5 data sample, when comparing to GARSINFIS. However, GARSINFIS only uti-

Table 6-4: Benchmarks on ovarian cancer diagnosis-1 data set

Model		Accuracy (%)	Sensitivity (%)	Specificity (%)	Features	Rules
GARSINFIS	mean	98.95	98.79	100.0	1.7	2.0
	std	1.89	2.16	0.00	0.48	0.00
C4.5	mean	98.07	97.79	100.0	1.1	2.1
	std	1.00	1.15	0.00	0.32	0.32
Naive Bayes	mean	98.25	100.0	86.30	28	N.A.
	std	1.43	0.00	11.69		
MLP	mean	99.82	99.80	100.0	28	17
	std	0.55	0.65	0.00		
RBF	mean	99.82	100.0	98.57	28	2
	std	0.55	0.00	4.52		
ANFIS	mean	99.30	99.20	100.0	28	2.0
	std	1.23	1.42	0.00		0.00
DENFIS	mean	99.82	99.80	100.0	28	39.3
	std	0.55	0.63	0.00		2.67

lizes at most two input features while the other three models do not perform feature selection. Furthermore, MLP and RBF employ incomprehensible rules and DENFIS employs a significantly large number of rules which is $39.3/2 = 19.65$ times when comparing to GARSINFIS.

In terms of sensitivity, Naive Bayes classifier and RBF perform best with the difference of $100\% - 98.79\% = 1.21\%$, which is approximately 0.6 data sample, when comparing to GARSINFIS. However, both Naive Bayes classifier and RBF make error(s) on specificity.

In terms of specificity, all models except Naive Bayes classifier and RBF perform perfectly in all ten experiments, i.e. the other five models make correct diagnoses on all normal patients.

In terms of interpretability, C4.5 utilizes the least number of input features. However, its accuracy is lower than that of GARSINFIS. More importantly, among all ten C4.5 decision trees, CA-125 is only utilized in three experiments and TPS antigen is not utilized at all. This implies that although C4.5 utilizes lesser input features in this ovarian cancer diagnosis-

1 application, it performs worse in terms of knowledge discovery. Both GARSINFIS and ANFIS employ the least number of rules, which is the minimum number of two to distinguish two different classes. However, ANFIS employs all input features and GARSINFIS employs no more than two.

6.1.4 Diagnosis-2: Benign vs. Malignant

In diagnosis-2, 78 samples are benign and 71 are malignant. The population size of GARSC is set to 100 and the number of generations is set to 20. Experimental results of applying GARSINFIS on ovarian cancer diagnosis-2 data set is summarized in Table 6-5.

Table 6-5: Summary of applying GARSINFIS on ovarian cancer diagnosis-2 data set

Experiment Index	Accuracy (%)	Sensitivity (%)	Specificity (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	72.00	58.33	84.62	5	7	18	2186.12
2	68.00	70.83	65.38	4	6	13	2405.43
3	62.00	58.33	65.28	6	11	29	2258.81
4	74.00	70.83	76.92	6	12	33	2552.52
5	77.55	65.22	88.46	5	9	20	2185.78
6	72.00	70.83	73.08	6	10	22	2203.56
7	69.39	56.52	80.77	5	8	19	2618.34
8	68.00	66.67	69.23	4	7	16	2136.77
9	75.71	69.57	80.77	4	6	15	2287.63
10	74.00	75.00	73.08	5	9	23	2696.64
mean	71.25	66.21	75.77	5.0	8.5	20.8	2353.16
std	4.52	6.43	7.92	0.82	2.07	6.25	202.51

Experiment-3 utilizes the most number of input features and employs the second most number of rules. However, its accuracy on the testing data set is the lowest. Experiment-3 suffers more from the over-fitting problem than all the other experiments.

On average, GARSINFIS utilizes 5 features and employs 8.5 rules to achieve 71.25% accuracy on the testing data sets. Among the 8.5 rules, 2.7

of them are negative ones and the other 5.8 are positive ones. Each rule employs averagely $20.8/8.5 = 2.45$ arguments in the antecedent part.

Among all input features, D-Dimer is employed in nine experiments. This again proves the capability of GARSINFIS in terms of feature selection because D-Dimer is utilized by doctors during diagnoses. In [Mirshahi 92], it is stated that “in patients with ovarian cancer before they receive chemotherapy, the level of fibrin degradation products (D-Dimer) is correlated with the tumor load.”

Experiments of applying ovarian cancer diagnosis-2 data set to other benchmarking models are conducted and results are compared in Table 6-6. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Table 6-6: Benchmarks on ovarian cancer diagnosis-2 data set

Model		Accuracy (%)	Sensitivity (%)	Specificity (%)	Features	Rules
GARSINFIS	mean	71.26	66.21	75.77	5.0	8.5
	std	4.52	6.43	7.92	0.82	2.07
C4.5	mean	63.89	63.29	64.62	7.7	10.6
	std	3.70	7.86	9.56	1.34	1.08
Naive Bayes	mean	68.41	55.27	80.38	28	N.A.
	std	3.81	6.30	8.59		
MLP	mean	63.78	59.92	67.31	28	17
	std	4.00	7.52	6.60		
RBF	mean	68.01	59.86	75.38	28	6
	std	5.94	10.15	7.52		
ANFIS	mean	68.41	59.07	76.92	28	5.8
	std	7.30	11.69	6.79		1.32
DENFIS	mean	66.80	62.87	70.38	28	37.9
	std	8.14	10.42	8.32		1.37

Comparing Table 6-6 to 6-4, performance of all models drop significantly. However, this observation is within expectation because diagnosis-2 data set is the most complex one among all four diagnoses. In nature, benign and malignant tumors are easily separable using extra information such as ultrasound image or CT scan. However, these more distinctive fea-

tures are not included in this data set which leads to the highest accuracy of 71.26% on the testing data set.

In terms of both accuracy and sensitivity, GARSINFIS performs best. In terms of specificity, ANFIS performs best with the difference of $76.92\% - 75.77\% = 1.15\%$, which is approximately 0.3 data sample, when comparing to GARSINFIS.

In terms of interpretability, GARSINFIS utilizes lesser input features than C4.5 which employs D-Dimer and CA-125 in eight experiments respectively. Although C4.5 also perform well in terms of feature selection, its accuracy is lower than that of GARSINFIS with the difference of $71.26\% - 63.89\% = 7.37\%$ which is approximately 1.7 data samples. ANFIS employs the least number of rules. However, it does not perform feature selection and requires extra efforts to determine the number of clusters for the employed FCM clustering technique through trial-and-error.

6.1.5 Diagnosis-3: Borderline vs. beyond Borderline

In diagnosis-3, 10 samples are borderline and 61 are beyond borderline. The population size of GARSC is set to 200 and the number of generations is set to 10. Experimental results of applying GARSINFIS on ovarian cancer diagnosis-3 data set is summarized in Table 6-7.

Sensitivity on the testing data set is 90.54% on average because ovarian cancer in late stages is easier to be discovered. Once the cancer grows beyond the borderline size, it will develop rapidly in the ovary and may

Table 6-7: Summary of applying GARSINFIS on ovarian cancer diagnosis-3 data set

Experiment Index	Accuracy (%)	Sensitivity (%)	Specificity (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	82.61	89.47	50.00	2	3	4	81.96
2	83.33	95.00	25.00	3	3	4	73.87
3	78.26	80.00	66.67	2	2	3	70.71
4	83.33	95.00	25.00	3	3	5	76.68
5	82.61	100.0	0.00	2	2	3	71.37
6	79.17	76.19	100.0	2	2	3	70.70
7	91.67	100.0	50.00	3	3	6	82.16
8	82.61	95.00	0.00	2	3	4	80.46
9	73.91	80.00	33.33	3	3	5	83.30
10	78.26	94.74	0.00	2	3	4	78.93
mean	81.58	90.54	35.00	2.4	2.7	4.1	77.01
std	4.69	8.73	32.58	0.52	0.48	0.99	5.03

spread to other organs. At that time, the characteristics of ovarian cancer become increasingly distinctive.

However, specificity on the testing data set varies drastically from 0% to 100% with the average below 50% because there are not enough data instances. In each experiment, only six or seven instances of borderline class are available for training and their common characteristics are not distinctive enough. Furthermore, there are only three or four instances of borderline class in the testing data set. Both of these lead to the drastic variations of specificity on the testing data set.

Among all ten experiments, only experiment-6 has higher specificity than sensitivity. On average, GARSINFIS utilizes 2.4 features and employs 2.7 rules to achieve 81.58% accuracy on the testing data sets. Among the 2.7 rules, 1.2 of them are negative ones and the other 1.5 are positive ones. Each rule employs averagely $4.1/2.7 = 1.52$ arguments in the antecedent part.

Among all the input features, no one is utilized for more than half of the time. The most frequently selected feature is PAI activity which is utilized in five experiments. PAI antigen and CA-125 are both utilized in

four experiments respectively. These features selected by GARSINFIS once again inosculate with the ones doctor suggests. In [Casslen 94], it is stated that PAI activity and antigen in blood and tumor fluids are examined to determine the seriousness of the tumor in ovary.

Experiments of applying ovarian cancer diagnosis-3 data set to other benchmarking models are conducted and results are compared in Table 6-8. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Table 6-8: Benchmarks on ovarian cancer diagnosis-3 data set

Model		Accuracy (%)	Sensitivity (%)	Specificity (%)	Features	Rules
GARSINFIS	mean	81.58	90.54	35.00	2.4	2.7
	std	4.69	8.73	32.58	0.52	0.48
C4.5	mean	76.07	85.35	25.00	2.7	4.0
	std	6.96	7.39	19.02	0.68	0.82
Naive Bayes	mean	67.09	73.23	33.33	28	N.A.
	std	8.08	7.40	24.04		
MLP	mean	76.50	85.86	25.00	28	17
	std	6.07	8.72	22.03		
RBF	mean	75.27	86.54	14.17	28	2
	std	8.73	10.45	21.89		
ANFIS	mean	71.79	75.25	52.78	28	2.2
	std	10.05	11.62	30.44		0.42
DENFIS	mean	80.77	86.87	47.22	28	28.8
	std	5.97	7.99	31.54		1.23

According to Table 6-8, RBF achieves the lowest specificity of 14.17%. However, its sensitivity is the third highest of 86.54%. These imply that RBF does not handle the imbalance in this testing data set well.

In terms of both accuracy and sensitivity, GARSINFIS performs best. In terms of specificity, ANFIS performs best with the difference of $52.78\% - 35\% = 17.78\%$, which is approximately 0.6 data sample, when comparing to GARSINFIS.

In terms of interpretability, GARSINFIS utilizes lesser input features

than C4.5 which employs CA-125 in five experiments and does not employ PAI antigen and activity in all experiments. ANFIS employs the least number of rules. However, it does not perform feature selection and requires extra efforts to determine the number of clusters for the employed FCM clustering technique through trial-and-error.

6.1.6 Diagnosis-4: Early Stage vs. Late Stage

In diagnosis-4, 19 samples are in early stage and 42 are in late stage. The population size of GARSC is set to 200 and the number of generations is set to 20. Experimental results of applying GARSINFIS on ovarian cancer diagnosis-4 data set is summarized in Table 6-9.

