Rule extraction frameworks using rough sets and neural networks

Xu, Yi

2005

Xu, Y. (2005). Rule extraction frameworks using rough sets and neural networks. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/2544>

<https://doi.org/10.32657/10356/2544>

Nanyang Technological University

Downloaded on 26 Apr 2023 10:14:54 SGT

Rule Extraction Frameworks Using Rough Sets and Neural Networks

Xu Yi

School of Computer Engineering

A thesis submitted to Nanyang Technological University
in fulfillment of the requirement for the degree of
Master of Engineering

2005

QA
76.9
•D343
X8
2005

Acknowledgements

The author is indebted to members of her thesis committee for their invaluable comments and suggestions. The author owes profound gratitude to her supervisor, Associate Professor Narendra S. Chaudhari, for his guidance, support, time and effort throughout the preparation of the thesis. It has been a very rewarding experience for her not only in academic research and training, but also in more broad senses. As the great Chinese poet Du Fu (Tu Fu) put it, "When a poet makes a poem, all his effort is beyond the poem."

The author appreciates many people in Nanyang Technological University for their help and support during this study.

Finally, the author thanks her parents, for their love, encouragement and sacrifices. She dedicates this work to them.

Table of Contents

Acknowledgements	I
List of Figures	VI
List of Tables.....	VII
Abstract.....	X
 Chapter 1 Introduction	
1.1 Background	1
1.2 Motivations	3
1.3 Objectives.....	4
1.4 Organization	5
 Chapter 2 Literature Review	
2.1 Data Mining.....	7
2.2 Rule Induction.....	10
2.2.1 Overview.....	10
2.2.2 Decision Trees	11
2.2.3 Association Rules.....	13
2.2.4 Fuzzy Sets	14

2.2.5	Neural Networks.....	16
2.2.6	Rough Sets	17
2.3	Soft Computing Framework	22
2.4	Summary	26
 Chapter 3 Rule Extraction Framework		
3.1	Rule Extraction Framework	27
3.2	Discretization	28
3.3	Attribute Selection	31
3.3.1	Data Clustering with SOM.....	31
3.3.2	The SOM Neural Network	32
3.4	Attribute Reduction	35
3.5	Rule Generation	35
3.6	Summary	37
 Chapter 4 Generalization of Attribute Reducts		
4.1	Binary Discernibility Matrix	38
4.2	Binary Representation of Attributes	41
4.3	Solve the Inconsistency in Decision Table	42

4.3.1	Explanation of Algorithm 4.1	44
4.3.2	A Note on the Number of Operations.....	45
4.4	Attribute Reduction by Rough Sets	45
4.4.1	Explanation of Algorithm 4.2	48
4.4.2	A Note on the Number of Operations.....	49
4.5	Example	49
4.6	Summary	52
 Chapter 5 Rule Generation		
5.1	Decision Rules	53
5.2	Algorithm for Rule Generation	54
5.2.1	Explanation of Algorithm 5.1	57
5.2.2	A Note on the Number of Operations.....	57
5.3	Rule Simplification.....	58
5.3.1	Explanation of Algorithm 5.2	59
5.3.2	A Note on the Number of Operations.....	59
5.4	Example	59
5.5	Summary	61

Chapter 6 Experiments and Discussions

6.1	Experiment One	62
6.1.1	Data Discretization.....	62
6.1.2	Attribute Selection with SOM	64
6.1.3	Attribute Reduction by Rough Sets	68
6.1.4	Generation of Rules	70
6.2	Experiment Two.....	71
6.3	Experiment Three	72
6.4	Experiment Four	74
6.5	Experiment Five	75
6.6	Summary.....	76

Chapter 7 Conclusions and Extensions

7.1	Conclusions	77
7.2	Extensions	79

List of Figures

Figure 2.1 The KDD Process.....	8
Figure 3.1 The Architecture of Rule Extraction Framework.....	28
Figure 4.1 Location of the Condition Attributes in a Binary String.....	42
Figure 5.1 Flowchart of Rule Generation Algorithm.....	55
Figure 6.1 Result of Iris Data by SOM.....	66

List of Tables

Table 4.1 A Sample Information System.....	40
Table 4.2 The Binary Discernibility Matrix for ($d=2$) Corresponding to the Sample Information System in Table 4.1.	40
Table 4.3 An Example for Information System.....	49
Table 4.4 Approximations for Decision Attributes.....	50
Table 4.5 Binary Discernibility Matrix for Decision "Accept" (Lower Approximation).....	51
Table 4.6 Binary Discernibility Matrix for Decision "Accept" (Upper Approximation).....	51
Table 4.7 Discernibility Degree of Condition Attributes.....	52
Table 5.1 Information System of the Application.....	61
Table 6.1 Sample of Iris Data.....	63
Table 6.2 Result of Discretization on Iris Data	63
Table 6.3 Decision Table for Iris Data.....	67
Table 6.4 Discernibility Degree of Attributes to the Decision "Iris-setosa"	68

Table 6.5 Binary Discernibility Matrix for "Iris-versicolor"

(Lower Approximation)..... 68

Table 6.6 Binary Discernibility Matrix for "Iris-versicolor"

(Upper Approximation). 69

Table 6.7 Discernibility Degree of Attributes to "Iris-versicolor"

(Lower Approximation)..... 69

Table 6.8 Discernibility Degree of Attributes to "Iris-versicolor"

(Upper Approximation). 69

Table 6.9 Values of Condition Attributes. 71

Table 6.10 Results of Robustness Experiments on MONK3 Dataset..... 72

Table 6.11 Comparison of Rule Induction Methods

for Australian Credit Data..... 72

Table 6.12 Comparison of Rule Induction Methods for

Breast Cancer (Wisconsin) Data..... 73

Table 6.13 Comparison of Rule Induction Methods for Flags Data..... 73

Table 6.14 Comparison of Rule Induction Methods

for Lung Cancer Data..... 74

Table 6.15 Comparison of Rule Induction Methods for Car Data..... 74

Table 6.16 Effectiveness of Discretization Method for Rule Induction..... 75

Table 6.17 Processing Time of Finding Attribute Reduct

(Time is given in seconds)..... 75

Abstract

In 1980's, rough sets have been introduced by Z. Pawlak as a mathematical approach to deal with vagueness and uncertainty of information. Thereafter, there are successful applications to reduce the number of attributes in datasets by using rough sets. Most of the previous works use rough sets as a preprocessing tool of neural classifier for feature selection. However, the accuracy is low because the classification region defined by rough sets is relatively simple. In order to solve this problem, we propose a hybrid system combining rough sets and neural networks for rule generation. In our present approach, we make the Self Organizing Map (SOM) network as preprocessing procedure to rough sets. This phase reduces the size of the training data to save analyzing time; additionally it improves the accuracy of prediction. After clustering phase by SOM, extracting rules from decision tables becomes straightforward by using rough sets. Since the optimal reduct will decrease the size of the condition attributes, the problem of selecting minimal attribute subsets to form the reduct is important. We propose a new attribute reduction algorithm based on binary discernibility matrix to obtain optimal reducts from decision tables. We also present a rule generation algorithm to get rules from lower and upper approximations. To evaluate the performance of our framework, we compare our approach with other rule induction methods: C4.5 and the Subset algorithm on five benchmark datasets in the experiment. The results show that the average accuracy of rules of our approach is higher than that of the two machine learning methods.

Chapter 1 Introduction

1.1 Background

With the extraordinary expansion of data in our lives, data mining has grown into a popular field. However, it is still young and in its early stage. Data mining covers many different aspects, including classification, clustering, association rules, time series analysis and prediction. Machine learning, statistics, neural computing, fuzzy system and some techniques of artificial intelligence have been employed as data analysis techniques to solve many practical problems in data mining. These techniques also draw from mathematical approaches such as approximation theory and dynamical systems. The way to solve problems in data mining is to try all the relevant methods that can be used to find useful knowledge in data.

As we have mentioned above, the purpose of data mining is to use one or more of techniques to automatically analyze and extract knowledge from data contained within a data repository [1]. Data mining does this by building a model of the real world based on data collected from different sources. The result of the model building

is a description of relationship of the data that can be used for prediction. Through the model, we can predict or classify what category or class a case falls.

Traditionally, decision trees are used to generate understandable rules. Knowledge in the form of rules, decision trees, etc, induced by learning from training examples, is easy for human to comprehend. Besides, such rules may be practically used by putting them into the knowledge base of rule-based system. However, it is difficult to represent general relationship of datasets since all rules must begin with the same feature in decision trees. They are unmanageable for large quantity of information in databases, which is a typical situation in data mining applications.

Neural networks are frequently applied for data mining because they have lower classification error rates and are robust to noise. Nevertheless, neural networks lack explanation facilities for the knowledge.

The modeling of imprecise and qualitative knowledge, ambiguous decision and the transmission of uncertainty are possible through the use of fuzzy sets. However, a fundamental difficulty with fuzzy set theory is the semantical interpretations of the degrees of membership.

The limitations of the above techniques are the main motivation for this research. We propose a methodology to overcome the limitations of individual technique by creating a hybrid system.

1.2 Motivations

The rule extraction techniques have been widely developed and used for data mining in many application areas, such as medical diagnosis, decision-making, classification and prediction. Rough set theory, proposed by Z. Pawlak in the early 1980's [2], has been used as rule extraction method for machine learning, knowledge discovery, and expert systems [3]. This technique has offered useful tools to discover hidden deterministic rules and associative patterns in all types of data. It has several distinguishing features in contrast with other techniques for data mining. Firstly, it can present knowledge in a clear mathematical manner. Secondly, it derives rules only from facts present in data. Thirdly, it can reduce information system to its simplest form. Rough sets can be used in different phases of the knowledge discovery process, like attribute selection, attribute extraction, data reduction, decision rule generation and pattern extraction. Rough set theory is a new approach to data analysis. The theory has many important advantages, such as providing efficient algorithms for finding hidden patterns in data, finding minimal sets of data, and generating sets of decision rules from data. These features and their combinations in data analysis can solve more complex and intrigued problems. On the other hand, neuro-fuzzy integration systems, which have been proposed to obtain knowledge by instruction or learning, lack rigid mathematical model. Moreover, fuzzy system modifies its fuzzy associations from representative numerical samples. However, rough sets can complement the weakness of neural networks in this situation. Thus, implementation of rough set theory in data mining, together with neural networks to construct a new architecture of hybrid system, is promising.

Another motivation for this research is more technically oriented. In rough set method, information is stored in tables. Two sets of rules, certain rules and possible rules [4] deal with relationship of data in these tables. However, complex data in tables are usually difficult to be analyzed especially when the number of attributes is very large. In this situation the rough set approach is weak. From a computational view, it is not practical. Another drawback of the existing rough set models is the inefficiency of computing the core and reducts (Reducts are those minimal attribute sets of information system, which keep the same classification capability with original attribute set [2]) to discern attributes. To overcome these problems, we have developed a new algorithm for generating reduct. Our algorithm is based on binary discernibility matrix. We demonstrate through experiments that our new hybrid system is equivalent to the traditional rough set model; additionally it is more efficient and scalable.

1.3 Objectives

The focus of this thesis is on the efficient rough neural system for rule generation based on rough sets. The ultimate goal is to generate useful data mining rules from large datasets more efficiently. Our method, which combines neural networks and rough sets, is based on rigid mathematical model. In addition, rules are generated from decision table rather than from sample data through adaptively learning. The specific objectives of our research are:

- To develop a hybrid framework that makes use of rough sets and neural networks for their complementary features.

- To present a new reduct generation algorithm based on binary discernibility matrix, which can improve efficiency in the attribute reduction phase.
- To design a new rule generation algorithm, which is used to produce certain and approximate decision rules from large datasets.

1.4 Organization

This thesis is organized into seven chapters.

Chapter 2 investigates existing rule induction techniques and soft computing frameworks in data mining. This chapter gives an overview on state-of-art techniques in this field. We contrast advantages and disadvantages of different models and methods.

Chapter 3 presents our proposal of hybrid framework for rule extraction and the outline steps of the framework. Functions and steps of each part of the framework are described in detail. In our framework, we combine Self-organizing Map (SOM) network and rough sets to generate rules.

Chapter 4 proposes attribute reduction algorithm to find optimal attribute reduct. This algorithm makes use of characteristic of the binary discernibility matrix. This method shows lower complexity and higher classification ability in contrast with other method.

Chapter 5 describes rule generation algorithm and applies rule simplification algorithm to get rule sets. Two kinds of rules: certain rules and approximate rules are obtained from the lower and upper approximations.

Chapter 6 gives the experimental results on our rule extraction system and some comparisons. Analysis of the utilization of entropy based discretization method and SOM are given through experiments. Furthermore, the experimental results for processing time are discussed.

Finally, Chapter 7 summarizes the status of our research in terms of the experimental results and gives recommendations for extensions of our work.

Chapter 2 Literature Review

2.1 Data Mining

Data mining is a process of efficiently extracting previously unknown information and finding hidden patterns or correlations from large databases. Different techniques are used to analyze and summarize the data into useful information for decision-making. Thus, data mining algorithms employ a wide variety of models from machine learning, statistics, artificial intelligence and databases [5].

The term KDD refers to the overall process of *knowledge discovery in databases* [1]. Data mining is a particular step in this process, involving applying specific algorithms for extracting patterns from data. The overall KDD process is outlined in Figure 2.1, which includes data selection, data preprocessing, data transformation, data mining and interpretation.

Figure 2.1 The KDD Process.

We give a brief example to illustrate data mining process. There are some medical datasets in clinic center, from which we wish to find out what kind of people are easy to get heart disease. Firstly, we remove irrelevant dimensions, and keep some important attributes, such as age, sex, blood pressure and so on. This process is called "data selection". Secondly, we need to remove incomplete and inconsistent data for preprocessing. In order to make the input data an appropriate form for data mining, datasets are transformed by normalization, scaling or summary operations. After we have used some models to extract patterns or rules from datasets, the rule extraction process may generate very concise knowledge as: "people who have high blood pressure are easy to get heart disease".

Several data mining techniques had been studied and applied in almost every imaginable scenario as long as data is abundant. The common techniques in current data mining practice include the following:

- (1) Classification: classifies a data item into one of several predefined categorical classes.
- (2) Regression: maps a data item to a real-valued prediction variable.
- (3) Clustering: maps a data item into one of several clusters, where clusters are natural groupings of data items based on similarity metrics or probability density models.
- (4) Rule generation: extracts classification rules from data.
- (5) Discovering association rules: describe association relationship among different attributes.
- (6) Summarization: provides a compact description for a subset of data.
- (7) Dependency modeling: describes significant dependencies among variables.
- (8) Sequence analysis: models sequential patterns, like time-series analysis. The goal is to model the states of the process generating the sequence or to extract and report deviation and trends over time.

Techniques used in data mining can be applied to different data for prediction, classification and generalization. Many decision makers utilize data mining tools to manage their massive data to improve performance and efficiency for their decisions. For instance, cluster and mining association rule tools are very well-known in business applications for discovering the customers' habits to make more benefits from them. Decision tree and rough set techniques are widely applied to knowledge acquisition applications in machine learning, medical and expert system, etc.

2.2 Rule Induction

2.2.1 Overview

Knowledge-based system is a popular technique in construction engineering and management. One of the most important problems in developing knowledge-based system is knowledge acquisition from experts. However, the bottleneck in development and implementation is that conventional knowledge-based systems (e.g. rule-based expert systems) cannot perform high-level tasks like induction. The developed knowledge-based systems can only resolve well-defined problems. Consequently, rule induction technique could have a profound effect on future intelligent systems [6].

In general, the induced decision rules can be mostly used for the following perspectives [7]:

(1) Creating classification systems.

The aim is to automatically find a collection of decision rules from the set of learning examples. These rules are used to classify new examples.

(2) Performing data exploration and knowledge discovery process.

The main purpose of discovery perspective is to extract unknown information patterns represented in the form of decision rules from datasets. Such discovered rules can facilitate understanding and explaining dependencies between values of attributes and definitions of decision classes.

The most popular perspectives of rule induction are classification and knowledge discovery. The evaluation of decision rules is quite different depending on the perspective. In classification-oriented induction, rules are parts of a classification system. Therefore, the evaluation refers to a complete set of rules. The evaluation criterion is usually unique and defined as classification accuracy. In discovery-oriented induction, each rule is evaluated individually and independently as a possible representation of an interesting pattern. The evaluation criteria are multiple and considering them together is not an easy task. Moreover, the definition of criteria depends on the application problem and the user's requirements [8].

Since the perspectives of rule induction are quite different, the derived rules may have different usefulness for the two perspectives. For classification task, the minimal set of rules seems to be more useful. However, satisfactory and exhaustive set of rules are mainly useful for knowledge discovery problems.

There are many different models for rule induction, such as decision trees, association rules, neural networks, fuzzy sets and rough sets.

2.2.2 Decision Trees

Decision trees algorithms are based on divide-and-conquer approach to the classification problem. They work in a top-down manner, seeking at each stage an attribute to split on, that separates the classes best, and then recursively processing the partitions resulted from the split. Quinlan's C4.5 [9] is probably the most popular one.

In decision tree induction, a rule may be constructed by forming a conjunct of every test that occurs on a path between the root node and a leaf node of a tree. The

collection of all such rules obtained by traversing every unique path from root node to leaf node is a corresponding rule based solution for classification.

Decision tree induction consists of the following steps within data mining:

- *Creation of decision trees.*

Decision trees are simple knowledge representation and they classify examples to a finite number of classes. The nodes are labeled with attribute names. The edges are labeled with possible values for this attribute and the leaves are labeled with different classes. Objects are classified by following a path down the tree, by taking the edges, corresponding to the values of the attributes in an object.

- *Rule induction.*

A data mining system has to infer a model from the database. This model constructs classes. These classes are based on some attributes, called "predicting attributes", which are used to construct these classes. Class can then be defined by condition on the predicting attributes. This condition (or a set of conditions) gives us the rules that govern the classification. In other words, the system can find the description of each class.

The ability of decision trees to generate understandable rules is the greatest strength of this technique. As it is generally easy to follow any one path through the tree, the rule for particular classification or prediction is relatively straightforward. However, decision trees may be viewed as a simplistic approach to rule discovery

because most decision-tree algorithms only examine a single field at a time. It is difficult to represent general relationship of datasets. In decision tree approach, all rules must have the same attribute at the root level.

2.2.3 Association Rules

An important area of data mining deals with the discovery of association rules. As the name implies, association rule techniques are used to discover interesting associations between attributes contained in a database. A Boolean association involves binary attributes. A generalized association involves hierarchical attributes. In addition, a quantitative association involves categorical attributes. Unlike traditional production rules, association rules can have one or several output attributes.

Association rules are a popular technique for market analysis as all possible combinations of potentially interesting product groupings can be explored. For this reason, we can generate hundreds of association rules using a limited number of attributes.

When there is large number of possible conditions for each rule, the process of mining association rules becomes unreasonable. Thus, some special algorithms have been developed to generate association rules efficiently. One of them is the *apriori* algorithm [10]. Two steps are completed during the process:

- Generate item sets.
- Create a set of association rules using the generated item sets.

Association rules have proved to be a practical tool in order to find tendencies in data mining. Nevertheless, the large number of rules obtained even from small-sized databases limit their effective use. The existence of a large number of association rules makes them unmanageable for any human user.

2.2.4 Fuzzy Sets

In the classical set theory, a set can be represented by enumerating all its elements. Thus, the notion

$$A = \{a_1, a_2, \dots, a_n\}$$

represents a set A having elements $a_1, a_2, \dots, a_n$. If these elements $a_i (i = 1, \dots, n)$ of A are a subset of the universal base set X , the set A can be represented for all elements $x \in X$ by its characteristic function (2.1).

$$\mu_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{otherwise} \end{cases} \quad (2.1)$$

In classical set theory $\mu_A(x)$ has only two values 0("false") and 1("true"). Such sets are also called crisp sets.

Fuzzy sets are distinguished from the crisp sets by the fact that the boundary of a fuzzy set is not precise [11]. The characteristic function for a set X returns a value indicating the degree of membership of an element x in X . For a crisp set, the characteristic function returns a value in $\{0, 1\}$. A fuzzy set is non-crisp, and was introduced by L. Zadeh [12]. In contrast with a crisp set, the characteristic function

for a fuzzy set returns a value in $[0, 1]$. Let U be a universe of objects, X be subset of U , A be a fuzzy set in U , and x be an individual object x in X , respectively. For a set X , $\mu_A : X \rightarrow [0,1]$ is a function which determines the degree of membership of an object x in X . A fuzzy set A is then defined to be a set of ordered pairs as in (2.2):

$$A = \{(x, \mu_A(x)) | x \in X\} \quad (2.2)$$

The set X is called the reference set. The counterpart of intersection and union (crisp sets) are t-norm and s-norm operators in fuzzy set theory. In many fuzzy set systems, membership functions of complement, intersection, and union of fuzzy sets are defined based solely on membership functions of the fuzzy sets involved. Another method to interpret fuzzy sets is to through the notions of α -level sets. One of the main advantage of the set based representation is that it explicitly establishes a connection between fuzzy sets and crisp sets.

A fundamental difficulty with fuzzy set theory is the semantical interpretations of the degrees of membership. Although one can define fuzzy set operators in the min-max system using set operators through α -level sets, it is mainly a technical result rather than a semantical interpretation. There are some solutions to this difficulty. One solution is the use of product logic, which interprets a many-valued logic based on the semantics of lesser-valued systems. The model proposed by G. J. Klir [13] corresponds to this solution. Another solution is the probabilistic approach to many-valued logic. The set-oriented view of rough sets corresponds to this solution.

2.2.5 Neural Networks

Neural networks can capture "hidden" relations between inputs and outputs with a high approximation. However, trained ANNs are often viewed as "black boxes", which map input data to an output through a number of connected weighted connections and neurons in the layers. This can be a major obstacle in domains where symbolic rules or other forms of knowledge structure are understandable to human experts. Therefore, extraction of small set of meaningful rules to represent these "hidden" relations has been investigated in last two decades by many researchers. This serves to identify the attributes that, either individually or in a combination, are the most significant determinants of the decision or classification. Unlike fuzzy sets, the main contribution of neural networks toward data mining is rule extraction and clustering [1].

(1) Rule Extraction.

In general, the primary input to a connectionist rule extraction algorithm is a representation of the trained neural network, in terms of its nodes, links and sometimes the dataset. One or more hidden and output units are used to automatically derive the rules, which may later be combined and simplified to arrive at a more comprehensible rule set. These rules can also provide new insights into the application domain. The models are usually suitable in data-rich environments.

However, there are some limitations for rule extraction methods for neural networks. In general, a large number of rules, each applicable to only a small

number of samples, do not provide meaningful or useful knowledge to the users. Hence, a balance must be achieved between rule accuracy and rule simplicity. What's more, computational complexity of the rule extraction process may be a limiting factor on what is achievable from such techniques.

(2) Clustering and Self-organization.

One of the big challenges of data mining is the organization and retrieval of documents from archives. T. Kohonen et al. [14] have demonstrated the utility of a huge SOM with more than one million nodes to partition a little less than seven million patent abstracts where the documents are represented by 500-dimensional feature vectors. D. Shalvi and N. D. Claris [15] have designed a data mining technique, combining Kohonen's self-organizing neural network with data visualization for clustering a set of pathological data containing information regarding the patient's drugs, body locations and physiological abnormalities.

2.2.6 Rough Sets

Rough set theory was introduced by Z. Pawlak in 1982 as a mathematical approach to deal with vagueness and uncertainty of information [2]. Rough set theory is based on refusing certain set boundaries. Every set can be roughly defined using its lower and upper approximation. Definition of rough sets is as follows:

A set D is rough with respect to a collection of sets C if it has a non-empty boundary when approximated by C . Otherwise it is crisp.

Rough-set logics allow correct reasoning and hidden-association discovery based on uncertain and inconsistent data. Reduction of knowledge by rough sets is used in inducing decision rules according to a specific class [16]. The reduction of knowledge eliminates condition attributes without affecting a decision attribute. After analyzing the relation of attribute and removing dispensable attribute, it induces a set of minimal rules.