Table 6-9: Summary of applying GARSINFIS on ovarian cancer diagnosis-4 data set

Experiment Index	Accuracy (%)	Sensitivity (%)	Specificity (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	80.95	80.00	83.33	3	3	4	206.57
2	75.00	84.62	57.14	2	2	3	176.61
3	85.00	92.86	66.67	3	3	5	187.80
4	71.43	71.43	71.43	2	3	4	221.09
5	90.00	85.71	100.0	3	3	4	235.06
6	85.00	84.62	85.71	3	3	4	237.07
7	85.71	86.67	83.33	2	3	4	174.93
8	85.00	85.71	83.33	2	2	3	223.05
9	90.00	100.0	66.67	3	3	5	232.96
10	85.00	85.71	83.33	2	2	3	241.67
mean	83.31	85.73	78.09	2.5	2.7	3.9	213.68
std	5.99	7.42	12.44	0.53	0.48	0.74	25.60

Comparing Table 6-9 to Table 6-7, performance of GARSINFIS increases in every aspect. This is because ovarian cancer in late stage is easier to be identified in nature. On average, GARSINFIS utilizes 2.5 features and employs 2.7 rules to achieve 83.31% accuracy on the testing data sets. Among the 2.7 rules, 1.5 of them are negative ones and the other 1.2 are positive ones. Each rule employs averagely $3.9/2.7 = 1.44$ arguments in the antecedent part.

Among all input features, D-Dimer is utilized in all ten experiments and CA-125 is utilized in eight. These two features are the ones suggested by doctors [Mirshahi 92].

Experiments of applying ovarian cancer diagnosis-4 data set to other benchmarking models are conducted and results are compared in Table 6-10. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Table 6-10: Benchmarks on ovarian cancer diagnosis-4 data set

Model		Accuracy (%)	Sensitivity (%)	Specificity (%)	Features	Rules
GARSINFIS	mean	83.31	85.73	78.09	2.5	2.7
	std	5.99	7.42	12.44	0.53	0.48
C4.5	mean	69.95	77.14	53.97	3.6	4.6
	std	10.66	11.70	17.53	0.84	0.84
Naive Bayes	mean	77.83	80.00	73.02	28	N.A.
	std	6.17	8.85	16.30		
MLP	mean	78.33	82.86	68.25	28	17
	std	6.21	9.53	16.54		
RBF	mean	75.33	82.31	61.19	28	2
	std	5.89	7.49	22.16		
ANFIS	mean	75.86	80.71	65.08	28	2.9
	std	10.34	9.19	24.49		0.99
DENFIS	mean	78.33	68.25	82.86	28	26.7
	std	6.00	20.30	5.82		1.06

In terms of both accuracy and sensitivity, GARSINFIS performs best. In terms of specificity, DENFIS performs best with the difference of $82.86\% - 78.09\% = 4.77\%$, which is approximately 0.3 data sample, when comparing to GARSINFIS.

In terms of interpretability, GARSINFIS utilizes lesser input features than C4.5 which employs D-Dimer and CA-125 in five experiments respectively. Furthermore, GARSINFIS employs the least number of rules.

6.1.7 Summary on Ovarian Cancer Diagnosis Application

The features selected by doctors, GARSINFIS, and C4.5 are summarized in Table 6-11. It is obvious that GARSINFIS performs better than C4.5 in terms of feature selection and more importantly, the features selected by GARSINFIS is more consistent with the ones suggested by doctors. The number shown in the parentheses denotes the number of occurrences among ten experiments.

Table 6-11: Summary of selected features on ovarian cancer diagnosis data set

Diagnosis	1	2	3	4
Doctor suggests	CA-125 TPS antigen	D-Dimer	PAI activity PAI antigen	D-Dimer CA-125
GARSINFIS selects	TPS antigen (10) CA-125 (7)	D-Dimer (9)	PAI activity (5) PAI antigen (4) CA-125 (4)	D-Dimer (10) CA-125 (8)
C4.5 decision tree selects	ATAC (7) CA-125 (3)	D-Dimer (8) CA-125 (8)	CA-125 (5)	D-Dimer (5) CA-125 (5)

To systematically perform diagnoses on new input data instances, all the four diagnoses are integrated to form one expert system which is illustrated in Figure 6-1. Table 6-12 summaries the accumulated accuracy of all models and Figure 6-3 visualizes Table 6-12 with respect to GARSINFIS.

Table 6-12: Accumulated accuracy of different ovarian cancer diagnosis systems

Model	Normal	Benign	Borderline	Stage I & II	Stage III & IV	Combined Staging %	Overall Accuracy	Overall Diagnoses
GARSINFIS	100.0%	74.85%	22.89%	50.77%	46.24%	97.01%	65.35%	111.74
C4.5	100.0%	63.19%	15.47%	28.51%	40.75%	69.26%	55.77%	95.37
Naive Bayes	86.30%	80.38%	18.42%	29.55%	32.38%	61.93%	60.08%	102.74
MLP	100.0%	67.18%	14.95%	35.04%	42.54%	77.58%	58.73%	100.42
RBF	98.57%	75.38%	8.48%	42.64%	31.70%	74.34%	60.09%	102.75
ANFIS	100.0%	76.30%	30.93%	28.70%	35.59%	64.29%	61.41%	105.01
DENFIS	100.0%	70.24%	29.63%	45.16%	37.20%	82.36%	60.80%	103.95
No. of samples	22	78	10	19	42	N/A	N/A	171

It is visualized in Figure 6-3 that GARSINFIS performs best in three types of diagnoses and in the middle level for the other two types. More-

Figure 6-3: Visualization of accuracy on ovarian cancer diagnosis application

over, it is stated in the last column of Table 6-12 that GARSINFIS achieves the most number of correct diagnoses if the testing data set has the same distribution as the whole data set. In each diagnosis, GARSINFIS utilizes a small number of features which varies from 1.7 to 5 and employs a small number of rules which varies from 2 to 8.5. Therefore, GARSINFIS utilizes only the most essential knowledge to achieve a high level of accuracy in general. More importantly, in GARSINFIS, the systematically derived rules are consistent with those suggested by doctors. This proves that GARSINFIS can automatically discover the intrinsic knowledge embedded in the ovarian cancer diagnosis data set without any prior information or human intervention.

Although the ovarian cancer diagnosis data set has been studied in

the literature [Tan 08], the data set studied in this thesis is updated by the doctors (two unimportant features are removed and all missing values are filled). Moreover, how the experiments are conducted in [Tan 08] is quite different with the natural diagnosis system illustrated in Figure 6-1 and the applied model (possesses inferior interpretability) requires the employment of a separate method to perform feature selections as the pre-processing step. Therefore, the model applied in [Tan 08] is not compared to GARSINFIS in this thesis. The work that shows how GARSINFIS performs in this real-world ovarian cancer diagnosis application is presented in [Wang 14b].

6.2 Bank Failure Prediction System

6.2.1 Overview of Bank Failure Prediction Application

Bank failure prediction is an important study for regulators in the banking industry because the failure of a bank leads to devastating consequences. Generally speaking, bank failures are due to financial distress and are predictable by assessments on certain financial covariates. If bank failures are correctly predicted, early warnings can be sent to the responsible authorities for precaution purposes. Therefore, reliable bank failure prediction system is invaluable to avoid adverse repercussion effects on other banks and to prevent drastic confidence losses in the society.

The bank failure prediction data set used in this thesis is extracted from the financial statements of 3635 banks in US which are downloadable at [(FRBC) 14]. The data set consists of nine financial covariates to categorize whether a bank is failed or survived. The selected features are advised by financial experts [Lane 86, Cole 95] and are listed in Table 6-13. Most empirical studies in banking finance run regressions on these covariates. However, in [Tung 04], it is stated that soft computing models overcome the deficiencies of traditional statistical models whose results do not possess semantic meanings.

Table 6-13: Input features of bank failure prediction data set

Category	Financial covariate	Expected impact on bank failure
Capital Adequacy	CAPADE: $\frac{\text{average total equity capital}}{\text{average total assets}}$	if ratio is high, then capital is relatively large to absorb losses, i.e. small probability to fail
Asset (Loan) Quality	OLAQLY: $\frac{\text{average (accumulated) loan loss allowance}}{\text{average total loans \& leases}}$	if ratio is small, then loan quality is relatively good, i.e. small probability to fail
	PROBLO: $\frac{\text{average (accumulated) loans 90+ days late}}{\text{average total loans \& leases}}$	
	PLAQLY: $\frac{\text{(annual) loan loss provision}}{\text{average total loans \& leases}}$	
Management	NIEOIN: $\frac{\text{non interest expense}}{\text{operating income}}$	if ratio is small, then bank operates relatively healthy, i.e. small probability to fail
Earning	NINMAR: $\frac{\text{total interest income} - \text{interest expense}}{\text{average total asset}}$	if ratio is high, then bank is relatively profitable, i.e. small probability to fail
	ROE: $\frac{\text{net income (after tax)} + \text{applicable income tax}}{\text{average total equity capital}}$	
Liquidity	LIQUID: $\frac{\text{(all taken the average values) cash} + \text{federal funds sold}}{\text{total deposit} + \text{fed funds bought} + \text{liability}}$	if ratio is small, then utilization of resources is relatively efficient, i.e. small probability to fail
Miscellaneous	GROWLA: $\frac{(\text{total loans \& leases})_t - (\text{total loans \& leases})_{t-1}}{(\text{total loans \& leases})_{t-1}}$	if ratio is high, then loan growth is relatively high, i.e. small probability to fail

6.2.2 Experimental Setups

The processed bank failure prediction data set is applied to GARSINFIS and other benchmarking models for comparisons. Experiments are conducted to evaluate the capability of GARSINFIS to derive a small number of rules and subsequently perform accurate inferences on a large number of unseen instances. Sensitivity and specificity which are defined in Equation

6-1 are reported besides accuracy on the testing data set. In bank failure prediction, sensitivity is critical because early warnings to poor performing banks buy more time to prevent unnecessary consequences and specificity is critical because false alarms to good performing banks create unnecessary panics.

There are three scenarios studied in this bank failure prediction application. Numbers of all available data samples in each scenario are listed in Table 6-14. In scenario-1, each model actually identifies whether a bank is failed. While in scenario-2 and scenario-3, each model predicts whether a bank will fail in one and two years respectively.

Table 6-14: Distributions of data samples in bank failure prediction data set

Index	Scenario	Total number	Survived (negative)	Failed (positive)
1	Last year available	3103	2555 (82.34%)	548 (17.66%)
2	One year prior	3046	2572 (84.44%)	474 (15.56%)
3	Two years prior	2943	2585 (87.84%)	358 (12.16%)

To prove that GARSINFIS utilizes only the most essential knowledge to perform accurate inferences, in each scenario and in each experiment, 100 instances from each class are randomly selected for training and all the other instances consist of the testing data set. In such a way, the ability to derive a correct inference knowledge base from a relatively small and balanced training data set is ultimately assessed.

In all three scenarios, GARSINFIS employs zero-order TSK type of fuzzy rules to maximize the level of interpretability because the requirement of accuracy on bank failure prediction application is relatively high and financial experts only approve simple and comprehensible rules. In all

three scenarios, GARSC utilizes the fitness function defined in Equation 3-3 on Page 65 with $\tau_1 = \tau_3 = \tau_5 = 1, \tau_2 = \tau_4 = 0$. The reason for only number of rules is taken into consideration during the optimization process is given as follows. All features in this application are financial covariates selected by experts. Therefore, everyone of them is guaranteed to be more or less related. Furthermore, all features are utilized when running regressions. Therefore, the number of arguments in each derived fuzzy rule is not a big concern to financial experts. In all three scenarios, the population size is set to 300 and the number of generations is set to 20. All the other parameters remain unchanged as given in Section 5.1 on Page 94.

6.2.3 Scenario-1: Last Year Available

Experimental results of applying GARSINFIS on bank failure prediction scenario-1 data set is summarized in Table 6-15.

Table 6-15: Summary of applying GARSINFIS on bank failure prediction scenario-1 data set

Experiment Index	Accuracy (%)	Sensitivity (%)	Specificity (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	96.76	96.65	96.78	4	4	9	618.80
2	96.42	95.09	96.66	4	5	9	700.74
3	95.73	94.87	95.89	4	6	13	664.22
4	96.45	96.21	96.50	5	7	15	709.54
5	97.21	95.76	97.47	3	4	8	663.12
6	96.80	92.63	97.56	4	4	7	583.02
7	96.66	91.96	97.52	4	4	7	545.96
8	96.18	95.31	96.33	4	4	8	663.97
9	96.56	93.30	97.15	4	5	11	705.57
10	96.14	95.54	96.25	4	6	13	598.07
mean	96.49	94.73	96.81	4	4.9	10.0	645.30
std	0.41	1.57	0.59	0.47	1.10	2.83	56.31

According to Table 6-15, performance of GARSINFIS on bank failure prediction scenario-1 data set is consistent. Although experiment-4 utilizes the most number of features and employs the most number of rules, it does

not achieve the highest accuracy. On the contrary, experiment-5 utilizes the least number of features and the least number of rules to achieve the highest accuracy. The derived fuzzy membership functions in experiment-5 is illustrated in Figure 6-4 and the employed fuzzy rules in the same experiment are shown in Table 6-16.

Figure 6-4: Derived fuzzy membership functions of bank failure prediction scenario-1 data set

Table 6-16: Derived fuzzy rules of bank failure prediction scenario-1 data set

IF PLAQLY is small and GROWLA is large	THEN the bank is survived
IF CAPADE is large and PLAQLY is medium	THEN the bank is survived
IF CAPADE is small and GROWLA is small	THEN the bank is failed
IF CAPADE is medium and PLAQLY is large	THEN the bank is failed

Experiment-5 utilizes three input features and two rules for each class to achieve the accuracy of 97.21% on the testing data set. It is more encouraging to learn that the rules shown in Table 6-16 are consistent with

expert knowledge presented in Table 6-13 that if CAPADE and GROWLA are large and PLAQLY is small, then the probability of bank failure is small.

On average, GARSINFIS utilizes 4 input features and employs 4.9 rules to achieve 96.49% accuracy on the testing data sets. Among the 4.9 rules, 3 of them are negative ones and the other 1.9 are positive ones. Each rule employs averagely $10/4.9 = 2.04$ arguments in the antecedent part.

Experiments of applying bank failure prediction scenario-1 data set to other benchmarking models are conducted and results are compared in Table 6-17. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Table 6-17: Benchmarks on bank failure prediction scenario-1 data set

Model		Accuracy (%)	Sensitivity (%)	Specificity (%)	Features	Rules
GARSINFIS	mean	96.49	94.73	96.81	4	4.9
	std	0.41	1.57	0.59	0.47	1.10
C4.5	mean	95.76	95.42	95.82	2.8	5.2
	std	0.68	1.85	0.95	0.63	1.40
Naive Bayes	mean	96.26	96.25	96.26	9	N.A.
	std	0.42	0.87	0.63		
MLP	mean	97.31	95.31	97.67	9	7
	std	0.60	1.84	0.68		
RBF	mean	96.06	95.27	96.20	9	2
	std	0.47	1.07	0.71		
ANFIS	mean	97.37	91.25	98.48	9	4.0
	std	0.55	3.56	0.38		1.33
DENFIS	mean	96.13	95.36	96.27	9	28.2
	std	1.16	1.12	1.33		5.49

In terms of accuracy, ANFIS performs best with the difference of $97.37\% - 96.49\% = 0.88\%$, which is approximately 25.5 data samples, when comparing to GARSINFIS. However, ANFIS achieves the lowest sensitivity. In bank failure prediction application, sensitivity is more important than specificity because the consequence of no early warnings is more serious than that of false alarms. Therefore, although ANFIS achieves the highest

accuracy on the testing data set, it is not a reliable model in this particular application.

In terms of sensitivity, Naive Bayes classifier performs best with the difference of $96.25\% - 94.73\% = 1.52\%$, which is approximately 6.8 data samples, when comparing to GARSINFIS.

In terms of specificity, ANFIS performs best with the difference of $98.48\% - 96.81\% = 1.67\%$, which is approximately 41 data samples, when comparing to GARSINFIS. However, as discussed earlier, ANFIS achieves the lowest sensitivity.

In terms of interpretability, C4.5 utilizes the least number of input features. However, its accuracy is lower than that of GARSINFIS and it employs more rules than GARSINFIS does. ANFIS employs the least number of rules. However, it utilizes all the input features and requires extra efforts to determine the number of clusters for the employed FCM clustering technique through trial-and-error.

In summary, GARSINFIS only employs 4.9 rules on average to achieve satisfactory accuracy on the testing data set. Although GARINFIS selects more features than C4.5 does, each rule in GARSINFIS only employs 2.04 arguments on average. Therefore, it is shown that GARSINFIS employs a compact and comprehensive inference knowledge base to achieve satisfactory sensitivity and specificity.

6.2.4 Scenario-2: One Year Prior

Experimental results of applying GARSINFIS on bank failure prediction scenario-2 data set is summarized in Table 6-18.

Table 6-18: Summary of applying GARSINFIS on bank failure prediction scenario-2 data set

Experiment Index	Accuracy (%)	Sensitivity (%)	Specificity (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	92.90	91.98	93.04	5	6	15	943.37
2	92.59	90.37	92.92	4	9	20	1145.53
3	92.27	93.32	92.11	6	11	26	1292.22
4	91.53	90.11	91.75	5	6	13	1038.10
5	94.59	92.25	94.94	6	8	18	1166.04
6	92.13	91.18	92.27	5	9	21	1308.21
7	92.45	93.85	92.23	5	5	12	979.59
8	91.74	89.84	92.03	4	7	14	1010.98
9	91.53	92.51	91.38	5	8	20	1195.62
10	92.13	89.04	92.60	6	10	25	1290.32
mean	92.39	91.44	92.53	5.1	7.9	18.4	1137.00
std	0.89	1.59	0.99	0.74	1.91	4.88	137.11

Comparing Table 6-18 to Table 6-15, performance of GARSINFIS on bank failure prediction scenario-2 data set decreases because accuracy decreases with more features utilized, more rules employed, and more computational time spent. However, this is within expectation because prediction of bank failures is naturally more difficult based on one year prior data.

On average, GARSINFIS utilizes 5.1 input features and employs 7.9 rules to achieve 92.39% accuracy on the testing data sets. Among the 7.9 rules, 5 of them are negative ones and the other 2.9 are positive ones. Each rule employs averagely $18.4/7.9 = 2.33$ arguments in the antecedent part.

Experiments of applying bank failure prediction scenario-2 data set to other benchmarking models are conducted and results are compared in Table 6-19. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Performance comparisons of all models reported in Table 6-19 are simi-

Table 6-19: Benchmarks on bank failure prediction scenario-2 data set

Model		Accuracy (%)	Sensitivity (%)	Specificity (%)	Features	Rules
GARSINFIS	mean	92.39	91.44	92.53	5.1	7.9
	std	0.89	1.59	0.99	0.74	1.91
C4.5	mean	90.59	91.04	90.52	4.7	7.9
	std	1.70	1.75	2.10	1.34	1.97
Naive Bayes	mean	92.70	85.83	93.74	9	N.A.
	std	1.10	3.29	1.71		
MLP	mean	93.15	91.42	93.41	9	7
	std	1.41	2.44	1.83		
RBF	mean	92.05	90.64	92.26	9	6
	std	1.24	1.57	1.55		
ANFIS	mean	94.36	88.72	95.21	9	6.3
	std	1.80	2.54	1.98		1.49
DENFIS	mean	92.14	89.68	92.51	9	29.8
	std	1.25	2.07	1.52		5.83

lar to that of Table 6-17. Except that GARSINFIS performs best in terms of sensitivity on the testing data set. In terms of both accuracy and specificity, ANFIS performs best with the differences of $94.36\% - 92.39\% = 1.97\%$ and $95.21\% - 92.53\% = 2.68\%$, which are approximately 56 and 66 data samples respectively, when comparing to GARSINFIS. However, as discussed earlier, because sensitivity is more important than specificity in this particular application, ANFIS is not a more reliable model than GARSINFIS.

In summary, GARSINFIS employs slightly more features and the same number of rules when comparing against C4.5. By employing a compact inference knowledge base, GARSINFIS achieves the highest sensitivity and satisfactory specificity.

6.2.5 Scenario-3: Two Year Prior

Experimental results of applying GARSINFIS on bank failure prediction scenario-3 data set is summarized in Table 6-20.

Table 6-20: Summary of applying GARSINFIS on bank failure prediction scenario-3 data set

Experiment Index	Accuracy (%)	Sensitivity (%)	Specificity (%)	Number of features	Number of rules	Number of arguments	Time (seconds)
1	91.36	91.09	91.39	7	13	33	2106.18
2	90.30	87.60	90.58	7	10	25	1819.11
3	89.17	88.76	89.22	6	8	22	2006.57
4	87.64	91.86	87.20	5	8	20	1750.96
5	91.62	88.37	91.95	7	15	39	1942.09
6	89.46	87.21	89.70	7	9	26	2118.63
7	91.32	89.15	91.55	6	13	37	2044.56
8	88.84	87.98	88.93	5	10	27	2175.83
9	91.18	86.43	91.67	5	11	29	1934.32
10	88.23	89.53	88.09	6	12	31	2033.17
mean	89.91	88.80	90.03	6.1	10.9	28.9	1993.14
std	1.44	1.69	1.65	0.88	2.33	6.17	133.89

Comparing Table 6-20 to 6-18, performance of GARSINFIS on bank failure prediction scenario-3 data set further decreases because accuracy decreases with more features utilized, more rules employed, and more computation time spent. However, this is within expectation because prediction of bank failures is naturally even more difficult based on two year prior data.

On average, GARSINFIS utilizes 6.1 features and employs 10.9 rules to achieve 89.91% accuracy on the testing data sets. Among the 10.9 rules, 6.8 of them are negative ones and the other 4.1 are positive ones. Each rule employs averagely $28.9/10.9 = 2.65$ arguments in the antecedent part.