Rough sets make it easy to convert data into knowledge. The advantages include:

- (1) No previous knowledge is needed on data.
- (2) No assumption is needed about the dependency of attributes.
- (3) Noisy data can be easily handled.

The rough set theory is mainly involved in the classificatory analysis of imprecise, uncertain or incomplete information acquired from experience. From the set-oriented view, the rough membership functions in rough sets may be interpreted as a special type of fuzzy membership functions in fuzzy sets, which can be interpreted in terms of probabilities defined simply for cardinalities of sets. With this view, rough set theory may be regarded as being a deviation of classical set theory, in the same way that fuzzy set theory is viewed. However, rough set operators are no longer truth-functional. A set is approximated by a pair of sets called lower and upper approximations. Fuzzy sets allow partial set memberships to handle vagueness, while rough sets allow multiple set memberships to deal with indiscernibility. In the fuzzy set theory, the membership value of an element does not depend on other elements. In contrast, with respect to an equivalent relation, the membership value of an element depends on other elements in the rough set theory.

Z. Pawlak [17] extends his rough approximate concept to create a new approximate analysis method including a program and an algorithm for approximate classification. This new approach can also be used in approximate clustering and approximate learning algorithms.

M. Beynon et al. [18] create a classification and rule induction method based on a novel classification metric implemented as the upper and lower approximation concepts of rough set theory. Decision rules are produced from "reducts", which are subsets of attributes searched by the proposed method.

N. Zhong et al. [19] illustrate two soft induction techniques for discovering classification rules from uncertain and incomplete databases. They aim at developing an "agent-oriented" and "knowledge-oriented" hybrid intelligent model including deduction, induction and symbolic reasoning techniques to discover knowledge from databases.

J. Stefanowski and D. Vanderpooten [20] demonstrate a widely used experiment study on comparing two different inductive learning approaches, rough-set-based LEM2 and their proposed algorithm Explore, using a number of well-known databases in classification and knowledge discovery-oriented viewpoints.

X. Hu and N. Cercone [21] present a method to learn maximal generalized decision rules from databases by integrating discretization, generalization and rough set feature selection. This method reduces the data horizontally and vertically. Thus, the experimental results demonstrate that their method can dramatically reduce the feature space and improve learning accuracy.

J. Yang [22] designs a hybrid model (AMASS), which integrates a rule induction (RI) approach, a rule-based knowledge system, and a database management system, to perform knowledge mining for rule-based knowledge systems. The author also describes an automated retaining wall selection system, in which rules are induced from previous retaining wall cases and automatically applied to generate feasible solutions to new retaining wall problems.

Y. Leung and D. Li [23] develop the rough-set-based methodology for rule acquisition in an incomplete system. The concept of a maximal consistent block has been used to formulate new definition of approximations to an object set and construct discernibility functions. By adopting the technique of maximal consistent blocks, a variety of discernibility functions in incomplete systems becomes simpler. Their approach is useful for reducing the number of conjunction terms. Y. Leung and D. Li [23] also report that it makes the absorption law more efficient.

J. Zhang et al. [24] propose a novel heuristic algorithm based on rough set theory to find out the feature subset. This algorithm employs appearing frequency of attributes as heuristic information to speed selection process. A simple but effective approach to eliminate redundant attributes from attribute subsets is also proposed. Experiments show in most times it can find minimal reduct.

In [25] S. Tsumoto introduces a new approach to generate diagnostic rules using rough sets and diagnostic model. In this method, the characteristics of experts' rules are closely examined, whose empirical results suggest that grouping of diseases are very important to realize automated acquisition of medical knowledge from clinical databases. The proposed method was evaluated on three medical databases, the

experimental results of which show that induced rules correctly represent experts' decision processes.

A. E. Hassanien and J. M. H. Ali [26] present a rough set method for generating classification rules from the breast cancer data. The attributes are selected, normalized and the dependency rules are generated directly from the real value attribute vector. Then the reduction technique is applied to find all reducts of the data, which contains the minimal subset of attributes that are associated with a class label for classification. Experimental results show that the rough set approach is a useful tool for inductive learning and a valuable aid for building expert systems.

As we mentioned above, there are advantages in both rough set and fuzzy set theory. Several researchers have studied various ways of combining the two technologies [27]. The motivation to include fuzzy information into the rough set framework is to enable users to specify approximate descriptions of queries. There are two kinds of combination: rough fuzzy sets and fuzzy rough sets. By using an equivalent relation on universe U , one can introduce lower and upper approximations in fuzzy set theory. This method is called rough fuzzy sets [28]. Alternatively, a fuzzy similarity relation can be used to replace an equivalent relation. The result is a deviation of rough set theory called fuzzy rough sets [29]. Fuzzy rough sets encapsulate the related but distinct concepts of vagueness for fuzzy sets and indiscernibility for rough sets.

From the literature review above, we briefly divide the rough set-based approaches into three groups based on its main functions:

- Data reduction by dependency analysis.
- Classification by set classifiers.

Rule induction by reduced reducts or set classifiers.

In rough set method, unnecessary attributes are removed by analyzing dependency of condition attributes. Objects belonging to some patterns are clustered by lower and upper approximations. There are three typical rough set-based rule reduction models.

- The first model is developed by Z.Pawlak [30], which starts from reducing the redundant condition attributes and creating all possible reducts, then generates certain rules using rule generation algorithm from a minimal set of reducts.
- The second model uses an induction engine to induce rules from lower and upper approximations [4]. The induction engine is based on decision-tree method, knowledge acquisition techniques or machine learning algorithms.
- The third model applies a feature reduction model to produce several smaller sets of rules from a rule-reduct sets generated by a reduct generation algorithm to help decision-maker predict outcomes when face a new case [31].

2.3 Soft Computing Framework

Soft computing uses hybridizations of methodologies for rule induction [32]. Such methodologies include all soft computation tools in rule induction area: fuzzy sets, neural networks, genetic algorithms, and rough sets. The development of integrated technologies has shown some advancement because no single technique is applicable

to the many sub-problems, which a given application may have. There is also a motivation for creating hybrids that can exhibit multiple information processing capabilities within one architecture.

L. X. Wang and J. M. Mendel [33] develop a method to create a fuzzy rule-base, which is a combination of rules generated from numerical examples and linguistic rules supplied by human experts. The input and output domain spaces are divided into a number of linguistic subspaces. Human intervention is sought to assign degrees to the rules and conflicts are resolved by selecting those rules yielding the maximum of a computed measure corresponding to each linguistic subspace.

R. Rovatti and R. Guerrieri [34] attempt to define the correct structure of rules of a fuzzy system when the input and output behavior is sampled at random points. The assumption that a rule can either be included or excluded from the rule set is relaxed, and degrees of membership are exploited to achieve good approximation results. Defuzzification methodologies are then used to extract well-behaving crisp rule sets. Symbolic minimization is carried out to obtain a compact structure that captures the high-level characteristics of the target behavior.

H. Ishibuchi *et al.* [35], on the other hand, use interval vectors to represent fuzzy input and output in an MLP. A backpropagation algorithm is applied on a cost function defined by α -level sets of actual and target fuzzy outputs, using the principles of interval arithmetic. Different fuzzy *if-then* rules are interpolated from a few sample rules.

L. Wang and J. Yen [36] have designed a hybrid algorithm using GA's for extracting fuzzy rules from a given rule-base to construct a parsimonious fuzzy model with high generalization ability. The parameters of the model are estimated using the Kalman filter.

Many studies have been done on the integration of neural networks and rough sets. These two technologies can complement each other to make more powerful systems.

P. Mitra and S. K. Pal [37] design a hybrid decision support system for detecting the different stages of cervical cancer. This system includes the evolution of knowledge-based subnetwork modules with genetic algorithms using rough set theory and ID3 algorithm. The performance of this methodology with modular network is superior in terms of classification score, training time, and network sparseness. Therefore, extracting rules become easier to practitioners.

A method of gradually finding the optimal number of rules by rough sets in a neuro-fuzzy model has been proposed in [38] by Y. Jung-heurn et al. The final set of rules is directly selected by iteratively removing rules in such a way that the overall behavior of the neuro-fuzzy system is preserved.

A hybrid intelligent system, which is proposed by B. S. Ahn, S. S. Cho, and C. Y. Kim [39], combines rough set approach with neural network to predict the failure of firms based on the past financial performance data. When a new object is predicted by rule set, it is fed into the neural network if it does not match any of the rules. Thus, they use rough sets as a preprocessing tool for neural networks. Though this approach

can achieve high classification accuracy, some knowledge in neural networks is still hidden and not comprehensible for user.

R. Yasdi [40] uses rough sets for the design of knowledge-based networks in the rough-neuro framework. This methodology consists of generating rules from training examples by using rough set concepts and mapping them into a single layer of connection weights of a four-layered neural network. In this rough-neuro framework, rough sets are used to speed up or simplify the process of using neural network.

R. W. Swiniarski and L. Hargis [41] use rough sets as a front end of backpropagation classifier. In [42] rough sets are employed to reconstruct the input variables and improve the output and performance of the neural network. An approach of unsupervised rough set clustering [43] is applied for grouping web users.

S. K. Pal et al. [44] propose a rough-fuzzy multi-layer perceptron with modular concept using a genetic algorithm to obtain a structured network suitable for both classification and rule extraction. Rough sets in this model are mainly used for generating the network parameters and roughness at the neuronal level has been incorporated.

Thus, there has been an explosive growth in the successful use of hybrid intelligent systems. However, the architecture and representation of hybrid systems may not be very efficient when the hybrid systems are related to the computational efficiency, accuracy, problem formulation and knowledge representation for particular task. Neural networks are often used as modules within intelligent hybrid systems for they are able to perform tasks, which are difficult to intractable for a rule-

based system. However, rough sets are able to interpret the imbedded knowledge to symbolic rules. This motivates the integration of neural networks and rough sets to provide a good solution to rule generation problems.

2.4 Summary

In this chapter, we have reviewed data mining process and one important topic in it: rule induction. We have also covered a comprehensive range of rule induction techniques, which includes decision tree, association rules, fuzzy sets, neural networks and rough sets. We have briefly given the analysis of advantages and drawbacks of these techniques when applying them in rule induction. Next, we have introduced concepts and methods of rough set theory. Finally, we have explored the development of soft computing framework in many diverse areas.

In Chapter 3, we shall introduce a new rule extraction framework using rough sets and neural networks.

Chapter 3 Rule Extraction

Framework

3.1 Rule Extraction Framework

We have explored several hybrid frameworks for rule induction in Chapter 2. Most of the previous works use rough sets as a preprocessing tool of neural classifier. As rough sets provide useful technology to reduce redundant attributes from information systems, they are mostly applied as a tool as feature selector. However, the accuracy is low because the classification region defined by rough sets is relatively simple. In order to solve this problem, we introduce SOM network into the system to improve selection of attributes.

The SOM network has been applied to various problems as a clustering tool such as image recognition, medical diagnosis and market segmentation [45]. The SOM network is good at transforming multidimensional inputs into a map of fewer dimensions. In addition, the training time for SOM is much less comparing with other

neural networks. However, drawbacks of SOM network is that it cannot explain the classification results specifically. Thus, it is natural to combine SOM network and rough sets for their complementary features.

Our proposed rule extraction framework is described in Figure 3.1. There are mainly four steps: data preprocessing, attribute selection, attribute reduction and rule generation. We will give detailed description of our framework in the following sections.

Figure 3.1 The Architecture of Rule Extraction Framework.

3.2 Discretization

The motivation for this phase is driven by the fact that continuous attributes are proven to be unsuitable for the extraction of concise symbolic rules. Therefore, it is necessary to discretize continuous attributes to discrete intervals. The domain of values of an attribute is reduced to a small number of attribute-value ranges, where a

label can represent each interval. Data discretization can be treated as a preprocessing of rule induction.

It is useful to develop appropriate techniques to discretize numerical data. There are mainly two kinds of data discretization methods including unsupervised discretization and supervised discretization.

In unsupervised discretization, class variable is not used. Method of equal width interval binning [46] divides feature values into N intervals of equal size between minimum and maximum values. For example, if L and H represent the lowest and highest value of a feature respectively, the intervals will have equal width $w = (H - L) / N$. However, this method does not make use of instance labels in the discretization process.