Experiments of applying bank failure prediction scenario-3 data set to other benchmarking models are conducted and results are compared in Table 6-21. All models utilize the same pairs of training and testing data sets for consistent comparisons.

Performance comparisons of all models reported in Table 6-21 are similar to that of Table 6-17. Except that GARSINFIS performs best in terms of sensitivity on the testing data set. In terms of both accuracy and specificity, MLP performs best with the differences of $92.55\% - 89.91\% = 2.64\%$

Table 6-21: Benchmarks on bank failure prediction scenario-3 data set

Model		Accuracy (%)	Sensitivity (%)	Specificity (%)	Features	Rules
GARSINFIS	mean	89.91	88.80	90.03	6.1	10.9
	std	1.44	1.69	1.65	0.88	2.33
C4.5	mean	85.46	87.25	85.28	5.5	9.2
	std	2.69	3.26	3.21	1.35	2.20
Naive Bayes	mean	89.86	75.47	91.35	9	N.A.
	std	2.43	5.99	3.21		
MLP	mean	92.55	87.21	93.10	9	7
	std	1.48	2.65	1.75		
RBF	mean	88.67	86.74	88.87	9	6
	std	2.02	3.75	2.51		
ANFIS	mean	91.91	82.17	92.93	9	7.4
	std	1.91	3.42	2.28		2.76
DENFIS	mean	91.26	87.75	91.62	9	26.6
	std	1.46	1.61	1.64		2.91

and $93.10\% - 90.03\% = 3.07\%$, which are approximately 72.4 and 76.3 data samples respectively, when comparing to GARSINFIS. However, knowledge derived by MLP does not possess semantic meanings.

In summary, although GARSINFIS employs more features and rules than C4.5 does, GARSINFIS outperforms C4.5 in terms of accuracy. By employing a compact inference knowledge base, GARSINFIS achieves the highest sensitivity and satisfactory specificity.

6.2.6 Summary on Bank Failure Prediction Application

In this bank failure prediction application, although only number of rules is defined in the fitness function besides accuracy terms, GARSINFIS still utilize a relatively small number of features and a relatively small number of arguments in the antecedent part of each rule. High level of interpretability is achieved.

Performance of all models decrease with an increase of the prediction time of bank failures. GARSINFIS utilizes more features and employs more rules to deal with the increasing complexity. It is encouraging to learn that GARSINFIS becomes more reliable with the increase of bank failure prediction time because more positive rules are employed to achieve the highest sensitivity among all models.

Table 6-22 summarizes the input features selected by GARSINFIS and C4.5. If a feature is utilized for more than half of the time, then its value is highlighted to denote the importance. Although in all the three scenarios, GARSINFIS utilizes more input features than C4.5 does, the features selected by GARSINFIS are more concentrated. Among all the nine financial covariates, CAPADE and PLAQLY are utilized in all GARSINFIS experiments and most C4.5 experiments.

Table 6-22: Summary of selected features on bank failure prediction data set

Feature	GARSINFIS			C4.5 Decision Tree		
	Scenario-1	Scenario-2	Scenario-3	Scenario-1	Scenario-2	Scenario-3
CAPADE	10	10	10	10	10	10
OLAQLY	4	2	3	0	3	1
PROBLO	3	4	7	1	5	6
PLAQLY	10	10	10	8	10	10
NIEOIN	4	7	7	2	3	4
NINMAR	0	1	3	0	4	3
ROE	1	6	9	4	7	10
LIQUID	2	9	10	2	3	8
GROWLA	6	2	2	1	2	3

The work that shows how GARSINFIS performs in this real-world bank failure prediction application is presented in [Wang 14a].

6.3 Summary

Different configurations of GARSINFIS are applied to different applications according to the respective complexities. In ovarian cancer diagnosis-1 application, GARSINFIS employs zero-order TSK type of fuzzy rules derived by GARSC which employs the fitness function to maximize both accuracy and interpretability. In the rest diagnoses of ovarian cancer application, GARSINFIS employs first-order rules and the fitness function maximizes accuracy only. In bank failure prediction application, GARSINFIS employs zero-order rules to preserve the interpretability and the fitness function consists of two accuracy terms and one term to minimize the number of rules. GARSINFIS is empirically shown as performing better than the other models in terms of both accuracy and interpretability. It is more encouraging to learn that GARSINFIS discovers intrinsic and expert-approved knowledge automatically from the data set without prior understanding of the application domain and human intervention.

Chapter 7

Conclusion and Future Work

“Stay hungry, stay foolish.”

- Steve Jobs

This chapter concludes on the success of the proposed research work and recommends possible future improvements.

7.1 Conclusion

This thesis presents the details of a novel hybrid intelligent system named Genetic Algorithm and Rough Set Incorporated Neural Fuzzy Inference System or GARSINFIS in short. The intelligent system employs Genetic Algorithm based Rough Set Clustering technique or GARSC in short to self-organize its network structure. The fuzzy inference knowledge base is

systematically derived by GARSC. Furthermore, the size of the knowledge base is relatively small because redundant and inconsistent knowledge have been removed by applying rough set approximations. The proposed hybrid model consists of various components in the field of soft computing, computational intelligence. The three major research areas covered in this thesis are illustrated in Figure 7-1 and further elaborated with cross-references in Table 7-1.

Figure 7-1: Overview of research areas studied in this thesis

Table 7-1: Summaries on major accomplishments presented in this thesis

Area	Model employed	Major Research Issues Addressed	Cross-reference
(a)	GARSINFIS	self-organizing of network structure	Section 4.1
	Genetic Algorithm	optimization of membership functions (MFs)	Section 4.3
	Recursive Least Squares	tuning consequent part of TSK fuzzy rules	Section 4.6
(b)	GARSC	automated MF and rule formulations	Section 3.2.1
(c)	constraints in GA	small number of interpretable MFs	Section 3.2.2
	Rough Set theory	feature selection and attribute reduction	Section 3.2.3
		removal of inconsistent rules	Section 3.2.4
(d)	multi-objective fitness function	balance between accuracy and interpretability	Section 3.2.6

In area (a) of Figure 7-1 and Table 7-1, GARSINFIS is proposed to address the questions of how to self-organize a compact network structure and how to tune the parameters. Generally speaking, neural fuzzy inference system combines the learning capabilities of neural networks and the transparent properties of fuzzy systems together by performing respective

fuzzy or non-fuzzy operations in each layer of the network. GARSINFIS self-organizes its network structure based on the fuzzy inference knowledge base derived by the proposed clustering technique. Because knowledge reduction is performed, the systematically derived knowledge base utilizes only the most essential information extracted from the given data set. As a result, GARINFIS has a relatively small network structure because it is partially connected according to the simplified fuzzy rules and it utilizes only the selected features. During the clustering process, numerous membership functions are evaluated. Therefore, when genetic algorithm terminates, the sets of best performing membership functions, which have been evaluated and preserved throughout the iterative clustering process, are provided. Fuzzy rules derived by the proposed clustering technique are zero-order TSK type. To improve the accuracy by sacrificing interpretability, the Recursive Least Squares learning algorithm is applied to formulate first-order fuzzy rules of TSK type. However, although the consequent part of the fuzzy rules become difficult for interpretations, the inference rule set still possesses other properties which are characterized as a high level of interpretability.

In area (b) of Figure 7-1 and Table 7-1, Genetic Algorithm based Rough Set Clustering technique or GARSC in short is proposed to tackle the issue on how to automatically formulate fuzzy membership functions and rules. GARSC is a supervised clustering technique which partitions the data elements in each dimension to form clusters. GARSC discretizes the given data set to construct simplified decision tables by applying rough set approximations. The crisp decision rules are easily transformed into fuzzy

rules by replacing the categorical values with fuzzy membership functions and assigning the decision attributes to the associated consequent part of zero-order TSK type of fuzzy rules. The translation from crisp to fuzzy is necessary because fuzzy rules allow tolerance between clusters and can deal with inexact information and unforeseen circumstances (see Section 5.2.1). The simplified fuzzy rules are presented to GARSINFIS for performance evaluation based on the pre-defined fitness function. Better performing chromosomes have greater chance to survive and to produce offspring. Therefore, chromosomes in the next generations are expected to perform better. Modified uniform crossover operator, which handles chromosomes with different lengths, is proposed in GARSC. Other strategies of GARSC are also carefully defined to ensure that GARSC performs more exploration in the early generations and more exploitation in the later generations. There are a number of parameters and constraints required to run GARSC. However, these parameters are only guidelines to follow and are not specific instructions to execute. The formulation of fuzzy membership functions and rules is still considered to be systematic. To better understand how GARSC works, Figure 7-2 is presented to better illustrate the iterative optimization process.

In area (c) of Figure 7-1 and Table 7-1, the objective to produce high level of interpretable inference knowledge base is achieved. A maximum number of separation boundaries allowable in each dimension is defined in GARSC. This constraint subsequently determines the maximum number of fuzzy membership functions formulated in each dimension. Lesser fuzzy membership functions make the corresponding linguistic labels more com-

Figure 7-2: Illustration of the iterative process of GARSC technique

prehensible. Furthermore, another constraint in GARSC is the minimum distance required between adjacent separation boundaries. This constraint intuitively prevents fuzzy membership functions from overlapping too much with adjacent ones, which again makes the corresponding linguistic labels more comprehensible. Rough set approximations are applied to remove all redundant and inconsistent information from the inference knowledge base. The removal of every dispensable condition attribute stated in each decision rule is termed attribute reduction. If for all rules in the decision table, an attribute is always dispensable, then the attribute is removed from the given data set which is often referred as feature selection. The primary criterion to remove inconsistent rules is based on the confidence values which are computed using rough set relations. According to the magic number of 7 ± 2 [Miller 56] and the Occam's Razor theory, the inference knowledge base should always be simplified if no essential information is lost. GARSC automatically derives a compact inference knowledge base which possesses a high level of interpretability.

In area (d) of Figure 7-1 and Table 7-1, how to leverage the balance

between accuracy and interpretability, which is the major focus of this thesis, is defined. Levels of both accuracy and interpretability are evaluated in the pre-defined multi-objective fitness function. Although the terms in the fitness function are ordered by the amount of effects according to the slightest change respectively. User is always welcome to redefine the order of importance by adjusting the respective coefficients. This multi-objective fitness function can be transformed into a pure accuracy-driven one by simply assigning coefficients of all the interpretability terms to zero. Although only accuracy is pursued after the transformation, the derived knowledge base is still compact because rough set approximations are still applied to remove all redundant and inconsistent information.

The overall system is conceptually simple to understand but carefully defined in mathematics. The parameters and constraints to derive an accurate and interpretable knowledge base are carefully defined. Extensive experiments are conducted to test the performance of GARSINFIS. The necessity of translation from crisp rules to fuzzy ones is demonstrated in Section 5.2. According to Section 5.3, the transformation to accuracy-driven fitness function improves accuracy without sacrificing much of interpretability. Section 5.4 shows that employing first-order TSK type of fuzzy rules degrades interpretability but is compensated with improved accuracy. Another approach to improve accuracy is to form a committee of models to reduce the individual modeling bias which is introduced and examined in Section 5.5. GARSINFIS is also applied to two real-world applications, namely ovarian cancer diagnosis in Section 6.1 and bank failure prediction in Section 6.2. Extensive experimental results on various data sets

show that GARSINFIS produces satisfactory accuracy with relatively less amount of inference knowledge compared to the other models. Although GARSINFIS does not achieve the highest testing accuracy among all the benchmarking models in every experiment, in most of the experiments, it achieves either the highest testing accuracy or its testing accuracy is not different from the respective winners (by applying ANOVA) among all the NFISs compared in this thesis. It is more encouraging to learn that GARSINFIS discovers intrinsic and expert-approved knowledge automatically from the data sets.

7.2 Future Work

Although experimental results prove the success of GARSINFIS, further improvements are still required. First of all, the fuzzy membership functions employed by GARSINFIS are formulated by the genetic algorithm optimization process. Once the best set of membership functions is determined, there will be no further tuning of them. Tuning fuzzy membership functions in an adaptive way is one possible extension to the existing model. Furthermore, Gaussian type of membership function is formulated based on the assumption that all data elements are normally distributed in the cluster. However, this assumption does not always stand. A possible way to ease this assumption is to employ irregular-shaped membership functions. Asymmetric Gaussian type of membership function [Lin 05], which is the most fundamental multivariate Gaussian type, allows two different widths defined on the opposite sides of the center. The introduction of

asymmetric geometry is expected to better describe the natural distribution of the data elements. An alternative way is to employ type-2 fuzzy membership functions which essentially introduce another level of uncertainty to the conventional degree of belongingness [Juang 08]. Both methods support online learning. Therefore, when the consequent part of TSK type of fuzzy rules is adaptively adjusted by employing the Recursive Least Squares algorithm, the antecedent part of the rules can also be adaptively tuned without jeopardizing the semantic meanings of the corresponding linguistic labels.

Current version of GARSINFIS requires the decision attribute of the given data set to be discrete. If the decision values are continuous, a proper way to automatically discretize the output dimension has to be defined. Rather than employing other discretization algorithms, GARSC could be extended to select separation boundaries on the decision attribute as well. However, this task is not as easy as it appears because the number of optimal partitions on the decision attribute is difficult to be self-determined. Unlike the maximum number of partitions constrained on each condition attribute where rough set approximations can be applied to reduce the actual number, partitions on the decision attribute cannot be removed by applying rough set approximations. Then the optimal number has to be estimated heuristically by the genetic algorithm optimization process. To search for optimal separation boundaries for both condition and decision attributes, the complexity of the algorithm increases exponentially. In such case, genetic algorithm requires more guidelines or more specific instructions to competently meet the challenge.

Although GARSC systematically derives the inference knowledge base, it starts from scratch. The capability of the pre-insertion of expert knowledge will be a great improvement. Moreover, it is always desirable to merge knowledge bases from different sources together (removing possible conflicts). To solve the disputes, rough set theory could be applied to assess the integrity or credibility of the knowledge bases and subsequently decides which one is more reliable than the others.

Last but not least, although all optimizations of GAs in various application domains are completed within hours, results of some experiments can be improved if more time is given. On the other hand, if certain constraints can be defined or higher level control can be realized to minimize the number of revisits, the required computation time of GA can be greatly reduced. Moreover, GA naturally exhibits parallel-distributed computations which can be mounted to a distributed system. To construct such a system, a commander unit is required to process all the reported evaluations and to distribute the disjoint candidates in the new generation to all the other units respectively. The incorporation of more processing units increases the total number of evaluated candidates and decreases the required computation time.

List of Publications

Journal papers:

- (1) **D. Wang**, C. Quek & G. S. Ng. *Ovarian cancer diagnosis using a hybrid intelligent system with simple yet convincing rules*. Applied Soft Computing, vol. 20, pages 25-39, 2014.
- (2) **D. Wang**, C. Quek & G. S. Ng. *Bank failure prediction using an accurate and interpretable neural fuzzy inference system*. AI Communications, Special Track on Soft Computing in Finance and Economics, 2014. Accepted.
- (3) **D. Wang**, C. Quek & G. S. Ng. *Novel self-organizing Takagi-Sugeno-Kang fuzzy neural networks based on ART-like clustering*. Neural Processing Letters, vol. 20, no. 1, pages 39-51, 2004.
- (4) **D. Wang**, C. Quek & G. S. Ng. *GARSINFIS: A novel hybrid intelligent system focusing on both accuracy and interpretability*. Submitted to IEEE Transactions on Fuzzy Systems.

Conference papers:

- (1) **D. Wang**, G. S. Ng & C. Quek. *A novel hybrid intelligent system: Genetic algorithm and rough set incorporated neural fuzzy inference system.* In Proceedings of IEEE Congress on Evolutionary Computation, pages 2749-2755, 2008.
- (2) **D. Wang**, G. S. Ng & C. Quek. *Ovarian cancer diagnosis using fuzzy neural networks empowered by evolutionary clustering technique.* In Proceedings of IEEE Congress on Evolutionary Computation, pages 9314-9320, 2006.
- (3) **D. Wang**, C. Quek & G. S. Ng. *MS-TSKfnn: Novel Takagi-Sugeno-Kang fuzzy neural network using ART like clustering.* In Proceedings of IEEE International Joint Conference on Neural Networks, pages 2361-2366, 2004.

Publications in other research fields which are not presented in this thesis:

- (1) **D. Wang** & A. H. Tan. *Creating autonomous adaptive agents in a real-time first-person shooter computer game*. IEEE Transaction on Computational Intelligence and AI in Games. Under 3rd review.
- (2) **D. Wang** & A. H. Tan. *Mobile humanoid agent with mood awareness for elderly care*. IEEE International Joint Conference on Neural Networks, 2014. Accepted.
- (3) **D. Wang**, B. Subagdja & A. H. Tan. *Creating human-like autonomous players in real-time first person shooter computer games*. In Proceedings of Conference on Innovative Applications of Artificial Intelligence, pages 173-178, 2009.

References

- [(AAFP) 14] American Academy of Family Physicians (AAFP). Available online: <http://www.familydoctor.org/>, 2014.
- [Abe 97] S. Abe. Neural networks and fuzzy systems: Theory and applications. Kluwer Academic Publishers, 1997.
- [(ACS) 14] American Cancer Society (ACS). Available online: <http://www.cancer.org/>, 2014.
- [Almuallim 94] H. Almuallim & T. G. Dietterich. *Learning Boolean concepts in the presence of many irrelevant features*. Artificial Intelligence, vol. 69, no. 1-2, pages 279–305, 1994.
- [Andrews 11] J. H. Andrews, T. Menzies & Felix C. H. L. *Genetic Algorithms for Randomized Unit Testing*. IEEE Transactions on Software Engineering, vol. 37, no. 1, pages 80–94, 2011.
- [Ang 03] K. K. Ang, C. Quek & M. Pasquier. *POPFNN-CRI(S): Pseudo outer product based fuzzy neural network using*

- the compositional rule of inference and singleton fuzzifier*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 33, no. 6, pages 838–849, 2003.
- [Ang 05] K. K. Ang & C. Quek. *RSPOP: Rough set-based pseudo outer-product fuzzy rule identification algorithm*. Neural Computation, vol. 17, no. 1, pages 205–243, 2005.
- [Angelov 05] P. P. Angelov & D. Filev. *Simpl_eTS: A simplified method for learning evolving Takagi-Sugeno fuzzy models*. In Proceedings of IEEE International Conference on Fuzzy Systems, pages 1068–1073, 2005.
- [Angelov 08] P. P. Angelov & X. Zhou. *Evolving fuzzy-rule-based classifiers from data streams*. IEEE Transactions on Fuzzy Systems, vol. 16, no. 6, pages 1462–1475, 2008.
- [Azeem 00] M. F. Azeem, M. Hanmandlu & N. Ahmad. *Generalization of adaptive neuro-fuzzy inference systems*. IEEE Transactions on Neural Networks, vol. 11, no. 6, pages 1332–1346, 2000.
- [Back 96] T. Back. *Evolutionary algorithms in theory and practice*. Oxford University Press, 1996.
- [Baddeley 75] A. Baddeley, N. Thomson & M. Buchanan. *Word length and the structure of short-term memory*. Journal of Verbal Learning and Verbal Behavior, vol. 14, no. 6, pages 575–589, 1975.

- [Baldwin 79] J. F. Baldwin. *A new approach to approximate reasoning using a fuzzy logic*. Fuzzy Sets and Systems, vol. 2, no. 4, pages 309–325, 1979.
- [Baldwin 80] J. F. Baldwin & N. C. F. Guild. *Feasible algorithms for approximate reasoning using fuzzy logic*. Fuzzy Sets and Systems, vol. 3, no. 3, pages 225–251, 1980.
- [Banerjee 98] M. Banerjee, S. Mitra & S. K. Pal. *Rough fuzzy MLP: Knowledge encoding and classification*. IEEE Transactions on Neural Networks, vol. 9, no. 6, pages 1203–1216, 1998.
- [Barakat 10] N. Barakat & A. P. Bradley. *Rule extraction from support vector machines: A review*. Neurocomputing, vol. 74, no. 1-3, pages 178–190, 2010.
- [Baraldi 99] A. Baraldi & P. Blonda. *A survey of fuzzy clustering algorithms for pattern recognition - part I and II*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 29, pages 778–785, 786–801, 1999.
- [Barlow 89] H. Barlow. *Unsupervised learning*. Neural Computation, vol. 1, pages 295–311, 1989.
- [Bernd 99] T. Bernd, M. Kleutges & A. Kroll. *Nonlinear black box modelling - fuzzy networks versus neural networks*. Neural Computing and Applications, vol. 8, pages 151–162, 1999.

- [Bezdek 81] J. C. Bezdek. Pattern recognition with fuzzy objective function algorithms. Kluwer Academic Publishers, 1981.
- [Bezdek 99] J. C. Bezdek, J. Keller, R. Krisnapuram & M. R. Pal. Fuzzy models and algorithms for pattern recognition and image processing. Kluwer Academic Publishers, 1999.
- [Bishop 95] C. Bishop. Neural networks for pattern recognition. Oxford University Press, 1995.
- [Blickle 95] T. Blickle & L. Thiele. *A comparison of selection schemes used in genetic algorithms*. Evolutionary Computation, vol. 4, no. 11, pages 311–347, 1995.
- [Bose 09] I. Bose & X. Chen. *Hybrid models using unsupervised clustering for prediction of customer churn*. Journal of Organizational Computing and Electronic Commerce, vol. 19, no. 2, pages 133–151, 2009.
- [Carpenter 88] G. A. Carpenter & S. Grossberg. *The ART of adaptive pattern recognition by a self-organizing neural network*. Computer: Special Issue on Artificial Neural Systems, vol. 21, pages 77–88, 1988.
- [Carpenter 91] G. A. Carpenter, S. Grossberg & D. B. Rosenm. *Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system*. Neural Networks, vol. 4, pages 759–771, 1991.