On the other hand, supervised discretization uses the information about classes of objects. 1R [47] is a supervised discretization method using interval binning. 1R divides the range of continuous values into a number of disjoint intervals and adjusts the boundaries based on the class labels associated with the continuous values. Each interval should contain a minimum of six instances. The adjustment at the boundary does not allow terminating an interval if the next instance has the same class label as the majority class label.

We make discretization on continuous data based on entropy heuristic approach proposed by U. M. Fayyad and K. B. Irani [48]. This supervised algorithm uses the class information entropy of candidate partitions to select bin boundaries for discretization. We are given a set of instances S , a feature A , and a partition boundary

T. T partition a set of instances S into the subsets S_1 and S_2 . Let there be k classes $C_1, C_2, \dots, C_k$ and $P(C_i, S_j)$ be the proportion of instances in S_j that have class i . The class entropy of the subset S_j is defined as:

$$Ent(S_j) = -\sum_{i=1}^k P(C_i, S_j) \log(P(C_i, S_j)) \quad (3.1)$$

The class information entropy of the partition induced by T, denoted $E(A, T; S)$ is given by:

$$E(A, T : S) = \frac{|S_1|}{|S|} Ent(S_1) + \frac{|S_2|}{|S|} Ent(S_2) \quad (3.2)$$

Where $|S_k|$ is the number of instances in S_k , $Ent(S_j)$ is the class entropy of the subset S_j .

For a given feature A, the boundary $T_{\min}$, which can minimize the entropy function over all possible partition boundaries, is selected as a binary discretization boundary. The discretization step may have a noticeable impact on the performance of the induced rules.

3.3 Attribute Selection

3.3.1 Data Clustering with SOM

In rule generation, the objective is to find the most important attributes for decisions. It is shown that some condition attributes are indispensable and the

classification ability of the decision table will be changed if they are deleted. However, some condition attributes have minor effect to decisions. If they are deleted, the classification ability of the decision table will not be influenced.

In our framework, the SOM network serves as a clustering tool that can group condition attributes into appropriate categories. It does not require the knowledge of the corresponding output for comparison and learning. Therefore, it is very suitable for presenting the relationships in the attribute space with a sufficient accuracy. The SOM presents some important advantages in comparison with other neural network models. In addition to the fact that its structure is simple to implement and that the required training time is reasonably short, the SOM provides a direct visualization of the significance of attributes and allows for its unsupervised clustering.

Attributes of noisy data have no classification ability. In addition, redundant attributes do not add anything new to classification. We input the condition attributes of objects to the SOM. During SOM training process, we are able to remove noisy data and find an optimal subset of all attributes.

We use SOM to cluster the attributes of objects so that we can find and remove noisy and redundant attributes. This step has two effects for further rule generation:

- (1) SOM groups the attributes based on decisions.
- (2) It is quite easy to find noisy data from the clustering map of the result.

In this process, the SOM network is composed of input layer and output layer. The SOM network is an unsupervised network, which can recognize a topological map from a random starting point. In SOM network, input nodes and output nodes are

fully connected with each other. Each input node contributes to each output node with a weight. Some parameters such as the learning rate, radius rate, converge error rate can be assigned different numbers. From the decision table, we get the relation between the attributes and objects. Then we use SOM to find the similarity of different attributes for all the objects. Objects with the same decision attributes belong to one cluster, which is effective for future attribute reduction. We will describe the detail of using SOM to reduce the attributes at section 3.3.2.

3.3.2 The SOM Neural Network

The origins of the SOM model are traceable to the study of biological neural networks in both humans and animals, which have revealed that neurons are organized in such a manner that neighboring neurons become sensitive to similar stimuli. The SOM has been successfully applied to various problem domains, including speech recognition, image data compression, custom targeting and medical diagnosis. Besides, the SOM has the ability of operating on incomplete data and providing a tool for performing data-mining tasks and visualizing the relationships existing in the pattern space.

The SOM network [45] consists of a set of neurons organized into a usually one or two dimensional regular structure: an input layer and an output layer, which is also called competition layer. The units in the network are connected to adjacent ones by a neighborhood relation. At first SOM generates a winning node. At the training phase, the winning node and the neighborhood weights are adjusted until the network converges. Hence, there are two important steps in the self-organizing feature map algorithm: (i) selecting neighborhood, and (ii) adaptive process. In the first step of

neighborhood selection, we find the activity neuron. In the adaptive process, we update the weight vectors using Hebbian learning law [45]. We now give the details of our SOM algorithm.

Algorithm 3.1: SOM

1. Find the winner: If we denote input vector as $X = (x_1, x_2, \dots, x_n)^T$, the connecting weight between the neurons in the input layer and those in the output layer is $W_j = (w_{1j}, w_{2j}, \dots, w_{nj})^T$, $j = 1, 2, \dots, h$, and the output competition layer neurons are:

$$y_j = \sum_{i=1}^n w_{ij} x_i = W_j^T X, j = 1, 2, \dots, k \quad (3.3)$$

During the training process, SOM can change the disordered input set into ordered topology connection in the output layer, such as the distribution of clustering centers. Among all the objects in the input set, clustering centers are those connection weights corresponding to the winner unit in the output layer's output. We use the following criterion to find the winner neuron.

$$\text{win}(X) = \arg \min \|X - W_j\|, j = 1, 2, \dots, k \quad (3.4)$$

where $\|*\|$ is the Euclidean norm of the argument vector, $\text{win}(X)$ is the subject of attention and the corresponding neuron win is called the winning neuron for the input vector X . Equation (3.4) describes a sort of mapping, which is a continuous input vector space to the discrete neuron output space by competitive actions among the competing units. Therefore, the self-organizing is the process of looking for connection weights that best match the input vector.

2. Find the neighborhood of the winner: The active neuron is replaced by a topological neighborhood N_c of the node c . The neighborhood N_c depends on the output map structure used in the SOM, and is defined below.
3. This neuron together with its neighbors is updated by the following rule:

$$m_i(t+1) = m_i(t) + h_{ic}(t)|x(t) - m_i(t)| \quad \forall i \in N_c(t) \quad (3.5)$$

where $m_i(t)$ is the weight of neuron i at time instant t . $x(t)$ is the input pattern, $N_c(t)$ is a function denoting the neurons, which belong to the neighborhood of the best matching neuron and $h_{ic}(t)$ is the gain factor. A widely applied neighborhood kernel can be written in terms of the Gaussian function:

$$h_{ci} = \alpha(t) \cdot \exp\left(-\frac{d_{ci}^2}{2\sigma^2(t)}\right) \quad (3.6)$$

where $\alpha(t)$ is the scalar-valued "learning-rate", neighbor distance d has the following definition:

$$d_{ci} = \|r_c - r_i\| \quad (3.7)$$

and the parameter $\alpha(t)$ defines the width of the kernel. Both the neighborhood function $N_c(t)$ and the adaptation gain parameter $\alpha(t)$ decrease as training is processed.

End **Algorithm 3.1**: SOM.

3.4 Attribute Reduction

Attribute reduction is a process of removing those condition attributes, which do not provide any additional information about the object. A relative reduct [2] is a minimal sufficient subset of attributes, which has the same ability to discern concepts as the full set of attributes. A good choice of attribute subset can increase classification accuracy, save the computational time, and simplify rules.

Rough sets, as described in Chapter 2, can be applied to remove the irrelevant or unnecessary attributes from decision tables. To find the minimal subset of attributes for decision table, we propose an optimal reduct algorithm based on binary discernibility matrix. It performs an internal search looking for attribute-value pairs, which have high discernibility degree. Details of this method are given in Chapter 4.

3.5 Rule Generation

A reduced table can be viewed as a rule set, where each rule corresponds to one row of the table. The rule set can be generalized further by applying approximation method of rough sets.

During attribute reduction, if the input data is inconsistent, our framework computes lower and upper approximations of all concepts. Rules induced from the binary discernibility matrix for lower approximation certainly describe the concept. On the other hand, rules induced from the binary discernibility matrix for upper approximation possibly describe the concept. Without losing generality, the rules, whose confidences are less than a threshold, are removed. The threshold can be calculated as the product of the user-specified tolerance level.

The obtained rules give highly redundant set of decision rules, where each object is covered by at least one rule. The simplest reduction of the obtained set of rules is that duplicate rules are eliminated. The advanced rule reduction method employed in our framework is rule simplification. Details of rule generation are given in Chapter 5.

In summary, the outline steps of the algorithmic framework based on the architecture in Figure 3.1 is given below.

Algorithm 3.2: Hybrid Rule Extraction Framework (HybREF)

1. Apply entropy-based discretization (given in section 3.2) to preprocess data.
2. Construct decision table and input these objects into the SOM network. Noisy data are removed through clustering by SOM. We have discussed Algorithm 3.1: SOM in section 3.3.
3. Apply attribute reduction algorithm based on binary discernibility matrix to get the optimal reduct from reduced decision table. We will discuss this algorithm (Algorithm 4.2: AR) in Chapter 4.
4. Apply rule generation algorithm on the obtained reduct to achieve certain and approximate rule sets. We will discuss this algorithm (Algorithm 5.1: RG) in Chapter 5.

End **Algorithm 3.2:** Hybrid Rule Extraction Framework (HybREF)

3.6 Summary

In this chapter, a new rule extraction framework is presented. Most of previous works on hybrid system use rough sets as a preprocessing tool of neural classifier. However, it is not easy to explain the knowledge of neural networks into rules because of the complexity of the structure of them. On the contrary, we make SOM as preprocessing tool to rough sets such that it is much easier and more straightforward

to induce rules from decision tables. Since SOM provides a direct visualization of the significance of attributes and allows for its unsupervised clustering, we make use of it to remove the noisy objects.

Chapter 4 Generalization of Attribute Reducts

4.1 Binary Discernibility Matrix

Binary encoding of discernibility relations and attributes in an information system makes a better utilization of computational memory. The use of bit-wise operation, instead of sets-oriented operation, gives rise to the parallel processing over groups of attributes [49].

A binary discernibility matrix represents the discernibility between pairs of objects in a decision table. Let $T = \langle U, C \cup D, V_q, f_q \rangle_{q \in C \cup D}$ be a decision table, a set of objects (rows in a table) $U = \{e_1, e_2, \dots, e_n\}$, a set of condition attributes $C = \{c_1, c_2, \dots, c_m\}$, a decision attribute $D = \{d\}$.

For each $q \in C \cup D$,

- V_q is a set of attribute value for q .
- Each $f_q : U \rightarrow V_q$ is an information function which assigns particular values from domains of attributes to objects such that $f_q(x_i) \in V_q$ for all $x_i \in U$ and $q \in C \cup D$.

Let M be the binary discernibility matrix. Each row in the matrix corresponds to a pair of objects (e_i, e_j) and the k^{th} bit in the row corresponds to the attribute c_k . The discernibility between e_i and e_j by a single attribute c_k , denoted by $d[(e_i, e_j), c_k]$, is defined as follows:

$$d[(e_i, e_j), c_k] = \begin{cases} 0 & \text{if } [e_i]_{c_k} = [e_j]_{c_k} \\ 1 & \text{otherwise} \end{cases} \quad (4.1)$$

where $1 \leq i < j \leq n$, and $k \in 1, 2, \dots, m$. This representation of discernibility allows it to determine if a pair of objects is discernible by the attribute c_k with a single bit. In the binary discernibility matrix, columns are single condition attributes and rows are objects belonging to different concepts. It can be seen that M has m columns. The maximal number of rows in M is $n(n-1)/2$.

Example. We present sample information system (the decision table) in Table 4.1 with $U = \{e_1, e_2, \dots, e_6\}$, $C = \{c_1, c_2, c_3, c_4\}$, $D = \{d\}$.

Table 4.1 A Sample Information System.

	c_1	c_2	c_3	c_4	d
e_1	1	0	2	1	1
e_2	1	0	2	0	1
e_3	1	2	0	0	2
e_4	1	2	2	1	0
e_5	2	1	0	0	2
e_6	2	1	1	0	2

The binary discernibility matrix for $\{d=2\}$ corresponding to the sample information system in Table 4.1 is presented in Table 4.2.