- [Caruana 88] R. A. Caruana & J. D. Schaffer. *Representation and hidden bias: Gray vs. binary coding for genetic algorithms*. In Proceedings of International Congress on Machine Learning, pages 153–161, 1988.
- [Casillas 03] J. Casillas, O. Cordon, F. Herrera & L. Magdalena, editors. Interpretability issues in fuzzy modeling, volume 128 of Studies in Fuzziness and Soft Computing. Springer, 2003.
- [Casslen 94] B. Casslen, T. Bossmar, I. Lecander & B. Astedt. *Plasminogen activators and plasminogen activator inhibitors in blood and tumour fluids of patients with ovarian cancer*. European Journal of Cancer, vol. 30A, no. 9, pages 1302–1309, 1994.
- [Chakraborty 04] D. Chakraborty & N. R. Pal. *A neuro-fuzzy scheme for simultaneous feature selection and fuzzy rule-based classification*. IEEE Transactions on Neural Networks, vol. 15, no. 1, pages 110–123, 2004.
- [Chen 04] M. Y. Chen & D. A. Linkens. *Rule-base self-generation and simplification for data-driven fuzzy models*. Fuzzy Sets and Systems, vol. 142, no. 2, pages 243–265, 2004.
- [Cherkassky 98] V. Cherkassky. *Fuzzy inference systems: A critical review*. In O. Kayak & L. A. Zadeh, editors, Computational Intelligence: Soft Computing and Fuzzy-

- Neuro Integration with Applications, pages 177–197. Springer, 1998.
- [Cho 05] S. Y. Cho & Z. Chi. *Genetic evolution processing of data Structures for image classification*. IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 2, pages 216–231, 2005.
- [Chouchoulas 01] A. Chouchoulas & Q. Shen. *Rough set-aided keyword reduction for text categorisation*. Applied Artificial Intelligence, vol. 15, no. 9, pages 843–873, 2001.
- [Coello 02] C. A. C. Coello & E. M. Montes. *Constraint-handling in genetic algorithms through the use of dominance-based tournament selection*. Advanced Engineering Informatics, vol. 16, no. 3, pages 193–203, 2002.
- [Cole 95] R. A. Cole & J. W. Gunther. *Separating the likelihood and timing of bank failure*. Journal of Banking and Finance, vol. 19, no. 6, pages 1073–1089, 1995.
- [Combs 98] W. E. Combs & J. E. Andrews. *Combinatorial rule explosion eliminated by a fuzzy rule configuration*. IEEE Transactions on Fuzzy Systems, vol. 6, no. 1, pages 1–11, 1998.
- [Conrad 64] R. Conrad & A. J. Hull. *Information, acoustic confusion and memory span*. British Journal of Psychology, vol. 55, no. 4, pages 429–432, 1964.

- [Cordon 00] O. Cordon & F. Herrera. *A proposal for improving the accuracy of linguistic modeling*. IEEE Transactions on Fuzzy Systems, vol. 8, no. 3, pages 335–344, 2000.
- [Cortes 95] C. Cortes & V. Vapnik. *Support-vector networks*. Machine Learning, vol. 20, no. 3, pages 273–297, 1995.
- [Cox 99] E. Cox. *The fuzzy systems handbook*. Morgan Kaufmann Publishers, 1999.
- [Deb 89] K. Deb & D. E. Goldberg. *An investigation of niche and species formation in genetic function optimization*. In Proceedings of International Conference on Genetic Algorithms, pages 42–50, 1989.
- [Deb 00] K. Deb. *An efficient constraint handling method for genetic algorithms*. Computer Methods in Applied Mechanics and Engineering, vol. 186, no. 2-4, pages 311–338, 2000.
- [Deb 01] K. Deb. *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, 2001.
- [Deep 11] K. Deep, K. P. Singh, M. L. Kansal & C. Mohan. *An interactive method using genetic algorithm for multi-objective optimization problems modeled in fuzzy environment*. Expert Systems with Applications, vol. 38, no. 3, pages 1659–1667, 2011.

- [Dorigo 96] M. Dorigo, V. Maniezzo & A. Coloni. *Ant system: Optimization by a colony of cooperating agents*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 26, no. 1, pages 29–41, 1996.
- [Eccles 04] I. Eccles & M. Su. *Illustrating the curse of dimensionality numerically through different data distribution models*. In Proceedings of International Symposium on Information and Communication Technologies, pages 232–237, 2004.
- [Elavarasi 11] S. A. Elavarasi, J. Akilandeswari & B. Sathiyabhama. *A Survey on partition clustering algorithms*. International Journal of Enterprise Computing and Business Systems, vol. 1, no. 1, pages 1–14, 2011.
- [(eMCH) 14] e-Medicine Consumer Health (eMCH). Available online: <http://www.emedicinehealth.com/>, 2014.
- [Espinosa 00] J. Espinosa & J. Vandewalle. *Constructing fuzzy models with linguistic integrity from numerical data-AFRELI algorithm*. IEEE Transactions on Fuzzy Systems, vol. 8, no. 5, pages 591–600, 2000.
- [(FIGO) 14] International Federation of Gynecology and Obstetrics (FIGO). Available online: <http://www.figo.org/>, 2014.
- [Frasconi 88] P. Frasconi, M. Gori & G. Soda. *Local feed-back multilayered networks*. Neural Computation, vol. 4, pages 120–130, 1988.

- [(FRBC) 14] Federal Reserve Bank of Chicago (FRBC). Available online: <http://http://www.chicagofed.org/>, 2014.
- [Glover 97] F. W. Glover & M. Laguna. *Tabu search*. Springer, 1997.
- [Goldberg 89] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Publishing, 1989.
- [Gonzalez 01] A. Gonzalez & R. Perez. *Selection of relevant features in a fuzzy genetic learning algorithm*. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 31, no. 3, pages 417–425, 2001.
- [Gougam 08] L. A. Gougam, M. Tribeche & F. Mekideche-Chafa. *A systematic investigation of a neural network for function approximation*. *Neural Networks*, vol. 21, no. 9, pages 1311–1317, 2008.
- [Grossberg 82] S. Grossberg. *Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control*. Boston studies in the philosophy of science. D. Reidel Publishers, 1982.
- [Hall 09] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann & I. H. Witten. *The WEKA data mining software: An update*. *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pages 10–18, 2009.

- [Han 10] H. Han & J. Qiao. *A self-organizing fuzzy neural network based on a growing-and-pruning algorithm*. IEEE Transactions on Fuzzy Systems, vol. 18, no. 6, pages 1129–1143, 2010.
- [Hartigan 79] J. A. Hartigan & M. A. Wong. *A K-means clustering algorithm*. Applied Statistics, vol. 28, no. 1, pages 100–108, 1979.
- [Hayashi 93] Y. Hayashi, J. J. Buckley & E. Czoga L. A. *Fuzzy neural network with fuzzy signals and weights*. International Journal of Intelligent Systems, vol. 8, pages 527–537, 1993.
- [Haykin 98] S. Haykin. *Neural networks: A comprehensive foundation*. Prentice Hall, 1998.
- [Holland 75] J. Holland. *Adaptation in natural and artificial systems*. MIT Press, 1975.
- [Hornik 89] K. Hornik, M. Stinchcombe & H. White. *Multi-layer feed-forward networks are universal approximators*. Neural Networks, vol. 2, pages 359–366, 1989.
- [Hu 06] X. Hu & J. Wang. *Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network*. IEEE Transactions on Neural Networks, vol. 17, no. 6, pages 1487–1499, 2006.

- [Huang 04] G.-B. Huang, P. Saratchandran & N. Sundararajan. *An efficient sequential learning algorithm for Growing And Pruning RBF (GAP-RBF) networks*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 34, no. 6, pages 2284–2292, 2004.
- [Hush 93] D. R. Hush & B. G. Horne. *Progress in supervised neural networks*. IEEE Signal Processing Magazine, vol. 10, no. 1, pages 8–39, Jan 1993.
- [Ishibuchi 07] H. Ishibuchi & Y. Nojima. *Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning*. International Journal of Approximate Reasoning, vol. 44, pages 4–31, 2007.
- [Jain 88] Anil K. Jain & Richard C. Dubes. Algorithms for clustering data. Prentice Hall, 1988.
- [Jang 93] J. S. R. Jang. *ANFIS: Adaptive network-based fuzzy inference systems*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 23, pages 650–684, 1993.
- [Jang 96] J. S. R. Jang, C. T. Sun & E. Mizutani. Neuro-fuzzy and soft computing. Pearson Education, 1996.
- [Jin 00] Y. Jin. *Fuzzy modeling of high-dimensional systems: Complexity reduction and interpretability improvement*. IEEE Transactions on Fuzzy Systems, vol. 8, no. 2, pages 212–221, 2000.

- [Juang 98] C. F. Juang & C. T. Lin. *An online self-constructing neural fuzzy inference network and its applications*. IEEE Transactions on Fuzzy Systems, vol. 6, no. 1, pages 12–32, 1998.
- [Juang 99] C. F. Juang & C. T. Lin. *A recurrent self-organizing neural fuzzy inference network*. IEEE Transactions on Neural Networks, vol. 10, no. 4, pages 828–845, 1999.
- [Juang 02] C. F. Juang. *A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms*. IEEE Transactions on Fuzzy Systems, vol. 10, no. 2, pages 155–170, 2002.
- [Juang 07] C. F. Juang, S. H. Chiu & S. W. Chang. *A self-organizing TS-type fuzzy network with support vector learning and its application to classification problems*. IEEE Transactions on Fuzzy Systems, vol. 15, no. 5, pages 998–1008, 2007.
- [Juang 08] C. F. Juang & Y. W. Tsao. *A self-evolving interval type-2 fuzzy neural network with online structure and parameter learning*. IEEE Transactions on Fuzzy Systems, vol. 16, no. 6, pages 1411–1424, 2008.
- [Juang 09] C. F. Juang, R. B. Huang & Y. Y. Lin. *A recurrent self-evolving interval type-2 fuzzy neural network for dynamic system processing*. IEEE Transactions on Fuzzy Systems, vol. 17, no. 5, pages 1092–1105, 2009.

- [Kang 05] S. Kang & C. Isik. *Partially connected feedforward neural networks structured by input types*. IEEE Transactions on Neural Networks, vol. 16, no. 1, pages 175–184, 2005.
- [Kartalopoulos 96] S. V. Kartalopoulos. Understanding neural networks and fuzzy logic: Basic concepts and applications. Wiley-IEEE Press, 1996.
- [Kasabov 99] N. K. Kasabov. *Evolving connectionist systems: A theory and a case study on adaptive speech recognition*. In Proceedings of International Joint Conference on Neural Networks, volume 5, pages 3002–3007, 1999.
- [Kasabov 01] N. K. Kasabov. *Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 31, no. 6, pages 902–918, 2001.
- [Kasabov 02] N. K. Kasabov & Qun Song. *DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction*. IEEE Transactions on Fuzzy Systems, vol. 10, pages 144–154, 2002.
- [Kennedy 95] J. Kennedy & R. Eberhart. *Particle swarm optimization*. In Proceedings of IEEE International Joint Conference on Neural Networks, pages 1942–1948, 1995.
- [Kim 99] J. Kim & N. K. Kasabov. *HyFIS: Adaptive neuro-fuzzy inference systems and their applications to nonlinear*

- dynamical systems*. Neural Networks, vol. 12, no. 9, pages 1301–1319, 1999.
- [Kira 92] K. Kira & L. Rendell. *A practical approach to feature selection*. In Proceedings of International Workshop on Machine Learning, pages 249–256, 1992.
- [Kirkpatrick 83] S. Kirkpatrick, C. D. Gelatt & M. P. Vecchi. *Optimization by simulated annealing*. Science, vol. 220, pages 671–680, 1983.
- [Ko 11] C. H. Ko & S. F. Wang. *Precast production scheduling using multi-objective genetic algorithms*. Expert Systems with Applications, vol. 38, no. 7, pages 8293–8302, 2011.
- [Kohonen 82] T. K. Kohonen. *Self-organized formation of topologically correct feature maps*. Biological Cybernetics, vol. 43, no. 1, pages 59–69, 1982.
- [Kolman 05] E. Kolman & M. Margalot. *Are artificial neural networks white boxes?* IEEE Transactions on Neural Networks, vol. 16, no. 4, pages 844–852, 2005.
- [Korkmaz 08] E. E. Korkmaz. *Multi-objective genetic algorithms for grouping problems*. Applied Intelligence, vol. 33, no. 2, pages 179–192, 2008.
- [Kosko 92] B. Kosko. Neural networks and fuzzy systems. Prentice Hall, 1992.

- [Koza 92] J. R. Koza. Genetic programming: On the programming of computers by means of natural selection. MIT Press, 1992.
- [(KRBD) 14] Kent Ridge Biomedical Data Set Repository (KRBD). Available online: <http://datam.i2r.a-star.edu.sg/datasets/krbd/>, 2014.
- [Kreinovich 93] V. Kreinovich, C. Quintana & O. Fuentes. *Genetic algorithms: What fitness scaling is optimal*. Cybernetics and Systems, vol. 24, no. 1, pages 9–26, 1993.
- [Ku 95] C. C. Ku & K. Y. Lee. *Diagonal recurrent neural networks for dynamic systems control*. IEEE Transactions on Neural Networks, vol. 6, no. 1, pages 144–156, 1995.
- [Kulkarni 08] A. Kulkarni & T. Pedersen. *Name discrimination and e-mail clustering using unsupervised clustering of similar contexts*. Journal of Intelligent Systems, vol. 17, no. 1-3, pages 703–722, 2008.
- [Kuncheva 04] L. I. Kuncheva. Combining pattern classifiers: Methods and algorithms. Wiley, 2004.
- [Kuo 01] R. J. Kuo, C. H. Chen & Y. C. Hwang. *An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network*. Fuzzy Sets and Systems, vol. 118, no. 1, pages 21–45, 2001.

- [Lane 86] W. R. Lane, S. W. Looney & J. W. Wansley. *An application of the cox proportional hazards model to bank failure*. Journal of Banking and Finance, vol. 10, no. 4, pages 511–531, 1986.
- [Lee 90] C. C. Lee. *Fuzzy logic in control systems: fuzzy logic controller - part I and II*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 20, no. 2, pages 404–435, 1990.
- [Lee 02] H. M. Lee, J. M. Chen & C. L. Liu. *Inconsistency resolution and rule insertion for fuzzy rule-based systems*. Journal of Information Science and Engineering, vol. 18, no. 2, pages 187–210, 2002.
- [Leng 06] G. Leng, T. M. McGinnity & G. Prasad. *Design for self-organizing fuzzy neural networks based on genetic algorithms*. IEEE Transactions on Fuzzy Systems, vol. 14, no. 6, pages 755–766, 2006.
- [Lespinsats 07] S. Lespinats, M. Verleysen, A. Giron & B. Fertil. *DD-HDS: A method for visualization and exploration of high-dimensional data*. IEEE Transactions on Neural Networks, vol. 18, no. 5, pages 1265–1279, 2007.
- [Li 06] X. Y. Li & N. Ye. *A supervised clustering and classification algorithm for mining data with mixed variables*. IEEE Transactions on Systems Man, and Cybernetics, Part A, vol. 36, no. 2, pages 396–406, 2006.

- [Lin 91] C. T. Lin & C. S. G. Lee. *Neural-network-based fuzzy logic control and decision system*. IEEE Transactions on Computers, vol. 40, no. 12, pages 1320–1336, 1991.
- [Lin 96a] C. T. Lin & C. S. G. Lee. *Neural fuzzy systems*. Prentice-Hall, 1996.
- [Lin 96b] J. X. Lin. *Integration of weighted knowledge bases*. Artificial Intelligence, vol. 83, no. 2, pages 363–378, 1996.
- [Lin 96c] J. X. Lin. *A semantics for reasoning consistently in the presence of inconsistency*. Artificial Intelligence, vol. 86, no. 1, pages 75–95, 1996.
- [Lin 97a] C. J. Lin & C. T. Lin. *An ART-based fuzzy adaptive learning control network*. IEEE Transactions on Fuzzy Systems, vol. 5, pages 477–496, 1997.
- [Lin 97b] T. Y. Lin & N. Cercone. *Rough sets and data mining*. Kluwer Academic Publishers, 1997.
- [Lin 05] C. J. Lin & W. H. Ho. *An asymmetry-similarity-measure-based neural fuzzy inference system*. Fuzzy Sets and Systems, vol. 152, no. 3, pages 535–551, 2005.
- [Lingras 01] P. Lingras. *Unsupervised rough set classification using GAs*. Journal of Intelligent Information Systems, vol. 16, no. 3, pages 215–228, 2001.
- [Liu 07] F. Liu, C. Quek & G. S. Ng. *A novel generic Hebbian ordering-based fuzzy rule base reduction approach*

- to Mamdani neuro-fuzzy system*. Neural Computation, vol. 19, no. 6, pages 1656–1680, 2007.
- [Liu 08] W. F. Liu, P. P. Pokharel & J. C. Principe. *The kernel Least-Mean-Square algorithm*. IEEE Transactions on Signal Processing, vol. 56, no. 2, pages 543–554, 2008.
- [Louis 05] S. J. Louis & C. Miles. *Playing to learn: Case-injected genetic algorithms for learning to play computer games*. IEEE Transactions on Evolutionary Computation, vol. 9, pages 669–681, 2005.
- [Lovato 10] A. V. Lovato & J. C. M. Oliveira. *Airplane level changes using fuzzy control*. In Proceedings of IEEE International Conference on Fuzzy Systems, pages 1–6, 2010.
- [Lozowski 96] A. Lozowski, T. J. Cholewo & J. M. Zurada. *Crisp rule extraction from perceptron network classifiers*. In Proceedings of International Conference on Neural Networks, volume Plenary, Panel, and Special Sessions, pages 94–99, 1996.
- [Lu 98] Y. Lu, N. Sundararajan & P. Saratchandran. *Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm*. IEEE Transactions on Neural Networks, vol. 9, no. 2, pages 308–318, 1998.

- [Lucas 06] S. M. Lucas & G. Kendall. *Evolutionary computation and games*. IEEE Computational Intelligence Magazine, vol. February, pages 10–18, 2006.
- [Mamdani 77] E. H. Mamdani. *Application of fuzzy logic to approximate reasoning using linguistic synthesis*. IEEE Transactions on Computers, vol. 26, no. 12, pages 1182–1191, 1977.
- [Meyerbase 07] A. Meyerbase, A. Saalbach, O. Lange & A. Wismuller. *Unsupervised clustering of fMRI and MRI time series*. Biomedical Signal Processing and Control, vol. 2, no. 4, pages 295–310, 2007.
- [Michalewicz 96] Z. Michalewicz. Genetic algorithms + data structures = evolution programs. Springer, 1996.
- [Miller 56] G. A. Miller. *The magical number seven, plus or minus two: Some limits on our capacity for processing information*. The Psychological Review, vol. 63, no. 2, pages 81–97, 1956.
- [Miller 96] B. L. Miller & D. E. Goldberg. *Genetic algorithms, tournament selection, and the effects of noise*. Complex Systems, vol. 9, pages 193–212, 1996.
- [Mirshahi 92] S. Mirshahi, E. Pujade-Lauraine, C. Soria, M. Mirshahi, J. Fretault, A. Bernadou & J. Soria. *D-dimer and CA 125 levels in patients with ovarian cancer dur-*

- ing antineoplastic therapy*. Cancer, vol. 69, no. 9, pages 2289–2292, 1992.
- [Mitra 00] S. Mitra & Y. Hayashi. *Neuro-fuzzy rule generation: Survey in soft computing framework*. IEEE Transactions on Neural Networks, vol. 11, no. 3, pages 748–768, 2000.
- [Mitra 01] S. Mitra, P. Mitra & S. K. Pal. *Evolutionary modular design of rough knowledge-based network using fuzzy attributes*. Neurocomputing, vol. 36, no. 1, pages 45–66, 2001.
- [Mitra 04] S. Mitra. *An evolutionary rough partitive clustering*. Pattern Recognition Letters, vol. 25, no. 12, pages 1439–1449, 2004.
- [Nauck 97] D. Nauck, F. Klawonn & R. Kruse. Foundations of neuro-fuzzy systems. John Wiley & Sons, 1997.
- [Nishikawa 11] H. Nishikawa & S. Ozawa. *Radial basis function network for multi-task pattern recognition*. Neural Processing Letters, vol. 33, no. 3, pages 283–299, 2011.
- [(NLM) 14] U.S. National Library of Medicine (NLM). Available online: <http://www.nlm.nih.gov/>, 2014.
- [Nock 05] R. Nock & F. Nielsen. *Semi-supervised statistical region refinement for color image segmentation*. Pattern Recognition, vol. 38, no. 5, pages 835–846, 2005.

- [Oliver 87] I. M. Oliver, D. J. Smith & J. R. C. Holland. *A study of permutation crossover operators on the travelling salesman problem*. In Proceedings of International Conference on Genetic Algorithms and Their Applications, pages 224–230, 1987.
- [Paiva 04] R. P. Paiva & A. Dourado. *Interpretability and learning in neuro-fuzzy systems*. Fuzzy Sets and Systems, vol. 147, no. 1, pages 17–38, 2004.
- [Pal 99] N. R. Pal & K. Pal. *Handling of inconsistent rules with an extended model of fuzzy reasoning*. Journal of Intelligent and Fuzzy Systems, vol. 7, no. 1, pages 55–73, 1999.
- [Pal 03] S. K. Pal, S. Mitra & P. Mitra. *Rough-fuzzy MLP: Modular evolution, rule generation, and evaluation*. IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 1, pages 14–25, 2003.
- [Pawlak 82] Z. Pawlak. *Rough sets*. International Journal of Information and Computer Science, vol. 11, no. 5, pages 341–356, 1982.
- [Pawlak 91] Z. Pawlak. Rough sets. Kluwer Academic Publishers, 1991.
- [Pedrycz 93] W. Pedrycz & A. Rocha. *Fuzzy-set based models of neurons and knowledge-based networks*. IEEE Transac-