Table 4.2 The Binary Discernibility Matrix for $\{d=2\}$ Corresponding to the Sample Information System in Table 4.1.

	c_1	c_2	c_3	c_4
e_3e_1	0	1	1	1
e_3e_2	0	1	1	0
e_3e_4	0	0	1	1
e_5e_1	1	1	1	1
e_5e_2	1	1	1	0
e_5e_4	1	1	1	1
e_6e_1	1	1	1	1
e_6e_2	1	1	1	0
e_6e_4	1	1	1	1

Characteristics of the matrix M are as follows:

- (1) If a row is all 0's, it means that the corresponding pair of objects are indiscernible even when using the whole set of attributes. The objects are inconsistent as they have the same condition attributes while belonging to

different decisions. In this situation, the lower and upper approximation methods should be used to solve the problem.

- (2) If a row in the matrix has only one "1", the corresponding attribute is the only one which is able to discern that pair of objects. Thus, this attribute is indispensable to the decision $D = \{d\}$.
- (3) If a column is all 1's, the condition attribute is able to discern all the pairs of objects. In such case, this attribute is satisfied with the condition of a minimal covering of the decision table.
- (4) If a column is all 0's, the condition attribute is completely irrelevant to the concept. This attribute cannot distinguish any pair of objects. Therefore, it can be removed from the decision table as redundant attribute.

The properties of the binary discernibility matrix are used to pick up attributes for reduct. We observe that attributes with higher discernibility degree contribute more classification power to the reduct. After calculating the discernibility degree of the attributes, we select the attribute with highest discernibility degree.

4.2 Binary Representation of Attributes

We use binary representation method in [49] to define a bit string with length m (the number of condition attributes), where each bit corresponds to a condition attribute. Figure 4.1 shows a format of the bits and attributes. Each bit of the bit string is set to "1" if the corresponding condition attribute is a member of the set; otherwise, it is set to "0". Therefore, the rows for one object in the binary discernibility matrix can be represented in one bit string. In this binary representation, set operations such

as adding or removing objects to the binary discernibility matrix are computed using bit-wise operations.

Figure 4.1 Location of the Condition Attributes in a Binary String.

4.3 Solve the Inconsistency in Decision Table

Some examples have the same value of condition attributes, yet the decision value is different. It is called "inconsistency". According to the properties of the binary discernibility matrix, we define the difficulty degree as follows:

Let M be a binary discernibility matrix having s rows and t columns. Its element value in the i^{th} row and j^{th} column is a_{ij} .

DEFINITION 1 (Difficulty Degree). The difficulty degree is a measure of the difficulty to discern the corresponding objects.

$$Dif(e_i, e_j) = \frac{1}{t} \sum_{j=1}^t a_{ij}$$

There are two special cases:

- (1) If $Dif(e_i, e_j) = 0$, the object e_i and e_j are inconsistent.

- (2) If $Dif(e_i e_j) = \frac{1}{t}$, there is only one attribute c_k , which is able to distinguish objects. c_k is the indispensable attribute in the decision class.

When we calculate $Dif(e_i e_j) = 0$ in the decision table, the object e_i and e_j are inconsistent. That is, for objects, whose values of any attribute are the same, their decision values are different. In this situation, the lower and upper approximations of the decision class are used to solve this problem.

Initially, we copy all the objects in the binary discernibility matrix M into the binary discernibility matrix $\underline{M}$ and $\overline{M}$ for lower and upper approximations respectively. If we detect object e_i and e_j are inconsistent, which satisfies $e_i, e_j \in U : d[e_i] = d, d[e_j] \neq d$ through calculating the difficulty degree $Dif(e_i e_j)$, e_i will be removed from $\underline{M}$ and e_j will be added into $\overline{M}$. After all, the inconsistent objects are found and classified into $\underline{M}$ and $\overline{M}$ respectively. Therefore, the process to solve inconsistency of data is also the process of constructing discernibility matrix $\underline{M}$ and $\overline{M}$ for the lower and upper approximations.

Both lower and upper approximations are modified depending on the results of the discernibility relations and completeness. The algorithm for solving inconsistency is based on approach in [49]. The methods for defining inconsistency of our approach are different with [49]. We use the calculation of $Dif(e_i e_j)$ to decide whether objects e_i and e_j are inconsistent or not. However, all the objects are compared with their condition attributes and decision attribute to find the inconsistency in method of [49].

Algorithm 4.1 SolveInconsistency (SI)

Input: A decision table $T = \langle U, C \cup \{d\}, V, f \rangle$, $U = \{e_1, e_2, \dots, e_n\}$,
 $C = \{c_1, c_2, \dots, c_m\}$

Output: Binary discernibility matrix $\underline{M}$ and $\overline{M}$ for Lower and upper approximations of decision class d

1. Set lower and upper approximations of decision class d as $\underline{M}$ and $\overline{M}$;
2. Construct binary discernibility matrix M for the decision table T .
3. Copy all the objects $e \in U : d[e] = d$ in M to $\underline{M}$ and $\overline{M}$ respectively;
4. Inconsistency_mark=False;
5. For objects e_i and e_j which satisfy $e_i, e_j \in U : d[e_i] = d, d[e_j] \neq d$, do {
6. Calculate the difficulty degree $Dif(e_i, e_j)$ for objects e_i and e_j in the binary discernibility matrix M .
7. if $Dif(e_i, e_j) = 0$ do{
8. Delete e_i from $\underline{M}$;
9. Add e_j to $\overline{M}$;
10. Inconsistency_mark=True; }
11. }
12. Return Binary discernibility matrix $\underline{M}$ and $\overline{M}$ for lower and upper approximations of decision class d .

End **Algorithm 4.1** SolveInconsistency (SI).

4.3.1 Explanation of Algorithm 4.1

The first step is to set lower approximation $\underline{M}$ and upper approximation $\overline{M}$ based on the decision d . To initialize $\underline{M}$ and $\overline{M}$, we copy all the objects in the binary discernibility matrix M into $\underline{M}$ and $\overline{M}$.

Subsequently, we find the inconsistent objects in the binary discernibility matrix M . The difficulty degree $Dif(e_i, e_j)$ is used to conclude whether the objects e_i and e_j are inconsistent or not. According to the properties of the binary discernibility matrix,

which is described in section 4.1, if $Dif(e_i, e_j) = 0$, e_i and e_j are inconsistent. In this situation, e_i will be removed from the lower approximation M . On the contrary, object e_j will be added into the upper approximation $\overline{M}$.

We shall further illustrate the steps in Algorithm 4.1 by an example in section 4.5.

4.3.2 A Note on the Number of Operations

Due to computation needed in steps 5-10 of Algorithm 4.1, the order of the number of operations needed for calculating the difficulty degree $Dif(e_i, e_j)$ for objects e_i and e_j is $O(n(n-1)/2)$. For constructing the lower and upper approximations, the order of the number of operations is $O(n)$.

4.4 Attribute Reduction by Rough Sets

In order to obtain the minimal rules, it is necessary to find the minimal reduct of attributes. The attribute reduct must cover the entire decision table. That is, the reduced decision table has the same classification ability as the original one. Therefore, the attributes, which are indispensable to discern any pair of objects even if there is only one pair of objects, are put into the attribute reduct firstly.

As we have mentioned in section 4.1, the properties of the binary discernibility matrix are useful to discern objects. Hence, the central idea of our attribute reduction algorithm is based on these properties. The column of the binary discernibility matrix with the most element "1"s is found because the attribute corresponding to the

column has the highest discernibility degree. These attributes are included into core of decision table.

According to the properties of the binary discernibility matrix, we define the discernibility degree as follows:

Let M be a binary discernibility matrix having s rows and t columns. Its element value in the i^{th} row and j^{th} column is a_{ij} .

DEFINITION 2 (Discernibility Degree). The discernibility degree is a measure of the classification ability of the attribute in the decision class.

$$Deg(c_k) = \frac{1}{s} \sum_{i=1}^s a_{ij} \quad (4.3)$$

The rate of 1's in a column over the number of pairs in the discernibility matrix represents the approximation of the condition attribute to a reduct. There are two special cases as follows:

(1) If $Deg(c_k) = 0$, c_k is completely irrelevant in the decision class.

(2) If $Deg(c_k) = \frac{n(n-1)}{2}$, c_k satisfies the reduct condition.

Finding all the reducts is a NP-complete problem [50]. Fortunately, it is usually not necessary to find all the reducts in practical applications including ours. In general, there is more than one reduct for a decision table.

Algorithm 4.2. Attribute Reduction (AR).

Input: A decision table $T = \langle U, C \cup \{d\}, V, f \rangle$, $U = \{e_1, e_2, \dots, e_n\}$,
 $C = \{c_1, c_2, \dots, c_m\}$

Output: Reducts of lower and upper approximations of the binary discernibility matrix $\underline{M}$ and $\overline{M}$

1. Use Algorithm 4.1: SolveInconsistency (SI) and get the binary discernibility matrix $\underline{M}$ and $\overline{M}$ for the lower and upper approximations of the decision class (given in Algorithm 4.1)
2. while ($\underline{M} \neq \phi$) { /*Compute the discernibility degree of attribute c_k */
3. Calculate $Deg(c_k)$ of each attribute in $\underline{M}$;
4. Set $Reduct = \phi$; /*Remove the irrelevant attribute in the first*/
5. if ($Deg(c_k) = 0$) {
6. Remove the c_k column from $\underline{M}$; }
7. if ($Deg(c_k) = \frac{n(n-1)}{2}$) {
8. Return Reduct of $\underline{M}$;
9. Break;
10. }
11. if (c_i is the attribute with highest discernibility degree) do
12. {
13. $Reduct \leftarrow Reduct \cup \{c_i\}$;
14. Remove the rows which have “1” in the c_i column from $\underline{M}$;
15. Remove the c_i column from $\underline{M}$; }
16. }
17. End While;
18. Return Reduct of $\underline{M}$.
19. Do the same process on attribute reduction on $\overline{M}$ and return the corresponding Reduct of $\overline{M}$.

End **Algorithm 4.2.** Attribute Reduction (AR).

Based on the binary discernibility matrix, we propose the attribute reduction algorithm of finding reducts for a decision table. Algorithm 4.2 is based on approach of using binary discernibility matrix in [49]. There are two methods to search for attribute reduct in [49]: direct search and Boolean conversion. However, our

algorithm computes the discernibility degree of all attributes and ranks them based on the discernibility degree. The discernibility degree indicates the attribute's effect on class distinguishability. So the search for attribute reduct in our approach is better than direct search.

4.4.1 Explanation of Algorithm 4.2

Input of our attribute reduction algorithm is the decision table T . Considering there are "inconsistency" among objects, step 1, which uses Algorithm 4.1 in section 4.3, is to solve inconsistency and to construct the binary discernibility matrix M and $\overline{M}$ for the lower and upper approximations.

Next, we generate attribute reduct from binary discernibility matrix M of lower approximation. We calculate discernibility degree of each attribute. From our previous analysis of discernibility degree, if the discernibility degree is 0, it shows the attribute is irrelevant to the decision. Therefore, we remove this attribute from discernibility matrix M of lower approximation. On the contrary, if the discernibility degree is equal to $n(n-1)/2$, the attribute is satisfied with reduct condition. In this situation, we break the loop and return the obtained reduct. After that, steps 11-15 are a process of selecting an attribute with the highest discernibility degree. A reduct consists of these attributes. The algorithm terminates when the binary discernibility matrix M for lower approximation is empty.

At last, we use the same method to get attribute reduct of binary discernibility matrix $\overline{M}$ for upper approximation.

We shall further illustrate the steps in Algorithm 4.2 by an example in section 4.5.

4.4.2 A Note on the Number of Operations

The number of operations needed by the algorithm cannot be determined exactly since it is highly dependent on the nature of the input data. Suppose n is the number of objects in the decision table T . m is the number of attributes. To compute an attribute reduct, it needs $O(mn+m\log m)$ operations in the worst case because computing the degree of discernibility for m attributes needs $O(mn)$ operations. Computing for the attribute with highest discernibility degree needs $O(m\log m)$ operations. To obtain the attribute reduct, we need $O(mn)$ operations.

4.5 Example

Table 4.3 An Example for Information System.

ID	Diploma	Experience	French	Reference	Decision
1	MBA	Medium	Yes	Excellent	Accept
2	MBA	Medium	Yes	Neutral	Reject
3	MSc	Low	Yes	Good	Reject
4	PhD	High	Yes	Neutral	Accept
5	PhD	Medium	Yes	Neutral	Accept
6	PhD	High	Yes	Neutral	Reject
7	MBA	High	No	Good	Accept
8	MSc	Low	No	Excellent	Reject

In this section, we give a simple example to explain the process of our algorithm for attribute reduction. Other elaborate experiments for comparisons and analysis are given in Chapter 6. Table 4.3 shows a simple information system. There are eight ,

objects in the universe $U = \{e_1, e_2, \dots, e_n\}$. Condition attributes C is composed of $\{c_1 = \text{Diploma}, c_2 = \text{Experience}, c_3 = \text{French}, c_4 = \text{Reference}\}$. Decision attribute $D = \{d = \text{accept or reject}\}$.

Table 4.4 shows approximations for the decision attribute D . Since we found there is inconsistency between e_4 and e_6 , we apply Algorithm 4.1: SI to solve the inconsistency in the decision table. Hence, the binary discernibility matrix of lower and upper approximations for decision "accept" and "reject" are generated respectively.

Table 4.4 Approximations for Decision Attributes.

ID	Binary Decision Attribute		Decision for Accept Approximations		Decision for Reject Approximations	
	Accept ⁺	Reject ⁺	Lower	Upper	Lower	Upper
1	Yes	Yes	Yes	Yes	No	No
2	No	Yes	No	No	Yes	Yes
3	No	No	No	No	Yes	Yes
4	Yes	No	No	Yes	No	Yes
5	No	Yes	No	No	No	No
6	No	Yes	No	Yes	No	Yes
7	No	No	No	No	No	No
8	No	Yes	No	No	Yes	Yes

The binary discernibility matrix of lower and upper approximations for decision "accept" are shown in Table 4.5 and Table 4.6. We calculate discernibility degree for each attribute in the lower approximation using Algorithm 4.2: AR and obtain the reduct for the lower approximation. As a result, $\{c_1 = \text{Diploma}, c_4 = \text{Reference}\}$ and $\{c_2 = \text{Experience}, c_4 = \text{Reference}\}$ are obtained as reducts for the lower approximation based on our attribute reduction algorithm. We use the same method to

obtain the reduct: $\{c_1 = \text{Diploma}, c_2 = \text{Experience}, c_4 = \text{Reference}\}$ for the upper approximation.

Table 4.5 Binary Discernibility Matrix for Decision "Accept" (Lower Approximation).

Pairs M_{ij}	Condition Attributes			
	c_1	c_2	c_3	c_4
e_1e_2	0	0	0	1
e_1e_3	1	1	0	1
e_1e_4	1	1	0	1
e_1e_5	1	0	0	1
e_1e_6	1	1	0	1
e_1e_7	0	1	0	1
e_1e_8	1	1	0	0

Table 4.6 Binary Discernibility Matrix for Decision "Accept" (Upper Approximation).

Pairs M_{ij}	Condition Attributes			
	c_1	c_2	c_3	c_4
e_1e_2	0	0	0	1
e_1e_3	1	1	0	1
e_1e_5	1	0	0	1
e_1e_7	0	1	0	1
e_1e_8	1	1	0	0
e_2e_4	1	0	0	0
e_2e_6	1	0	0	0
e_3e_4	1	1	0	1
e_3e_6	1	1	0	1
e_4e_5	0	1	0	0
e_4e_7	1	0	1	1
e_4e_8	1	1	1	1
e_5e_6	0	1	0	0
e_6e_7	1	0	1	1
e_6e_8	1	1	1	1

Table 4.7 shows the result of discernibility degree for each attribute in the lower approximation of decision "accept" using Algorithm 4.2: AR.

Table 4.7 Discernibility Degree of Condition Attributes.

Discernibility Degree	Condition Attributes			
	c_1	c_2	c_3	c_4
	5	5	0	6

4.6 Summary

In this chapter, an algorithm for attribute reduction is proposed. Our method is based on binary discernibility matrix. Therefore, we propose the algorithm to solve inconsistency in the decision table and construct the binary discernibility matrix for lower and upper approximations. In the next step, an optimal subset of attributes is selected by calculating the discernibility degree of each attribute in the discernibility matrix. In addition, discussions on the order of operations needed for the two algorithms are given. Finally, we give a simple example to illustrate the steps of our algorithm for attribute reduction.

Chapter 5 Rule Generation

5.1 Decision Rules

In general, computing decision rules is especially important in data mining applications since they can represent the most general patterns existing in the data. Decision rules are used because they are easy for human comprehension and more compact. Usage of decision rules is the prediction of new objects, which are beyond visual observations.

A classification rule is a combination of values of some condition attributes such that the set of all examples matching it is contained in the set with the same decisions. Our aim is to produce rules in the learning process, which are maximally generalized by removing the maximum number of condition attribute values without decreasing classification accuracy. Using a reduced decision table, the rules are found through determining the decision attribute values based on condition attribute values. Therefore, the rules are presented in an "IF condition(s) THEN decision(s)" format. If

the condition(s) in the IF part matches with the given fact(s), the decision(s) in the THEN part will be performed. In addition, a decision rule always consists of a single and relatively independent piece of information.

As we have discussed in Chapter 2, the rough set theory is particularly well suited to deal with inconsistency in the input decision table. If the input decision table is inconsistent, the lower and upper approximations of the decision classes are computed. The decision rules are generated from these approximations. Consequently, two kinds of rules are distinguished as follows:

- (1) Certain rules (also called exact, discriminant, or deterministic)
- (2) Approximate rules (also called possible, or non-deterministic)

For each decision class, certain decision rules are generated from objects belonging to its lower approximation. Approximate decision rules are generated from the upper approximation. The certain rules indicate unique decision to be made while approximate rules indicate a few possible decisions.

5.2 Algorithm for Rule Generation

A reduced table can be seen as a rule set where each rule corresponds to one object of the table. Such rules can be generalized further by dropping condition attributes. We obtain the lower and upper approximations for these attributes. The rule induction process is performed on the reducts of lower and upper approximations. The three objects of the algorithm of rule generation are:

- (1) Generate strong, non-redundant classification rules. Non-redundant means that it contains the minimum number of condition attributes.
- (2) Analyze patterns by generating rules that can be inspected.
- (3) Predict from new test cases based on the matching of rules to the test cases and produce the most likely outcome.

Figure 5.1 Flowchart of Rule Generation Algorithm.

We use the rough membership function, which is called confidence function to compute the confidence of the rules. The confidence function is defined as:

$$Conf = \frac{Num(e_r)}{Num(e_n)} \quad (5.1)$$

where $Num(e_r)$ denotes the number of object e , which satisfies with rule r , $Num(e_n)$ denotes the number of total object e .

We propose a new rule generation method for our rule extraction framework. The flowchart of the rule generation processing is described in Figure 5.1. As a whole, there are two steps for rule generation: rule induction and rule simplification. Rule induction is to induce two kinds of rules from the two reduced decision tables respectively since we have obtained reducts for lower and upper approximations after processing of attribute reduction. Rules generated from reduced decision table, which is based on reduct, for lower approximation belong to certain rules. However, rules generated from reduced decision table for upper approximation need to be filtered those rules with low confidence. After that, redundant rules are removed through rule simplification algorithm. The efficiency of the rule generation method is proved through experiments in Chapter 6.

Algorithm 5.1 Rule Generation (RG).

Input: Reduced decision table of the lower and upper approximations T_1 and T_2

Output: Rule Sets R

1. Set threshold;
2. Generate rule r_i from T_1 , which denoted as certain rules, $R \leftarrow R \cup \{r_i\}$;
3. Generate rule r_i from T_2 , and calculate confidence based on (5.1);
4. If (confidence $\geq$ threshold) {
5. $R \leftarrow R \cup \{r_i\}$;
6. else {
7. filter this rule r_i ;
8. }
9. Apply rule simplification algorithm (given in section 5.3) to obtain the generated rules.

End **Algorithm 5.1** Rule Generation (RG).

5.2.1 Explanation of Algorithm 5.1

The algorithm for rule induction is described in Algorithm 5.1. Input of the algorithm is reduced decision table of the lower and upper approximations $\mathbf{I}_1$ and T_2 . The threshold value is set to filter the upper approximation rules, which are unsatisfying by users with lower explainable ability. First, we initialize the threshold. Next, we generate rules from the reduced decision table $\mathbf{I}_1$ for lower approximation. Each row in reduced decision table T_1 represents one rule. We only need to combine the condition attributes in the row together. These rules are denoted as certain rules. In step 3, we obtain rules from the reduced decision table T_2 for upper approximation in the same way as obtaining rules from $\mathbf{I}_1$. The confidence for each rule is calculated based on (5.1). Subsequently, the rules, whose confidence is lower than the threshold, are removed from rule sets. The obtained rules are denoted as approximate rules.

We further illustrate Algorithm 5.1 in section 5.4 by an example.

5.2.2 A Note on the Number of Operations

Suppose there are n rules with m attributes. The computation of processing certain rules requires $O(mn)$ operations. Processing the approximate rules needs $O(mn + n)$ operations.

5.3 Rule Simplification

Computing general rules is of particular importance in knowledge discovery since they represent general patterns existing in the data. Rules induced from reduced decision table can be generalized further by dropping condition attributes. The process by which the maximum number of condition attribute values are removed without losing essential information is called value reduction [51]. The maximal generalized rules minimize the number of rule conditions and are optimal because their conditions are non-redundant.

Algorithm 5.2 Rule Simplification (RS).

Input: A set of specific decision rules *RULE*

Output: A set of general rules *GRULE*

1. Set *GRULE* to empty;
2. Set the number of rules: $N \leftarrow |RULE|$;
3. For $i = 0$ to $N - 1$ do {
4. Get the i^{th} rule r_i ;
5. Set $M \leftarrow |r_i|$
6. For $j = 0$ to $M - 1$ do {
7. Remove the j^{th} condition attribute a_j in rule r_i ;
8. If (r_i is inconsistent with any rule r_i in *RULE*) {
9. Restore the dropped condition attribute a_j ;
10. }
11. }
12. Remove any rule r_i in *RULE* that is logically included in rule r_i ;
13. If (rule r_i is not logically included in a rule $r' \in GRULE$) {
14. $GRULE \leftarrow r_i \cup GRULE$;
15. }
16. }

End **Algorithm 5.2** Rule Simplification (RS).

5.3.1 Explanation of Algorithm 5.2

We apply simplification rule algorithm in [26]. The rule simplification algorithm initializes general rules GRULE to empty set and copies one rule $r_1 \subseteq RULE$ to rule r . A condition attribute is dropped from rule r when rule r is checked for decision consistency with every rule $r_i \subseteq RULE$. This step is repeated until every condition of the rule has been dropped once. The resulting rule is the generalized rule. Before rule r is added to GRULE, it is checked for rule redundancy. If rule r is logically included in any rule $r_i \subseteq GRULE$, rule r is deleted. If any rules in GRULE are logically included in rule r , these rules are removed from GRULE. After all rules in RULE are processed, GRULE contains a set of general rules.

We further illustrate Algorithm 5.2 in section 5.4 by example.

5.3.2 A Note on the Number of Operations

Suppose there are n rules with m attributes. The processing for dropping condition attributes on rules requires $O(mn \log n)$ operations. Finding the reduplicate rules for n rules is $O(mn)$ operations. Therefore, the rule simplification algorithm needs $O(mn \log n + mn)$ operations.

5.4 Example

The following simple example is used to illustrate our rule generation algorithm. Further experiments for comparison are demonstrated in Chapter 6. We apply our attribute reduction algorithm on datasets in Table 5.1. After that, we obtain the reduct

consisting of attribute c_3 and attribute c_4 for the lower approximation. The threshold is set as 40%. Therefore, five certain rules are extracted from the reduced decision table consisting of c_3 and c_4 for lower approximation by using Algorithm 5.1: RG.