- tions on Fuzzy Systems, vol. 1, no. 4, pages 254–266, 1993.
- [Peters 08] G. Peters, M. Lampart & R. Weber. *Evolutionary rough k-medoid clustering*. In J. F. Peters & A. Skowron, editors, Transactions on Rough Sets VIII, volume 5084 of *Lecture Notes in Computer Science*, pages 289–306. Springer, 2008.
- [Qu 04] Y. Qu & S. Z. Xu. *Supervised cluster analysis for microarray data based on multivariate Gaussian mixture*. Bioinformatics, vol. 20, no. 12, pages 1905–1913, 2004.
- [Quek 99] C. Quek & R. W. Zhou. *POPFNN-AAR(S): A pseudo outer-product based fuzzy neural network*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 29, no. 6, pages 859–870, 1999.
- [Quinlan 93] J. R. Quinlan. C4.5: Programs for machine learning. Morgan Kaufmann, 1993.
- [Rong 06] H.-J. Rong, N. Sundararajan, G.-B. Huang & P. Saratchandran. *Sequential Adaptive Fuzzy Inference System (SAFIS) for nonlinear system identification and prediction*. Fuzzy Sets and Systems, vol. 157, no. 9, pages 1260–1275, 2006.
- [Rong 11] H.-J. Rong, N. Sundararajan, G.-B. Huang & G.-S. Zhao. *Extended sequential adaptive fuzzy inference*

- system for classification problems*. Evolving Systems, vol. 2, no. 2, pages 71–82, 2011.
- [Ross 85] S. M. Ross. Introduction to probability models. Academic Press, 1985.
- [Rubio 10] J. J. Rubio, M. V. Diana & P. Jaime. *Backpropagation to train an evolving radial basis function neural network*. Evolving Systems, vol. 1, no. 3, pages 173–180, 2010.
- [Sanner 92] R. M. Sanner & J.-J. E. Slotine. *Gaussian networks for direct adaptive control*. IEEE Transactions on Neural Networks, vol. 3, no. 6, pages 837–863, 1992.
- [Setnes 98] M. Setnes, R. Babuska & H. B. Verbruggen. *Complexity reduction in fuzzy modeling*. Mathematics and Computers in Simulation, vol. 46, no. 5-6, pages 507–516, 1998.
- [Setnes 01] M. Setnes & R. Babuska. *Rule base reduction: Some comments on the use of orthogonal transforms*. IEEE Transactions on Systems, Man, and Cybernetics, Part C, vol. 31, no. 2, pages 199–206, 2001.
- [Shen 01] Q. Shen & A. Chouchoulas. *Rough set-based dimensionality reduction for supervised and unsupervised learning*. Applied Mathematics and Computer Science, Special Issue on Rough Sets and Their Applications, vol. 11, no. 3, pages 583–601, 2001.

- [Shen 02] Q. Shen & A. Chouchoulas. *A rough-fuzzy approach for generating classification rules*. Pattern Recognition, vol. 35, no. 11, pages 2425–2438, 2002.
- [Shen 04] Q. Shen & R. Jensen. *Selecting informative features with fuzzy-rough sets and its application for complex systems monitoring*. Pattern Recognition, vol. 37, no. 7, pages 1351–1363, 2004.
- [Sliutz 95] G. Sliutz, C. Tempfer, C. Kainz, G. Mustafa, G. Gitsch, H. Koelbl & C. Biegelmayer. *Tissue polypeptide specific antigen and cancer associated serum antigen in the follow-up of ovarian cancer*. Anticancer Research, vol. 15, no. 3, pages 1127–1129, 1995.
- [Slowinski 92] R. Slowinski, editor. *Intelligent decision support*. Kluwer Academic Publishers, 1992.
- [Song 01] Q. Song & N. Kasabov. *ECM: A novel on-line, evolving clustering method and its applications*. In Proceedings of Conference on Artificial Neural Networks and Expert Systems, pages 87–92, 2001.
- [Stanley 05] K. O. Stanley, B. D. Bryant & R. Miikkulainen. *Real-time neuroevolution in the NERO video game*. IEEE Transactions on Evolutionary Computation, vol. 9, pages 653–668, 2005.

- [Sugeno 88] M. Sugeno & G. T. Kang. *Structure identification of fuzzy model*. Fuzzy Sets and Systems, vol. 28, pages 13–33, 1988.
- [Sun 10] X. Y. Sun & D. W. Gong. *Surrogate model-assisted interactive genetic algorithms with individuals fuzzy and stochastic fitness*. Journal of Control Theory and Applications, vol. 8, no. 2, pages 189–199, 2010.
- [Takagi 85] T. Takagi & M. Sugeno. *Fuzzy identification of systems and its applications to modelling and control*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 15, no. 1, pages 116–132, 1985.
- [Tan 08] T. Z. Tan, C Quek, G. S. Ng & K. Razvi. *Ovarian cancer diagnosis with complementary learning fuzzy neural network*. Artificial Intelligence in Medicine, vol. 43, pages 207–222, 2008.
- [Tan 10] A. H. Tan, Y. H. Feng & Y. S. Ong. *A self-organizing neural architecture integrating desire, intention and reinforcement learning*. Neurocomputing, vol. 73, no. 7-9, pages 1465–1477, 2010.
- [Tay 02] E. H. Tay & L. Shen. *Economic and financial prediction using rough sets model*. European Journal of Operational Research, vol. 141, pages 643 – 661, 2002.
- [Teng 08] T. H. Teng, Z. M. Tan & A. H. Tan. *Self-organizing neural models integrating rules and rein-*

- forcement learning*. In proceedings of IEEE International Joint Conference on Neural Networks, pages 3771–3778, 2008.
- [Tsukamoto 79] Y. Tsukamoto. *An approach to fuzzy reasoning method*. Advances in Fuzzy Set Theory and Applications, pages 137–149, 1979.
- [Tsumoto 98] S. Tsumoto. *Extraction of experts’ decision rules from clinical databases using rough set model*. Journal of Intelligent Data Analysis, vol. 2, no. 3, pages 215–227, 1998.
- [Tu 12] C. C. Tu & C. F. Juang. *Recurrent type-2 fuzzy neural network using Haar wavelet energy and entropy features for speech detection in noisy environments*. Expert Systems with Applications, vol. 39, no. 3, pages 2479 – 2488, 2012.
- [Tung 02a] W. L. Tung & C. Quek. *DIC: A novel discrete incremental clustering technique for the derivation fuzzy membership functions*. In Proceedings of Pacific Rim International Conference on Artificial Intelligence, volume 178-187, 2002.
- [Tung 02b] W. L. Tung & C. Quek. *GenSoFNN: A generic self-organizing fuzzy neural network*. Neural Networks, vol. 13, pages 1075–1086, 2002.

- [Tung 04] W. L. Tung, C. Quek & P. Cheng. *GenSo-EWS: A novel neural-fuzzy based early warning system for predicting bank failures*. Neural Networks, vol. 17, no. 4, pages 567–587, 2004.
- [Turksen 90] I. B. Turksen & Z. Zhong. *An approximate analogical reasoning schema based on similarity measures and interval-valued fuzzy sets*. Fuzzy Sets and Systems, vol. 34, no. 3, pages 323–346, 1990.
- [(UCI) 14] Machine Learning Repository (UCI). Available online: <http://archive.ics.uci.edu/ml/>, 2014.
- [Vinterbo 97] S. Vinterbo & A. Ohrn. *A rough set approach to clustering*. In Proceedings of Joint Conference of Information Sciences, volume 3, pages 383–386, 1997.
- [Wang 94] L. X. Wang. Adaptive fuzzy systems and control: Design and stability analysis. Prentice-Hall, 1994.
- [Wang 98] L. Wang. *Effects of noise in training patterns on the memory capacity of the fully connected binary Hopfield neural network: Mean-field theory and simulations*. IEEE Transactions on Neural Networks, vol. 9, no. 4, pages 697–704, 1998.
- [Wang 03] W. Y. Wang & Y. H. Li. *Evolutionary learning of BMF fuzzy-neural networks using a reduced-form genetic algorithm*. IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 33, no. 6, pages 966–976, 2003.

- [Wang 04] D. Wang, C. Quek & G. S. Ng. *Novel self-organizing Takagi-Sugeno-Kang fuzzy neural networks based on ART-like clustering*. Neural Processing Letters, vol. 20, no. 1, pages 39–51, 2004.
- [Wang 08] L. Wang, N. Zhou & F. Chu. *A general wrapper approach to selection of class-dependent features*. IEEE Transactions on Neural Networks, vol. 19, no. 7, pages 1267–1278, 2008.
- [Wang 09] D. Wang, B. Subagdja & A. H. Tan. *Creating human-like autonomous players in real-time first person shooter computer games*. In Proceedings of Conference on Innovative Applications of Artificial Intelligence, pages 173–178, 2009.
- [Wang 14a] D. Wang, C. Quek & G. S. Ng. *Bank failure prediction using an accurate and interpretable neural fuzzy inference system*. AI Communications, Special Track on Soft Computing in Finance and Economics, 2014. Accepted.
- [Wang 14b] D. Wang, C. Quek & G. S. Ng. *Ovarian cancer diagnosis using a hybrid intelligent system with simple yet convincing rules*. Applied Soft Computing, 2014. In press.
- [Wong 09] W. K. Wong, X. H. Zeng & W. M. R. Au. *A decision support tool for apparel coordination through inte-*

- grating the knowledge-based attribute evaluation expert system and the TS fuzzy neural network.* Expert Systems with Applications, vol. 36, no. 2, pages 2377–2390, 2009.
- [Wu 01] S. Q. Wu, M. J. Er & Y. Gao. *A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks.* IEEE Transactions on Fuzzy Systems, vol. 9, no. 4, pages 578–594, 2001.
- [Wu 07] C. Wu, G. Tzeng, Y. Goo & W. Fang. *A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy.* Expert Systems with Applications, vol. 32, no. 2, pages 397–408, 2007.
- [Xu 05] R. Xu & D. Wunsch. *Survey of clustering algorithms.* IEEE Transactions on Neural Networks, vol. 16, no. 3, pages 645–678, 2005.
- [Yao 98] Y. Y. Yao. *A comparative study of rough sets and fuzzy sets.* Journal of Information Sciences, vol. 109, pages 227–242, 1998.
- [Yen 99] J. Yen & L. Wang. *Simplifying fuzzy rule-based models using orthogonal transformation methods.* IEEE Transactions on Systems, Man, and Cybernetics, Part B, vol. 29, no. 1, pages 13–24, 1999.

- [Zadeh 65] L. A. Zadeh. *Fuzzy Sets*. Information Control, vol. 8, pages 338–353, 1965.
- [Zadeh 75] L. A. Zadeh. *The concept of a linguistic variable and its application to approximate reasoning - part I, II, and III*. Information Sciences, vol. 8, 8, 9, pages 199–249, 301–357, 43–80, 1975.
- [Zadeh 94] L. A. Zadeh. *Fuzzy logic, neural networks and soft computing*. Communications of ACM, vol. 37, no. 3, pages 74–84, 1994.
- [Zadeh 96] L. A. Zadeh. *Fuzzy logic = computing with words*. IEEE Transactions on Fuzzy Systems, vol. 4, no. 2, pages 103–111, 1996.
- [Zadeh 04] L. A. Zadeh. *Computing with words and its applications*. Technical report, University of California, Berkeley, 2004.
- [Zahalka 10] J. Zahalka & F. Zelezny. *An experimental test of Occam's razor in classification*. Machine Learning, vol. 82, no. 3, pages 475–481, 2010.