1. IF $c_3=1 \rightarrow d=1$
2. IF $c_3=2 \rightarrow d=2$
3. IF $c_3=2$ and $c_4=3 \rightarrow d=2$
4. IF $c_4=4 \rightarrow d=3$
5. IF $c_3=4$ and $c_4=3 \rightarrow d=3$

The approximate rules for decisions 1, 2 and 3 are extracted from objects in the upper approximation. Reduct of $\{c_1, c_2, c_3, c_4\}$ is obtained for the upper approximation by using Algorithm 4.2: AR. The confidence of each rule is calculated by (5.1). Hence, two approximate rules are easily generated as:

6. IF $c_1=3$ and $c_2=2$ and $c_3=3$ and $c_4=3 \rightarrow d=2$ or 3

The confidence of this rule is 50% (6/12).

7. IF $c_1=3$ and $c_2=1$ and $c_3=3$ and $c_4=3 \rightarrow d=2$ or 3

The confidence of this rule is 16% (4/24). Since the confidence of this rule is lower than threshold 40%, we can remove this rule from rule sets.

It is shown there are redundant rules in the above rules 1-7. In rule 3, c_4 is redundant to the decision " $d=2$ ". After dropping attribute c_4 , we combine rule 2 and rule 3 by applying Algorithm 5.2: RS.

$$2' \text{ IF } c_3=2 \rightarrow d=2$$

The same process is used for combining rule 3 and rule 5:

$$5' \text{ IF } c_3=4 \rightarrow d=3$$

Table 5.1 Information System of the Application.

ID	c_1	c_2	c_3	c_4	d	ID	c_1	c_2	c_3	c_4	d
1	2	3	1	1	1	13	3	2	3	3	2
2	1	2	1	1	1	14	1	2	2	3	2
3	2	4	1	1	1	15	3	2	3	3	2
4	1	3	1	1	1	16	2	2	2	3	2
5	2	2	1	1	1	17	3	2	3	3	3
6	1	1	1	1	1	18	3	1	3	3	3
7	2	3	1	1	1	19	3	2	3	3	3
8	2	4	1	1	1	20	3	3	3	4	3
9	3	2	3	3	2	21	4	3	4	3	3
10	2	1	3	2	2	22	3	1	3	3	3
11	3	1	3	3	2	23	3	1	3	3	3
12	3	2	3	3	2	24	4	2	4	3	3

5.5 Summary

In this chapter, a new rule generation algorithm is presented. Two kinds of rules: certain rules and approximate rules are induced from the reduced decision tables of lower and upper approximations. The algorithm developed is suitable for our HybREF framework. Furthermore, we apply rule simplification algorithm to create a set of simplified rules.

Chapter 6 Experiments and Discussions

6.1 Experiment One

6.1.1 Data Discretization

The experiments are done on a personal computer running windows XP with Pentium4 2.0 GHz processor and 512Mb memory installed.

We use iris data from UCI repository [52] for the first experiment. There are four condition attributes: sepal length, sepal width, petal length, petal width and three decision values: Iris Setosa, Iris Versicolour, and Iris Virginica. The data format is shown in Table 6.1. The number of objects is 150 and each of three classes has 50 objects.

Table 6.1 Sample of Iris Data.

ID	Sepal length	Sepal width	Petal length	Petal width	Decision
1	5.1	3.5	1.4	0.2	Iris setosa
2	4.9	3.0	1.4	0.2	Iris setosa
3	4.7	3.2	1.3	0.2	Iris setosa
4	4.6	3.4	1.4	0.3	Iris setosa
5	7.0	3.2	4.7	1.4	Iris Versicolour
6	4.9	2.4	3.3	1	Iris Versicolour
7	6.1	2.9	4.7	1.4	Iris Versicolour
8	6.9	3.1	5.4	2.1	Iris Virginica
9	6.4	3.1	5.5	1.8	Iris Virginica
10	7.1	3.0	5.9	2.1	Iris Virginica
.....					
150 samples of Iris data					

Since the four attribute values of iris datasets are continuous, the first step of our framework is to discretize the attribute values into interval domains. We now illustrate the step 1 of Algorithm 3.2: HybREF framework using the above data. The result of applying step 1 of Algorithm 3.2 is given in Table 6.2.

Table 6.2 Result of Discretization on Iris Data.

Sepal length	[4.25,5.15]	(5.15,6.05]	(6.05,6.85]	(6.85,7.90]
Coding	0	1	2	3
Sepal width	[1.90,2.65]	(2.65,3.25]	(3.25,3.75]	(3.75,4.40]
Coding	0	1	2	3
Petal length	[1.00, 2.00]	(2.00,4.93]	(4.93,5.65]	(5.65,7.00]
Coding	0	1	2	3
Petal width	[0.10,0.60]	(0.60,1.70]	(1.70,2.05]	(2.05,2.50]
Coding	0	1	2	3
Classes	Iris-setosa	Iris-versicolor	Iris-virginica	
Coding	0	1	2	

6.1.2 Attribute Selection with SOM

After data discretization, we use the SOM Toolbox for the Matlab environment to simulate the SOM. There are two learning algorithms for SOM [45]. Equation (3.5) is called the sequential (also on-line, stochastic or incremental) learning algorithm since reference vectors are updated after a single input vector is presented. Another algorithm is batch learning algorithm. In the batch algorithm, the update is done using all the input vectors. The batch algorithm is much faster than the sequential algorithm. Furthermore, the batch algorithm does not suffer from convergence problems due to the learning rate. However, the sequential algorithm is of stochastic nature. Hence, it is less likely trapped to a local minimum than the batch algorithm. Therefore, we choose the sequential learning algorithm to train the SOM.

Distance measure for computing the difference between data and reference vectors is another importance factor. Choice of distance measure is depending on the data and purpose of the experiment. In our application, we choose the common used distance measurement – Euclidian distance in equation (3.7).

To initialize the reference vectors, we have two methods: random initialization and linear initialization. The reference vectors are initialized using principal component analysis in linear initialization. It has been shown to be a better choice than random initialization [45].

The neighborhood function plays central role in the SOM algorithm regardless of the type of the learning algorithm. There are three frequently used neighborhood functions for the SOM: Gaussian; Rectangular and Cut Gaussian [45]. Rectangular is

the simplest neighborhood function and all the nodes belonging to the neighborhood set are updated with same intensity. Gaussian function is a very popular choice for neighborhood function because it is biologically more suitable than the rectangular one. Therefore, we choose the Gaussian function as the neighborhood function for our application.

The SOM discrimination accuracy is determined by assigning a class label to each node via a majority-voting scheme. More precisely, if the total number of patterns in the j^{th} class is O_j and the number of patterns of class j assigned to the i^{th} node is $O_{j,i}$, then the node is considered to represent class c_i :

$$c_i = \max_j \frac{O_{j,i}}{O_j} \quad (6.1)$$

Thus, the node represents the class with the highest relative frequency of occurrence of patterns, normalized over the total number of patterns of that class encountered by the network. The normalization permits the use of this metric for datasets where the number of patterns per class varies, allowing nodes to represent both more and less frequent classes.

We set the map size as $6*7$, and input 150 object data to SOM. After batch learning, these objects are grouped in three clusters. The result is shown in Figure 6.1. The label on the node represents the category of the object. For example, "Setosa(1)" means there are one object which belongs to "Decision = Setosa" represented by this node. And "Setosa(7)" means there are 7 objects which belong to "Decision = Setosa" represented by this node. If there are two labels on one node, we will remove

all the objects on the label because rules generated from these objects have lower accuracy. For example, there are five "Versicolor" and two "Virginica" on one node. We can not certainly tell objects on this node, which belong to "Versicolor" or "Virginica". In this situation, we regard these data as "noisy data" and remove these object data.

Figure 6.1 Result of Iris Data by SOM.

From Figure 6.1, we can see that objects which belong to "decision = Setosa" are clearly separated from objects which belong to "decision = Versicolor". However, objects who belong to "decision = Versicolor" have some intersection with "decision = Virginica". Consequently, 34 noisy data are removed from decision table. As a result of applying step 2 of Algorithm 3.2: HybREF framework, we obtain decision

table, which is listed in Table 6.3. This table forms the input for step 3 of Algorithm 3.2: HybREF framework.

Table 6.3 Decision Table for Iris Data.

ID	Sepal length	Sepal width	Petal length	Petal width	Decision
1	0	2	0	0	0
2	1	1	0	0	0
3	0	1	0	0	0
4	0	2	0	0	0
5	0	3	0	0	0
6	3	2	0	0	0
7	2	3	0	0	0
8	1	2	0	0	0
9	2	2	0	0	0
10	3	3	0	0	0
.....					
51	0	1	0	0	1
52	3	1	2	1	1
53	0	0	2	1	1
54	2	1	2	1	1
55	2	1	1	1	1
56	3	1	1	1	1
57	3	1	1	1	1
58	2	0	1	1	1
59	3	0	2	1	1
60	0	0	2	1	1
61	1	0	2	1	1
.....					
101	3	1	2	3	2
102	2	1	2	2	2
103	3	1	3	3	2
104	1	1	2	2	2
105	1	1	2	3	2
106	1	1	2	3	2
107	1	1	2	3	2
108	2	1	2	3	2
109	2	1	3	3	2
110	2	1	3	2	2
.....					

6.1.3 Attribute Reduction by Rough Sets

In this phase, we apply our attribute reduction algorithm on the obtained decision table. From Table 6.4, we calculate discernibility degree (Deg) of condition attributes to the decision "Iris-setosa" based on step 3 of Algorithm 3.2: HybREF framework (which is also Algorithm 4.2: Attribute Reduction-AR). Reduct of this decision table is composed of attribute "petal length" and "petal width".

Table 6.4 Discernibility Degree of Attributes to the Decision "Iris-setosa".

	Sepal length	Sepal width	Petal length	Petal width
Discernibility Degree	8	12	20	18

For "decision=Iris-versicolor", we calculate the difficulty degree $Dif(e_i, e_j)$ for objects e_i and e_j in decision table. Then we apply Algorithm 4.1: SI to solve inconsistency. Table 6.5 and Table 6.6 show binary discernibility matrix of lower and upper approximations for "Iris-versicolor".

Table 6.5 Binary Discernibility Matrix for "Iris-versicolor" (Lower Approximation).

Pairs M_{ij}	Condition Attributes			
	Sepal length	Sepal width	Petal length	Petal width
e_1e_3	1	0	0	0
e_1e_4	1	0	1	1
e_1e_5	0	0	1	1
e_1e_6	0	0	1	1
e_1e_7	1	1	1	1
e_1e_8	0	1	0	0
e_1e_9	1	1	0	1
e_1e_{10}	1	0	0	1
e_1e_{11}	1	1	1	0
.....				

Table 6.6 Binary Discernibility Matrix for "Iris-versicolor" (Upper Approximation).

Pairs M_{ij}	Condition Attributes			
	Sepal length	Sepal width	Petal length	Petal width
e_1e_3	1	0	0	0
e_1e_4	1	0	1	1
e_1e_5	0	0	1	1
e_1e_6	0	0	1	1
e_1e_8	0	1	0	1
e_1e_9	0	1	0	1
e_1e_{10}	1	1	0	1
e_1e_{12}	1	1	0	0
.....				

Table 6.7 and Table 6.8 show the discernibility degree of each attribute to decision "class = Iris-versicolor" in lower approximations and upper approximations respectively. After applying step 3 of Algorithm 3.2: HybREF framework (which is also Algorithm 4.2: Attribute Reduction-AR), (petal length, petal width) is obtained as reduct for decision "class = Iris-versicolor".

Table 6.7 Discernibility Degree of Attributes to "Iris-versicolor" (Lower Approximation).

	Sepal length	Sepal width	Petal length	Petal width
Discernibility Degree	9	7	24	15

Table 6.8 Discernibility Degree of Attributes to "Iris-versicolor" (Upper Approximation).

	Sepal length	Sepal width	Petal length	Petal width
Discernibility Degree	6	8	28	18

6.1.4 Generation of Rules

In this phase, we apply rule generation algorithm (step 4 of Algorithm 3.2: HybREF framework) on the reduced decision table, which is based on attribute reduct to get rules. The threshold for filtering the upper approximation rules is set to 80%. The rules developed for each category are:

1. If petal length = [1.00,2.00] or petal width = [0.10, 0.60] Then Class = Iris-setosa

The confidence of this rule is 100% (35/35).

2. If petal length = (2.00, 4.93] and petal width = [0.60, 1.70] Then Class = Iris-versicolor

The confidence of this rule is 100% (35/35).

3. If petal length = (4.93, 5.65] or petal width = (1.70,2.05] Then Class = Iris-virginica

The confidence of this rule is 94% (33/35).

Since some condition attributes may be dropped, rules can be simplified. After applying Algorithm 5.2: rule simplification(RS), the first rule is simplified as:

If petal length = [1.00,2.00] Then Class = Iris-setosa

6.2 Experiment Two

The aim of the experiment is to analyze the affection of SOM (given in Algorithm 3.1) to the rule generation. In our framework, we use SOM to remove the noisy data and form decision table based on clusters. In order to show the robustness of our framework, the following measures are taken into account:

- (1) Average accuracy of rules.
- (2) Average number of rules.
- (3) Average number of condition attributes.
- (4) Average number of noises deleted.

We select MONK3 dataset [52] in our experiment. The dataset contains 432 objects and there are six attributes, which are described in Table 6.9.

Table 6.9 Values of Condition Attributes.

Attributes	Value
Head_shape	Round, square, octagon
Body_shape	Round, square, octagon
Is_smiling	Yes, no
Holding	Sword, balloon, flag
Jacket_color	Red, yellow, green, blue
Has_tie	Yes, no

We randomly add 5, 10, 20 noises to the MONK3 datasets. We set an object as a noise by changing its decision label. That is, we label the noisy object as "0" ("1" is monk). Each problem is done 20 times and results are shown in Table 6.10 for the three problems. It can be seen that the rule sets remain relatively stable under the different noisy level conditions. Noisy data can be effectively deleted by the

preprocessing of SOM. Hence, it guarantees that concise and accurate rules are generated.

Table 6.10 Results of Robustness Experiments on MONK3 Dataset.

Item	5 noises	10 noises	20 noises
Average accuracy (%)	97.8	97.7	96.5
Average no. of rules	3.5	4.2	4.0
Average no. of condition attributes	1.4	1.5	1.4
Average no. of noises deleted	4.8	9.4	18.2

6.3 Experiment Three

As we have surveyed in Chapter 2, other intelligent methods such as decision trees and neural networks have been successfully applied to rule induction. In this experiment, we compare Algorithm 3.2: HybREF framework with the standard intelligent method: C4.5 [9] and the Subset algorithm [53].

Table 6.11 Comparison of Rule Induction Methods for Australian Credit Data.

Method	Data size	Feature	Accuracy (%)	No. of rules	Uncovered region (%)
HybREF	690	14	80.7	16	2.3
C4.5			71.1	21	2.9
Subset			68.2	28	3.5

The experiments are carried on five datasets from UCI repository [52]. Table 6.11 to Table 6.15 give the comparison between our HybREF framework, C4.5 [9] and the

subset algorithm [53] with respect to three aspects: the average classification accuracy, the number of induced rules and the uncovered region. The experiments are performed by the following three steps. Firstly, the continuous attributes are discretized by entropy based method. Secondly, noisy data are removed by using SOM network. Thirdly, the datasets after cleaning are applied to attribute reduction and rule generation algorithm (step 3 and 4 in Algorithm 3.2: HybREF framework). These procedures are repeated for 20 times and the average of the results are presented. The results show that our approach outperforms these machine learning methods.

Table 6.12 Comparison of Rule Induction Methods for Breast Cancer (Wisconsin) Data.

Method	Data size	Feature	Accuracy (Y_o)	No. of rules	Uncovered region (%)
HybREF	699	9	88.4	10	1.2
C4.5			84.2	21	2.6
Subset			78.3	14	2.1

Table 6.13 Comparison of Rule Induction Methods for Flags Data.

Method	Data size	Feature	Accuracy (Y_o)	No. of rules	Uncovered region (%)
HybREF	194	30	85.2	85	15.6
C4.5			76.5	104	20.3
Subset			72.6	120	25.4

Table 6.14 Comparison of Rule Induction Methods for Lung Cancer Data.

Method	Data size	Feature	Accuracy (%)	No. of rules	Uncovered region (%)
HybREF	32	56	95.1	36	12.6
C4.5			88.3	44	14.5
Subset			80.6	52	18.0

Table 6.15 Comparison of Rule Induction Methods for Car Data.

Method	Data size	Feature	Accuracy (%)	No. of rules	Uncovered region (%)
HybREF	1728	6	75.2	8	2.9
C4.5			74.8	19	2.4
Subset			71.7	10	2.0

6.4 Experiment Four

Another experiment is made to examine the effectiveness of the discretization method (given in section 3.2 and used in step 1 in Algorithm 3.2: HybREF framework). Three datasets including diabetes, Australian credit and breast-cancer datasets with all numeric features and varying data sizes are selected from the UCI repository [52]. Three discretization methods are chosen to compare their performances to the rule generation. Table 6.16 gives an indication how the discretization methods affect the number of rules. The entropy-based method gives better average accuracy of rules, whereas equal width method performs worse than 1R [47].

Table 6.16 Effectiveness of Discretization Method for Rule Induction.

Dataset	Data Size	Continuous attributes	Average accuracy of rules (%)		
			Equal width	Entropy-based	1R
Diabetes	768	8	65	75	70
Australian credit	690	6	76	86	84
Breast-cancer (Wisconsin)	699	9	83	95	88

6.5 Experiment Five

In our attribute reduction algorithm (Algorithm 4.2: AR), condition attributes can be efficiently represented in bit strings. Besides, we have used this representation to convert the discernibility matrix into a binary form. This representation allows making a better use of computational memory because it takes advantage of the parallel processing over groups up to 64 bits to analyze groups of attributes at one time. Table 6.17 shows the processing time of the procedures of finding attribute reduct. Results have shown that the processing time is improved using binary representation on sets operations.

Table 6.17 Processing Time of Finding Attribute Reduct (Time is given in seconds).

Dataset	Calculation time of finding attribute reduct	
	With binary representation	Without binary representation
Australian credit	3.61	7.45
Diabetes	3.84	7.90
Flags	0.82	2.21
Car	6.27	14.5
TicTacToe	4.20	8.66

6.6 Summary

In this chapter, we use five experiments to analyze the performance of our HybREF framework. The first experiment is carried out on iris data in order to show our framework step by step. Then the second experiment analyzes the utilization of SOM network for noise removing. To evaluate the performance of our framework, we compare our approach with other rule induction methods on five benchmark datasets in the third experiment. Through another experiment, we discuss how the discretization methods affect the number of rules. Finally, we compare the processing time for our method in the fifth experiment.

Chapter 7 Conclusions and Extensions

7.1 Conclusions

This thesis proposes a hybrid framework to generate classification rules based on neural networks and rough sets. A simplified scheme by using SOM neural network for removing noise is used for this hybrid system. Our benchmarking using MONK3 dataset shows average accuracy of 97.8%, 97.7% and 96.5% at 5, 10 and 20 noises respectively. A new attribute reduction algorithm can generate an optimal reduct based on binary discernibility matrix. Finally, a new rule generation algorithm is also proposed to obtain certain and approximate rules.

The key contributions of this work are:

- (1) We introduce the hybrid framework consisting of SOM and rough sets for rule extraction (Algorithm 3.2: HybREF framework).

- (2) For cleaning the noisy data, we apply SOM neural network as the clustering tool to remove the noisy data. The utilization of SOM takes advantage of the simple structure of this neural network. Thus, it has reasonably short training time. Moreover, the SOM provides a direct visualization of the relationship in the attributes, which is helpful for rule induction.
- (3) In attribute reduction, we develop new attribute reduct algorithm (Algorithm 4.2: AR). This algorithm reduces the decision variables represented in bit strings. This algorithm can make better usage of computational memory, which results in an improvement of the processing time.
- (4) In rule induction, we propose a new rule generation algorithm (Algorithm 5.1: RG). This algorithm produces a set of minimal rules including both certain rules and approximate rules.

We have done five experiments on benchmark datasets to analyze our rule extraction framework. The results of the comparison experiments show that our rule generation system can generate concise and simple rules, which are useful to induce knowledge from large information systems. Besides, the bit-wise operations on binary discernibility matrix improve the computation time. To summarize, the hybrid method of neural networks and rough sets presented in this work can serve as a very useful tool complementing other existing rule extraction techniques.

7.2 Extensions

Although our techniques provide a simple process to rule generation, there is still more potential research in this area to improve the efficiency of the proposed rule generation system:

- (1) As not all features in a dataset play key roles in the decision process, we can optimize methods of eliminating those non-key factors before attribute reduction phase. Therefore, a possible extension is to develop an optimized feature-selecting algorithm to reduce superfluous features.
- (2) In order to improve the ability of rule generation, some other rule filtering techniques such as fuzzysets and genetic algorithm can also be integrated with our system.
- (3) Most real-life data in information system are incomplete. Rough set theory can deal with incomplete information system as well. Hence, it is useful to extend our rule-generation algorithm to generate minimal sets of rules from incomplete information system.

Author's Publications

Journal

1. Yi Xu and Narendra S. Chaudhari, "Rule Extraction Frameworks using Neural Networks and Rough Sets", *International Journal of System Modeling and Simulation* (IJSMS) (Accepted for Publication).

Conference

2. Yi Xu and Narendra S. Chaudhari, "Application of Binary Neural Networks for Classification", In, *Proceedings of the second International Conference on Machine Learning and Cybernetics*, Xi'an, China, (IEEE Cat. No.03EX693), Vol. 2, pp. 1343-1348 (November 2-5, 2003).
3. Yi Xu and Narendra S. Chaudhari, "Rule Extraction Framework Using Rough Sets and Neural Networks", In, *11th International Conference on Neural Information Processing* (ICONIP 2004), Calcutta, India (22-25 Nov. 2004). Also revised version published in, *Lecture Notes in Computer Science*, Vol. 3316, pp. 1002-1007, Springer Verlag, Berlin (ISBN: 3-540-23931-6), 2004.
4. Narendra S. Chaudhari and Yi Xu, "Hybrid Rule Extraction Framework using Self-organizing Maps (SOM) and Rough Sets", Accepted for presentation in, *Third International Conference on Computational Intelligence, Robotics and Autonomous Systems*, Singapore (13-16 Dec 2005).

Book Chapter

5. Yi Xu and Narendra S. Chaudhari, "Rule Extraction Frameworks for Data Mining Using Neural Networks", In, *Neural Networks: Applications in Web Technology and Web Engineering*, (Ed. Dianhui Wang and Nung Kion Lee) Borneo Publishing, Sarawak, Malaysia (ISBN: 983-42559-0-x), pp. 48-63, (First Edition, 2005).

References

- [1] S. Mitra, Sankar K. Pal, and P. Mitra, "Data Mining in Soft Computing Framework: a Survey", *IEEE Trans. on neural networks*, vol. 13, no. 1, pp. 3–14, 2002.
- [2] Z. Pawlak, "Rough Sets", *International Journal of Computer and Information Science*, vol. 11, pp. 341-356, 1982.
- [3] J. Han, X. Hu, and T. Y. Lin, "A New Computation Model for Rough Set Theory Based on Database Systems", the *Proc. of 5th International Conference on Data Warehousing and Knowledge Discovery*, Sep. 2003, pp. 381-390.
- [4] M. E. Yahia, R. Mahmod, N. Sulaiman, and F. Ahmad, "Rough Neural Expert Systems", *Expert Systems with Applications*, vol. 18, pp. 87-99, 2000.
- [5] N. Ramakrishnan and A. Y. Grana, "Data Mining: from Serendipity to Science", *Computer*, vol. 32, no. 8, 1999.
- [6] J. Yang, "A Rule Induction-based Knowledge System for Retaining Wall Selection", *Expert Systems with Applications*, vol. 23, no. 3, pp. 273-279, 2002.
- [7] J. Stefanowski, "On Rough Set Based Approaches to Induction of Decision Rules", *Rough Sets in Data Mining and Knowledge Discovery*, vol. 1, pp. 500-529, 1998.
- [8] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "The KDD Process for Extracting Useful Knowledge from Volumes of Data", *Communications of the ACM*, vol. 39, no. 11, pp. 27-35, 1996.
- [9] J. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.

References

- [10] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules", the Proc. of the 20th VLDB Conference, Santiago, Chile, 1994, pp. 487-499.
- [11] G. J. Klir and M. J. Wierman, "Uncertainty-Based Information: Elements of Generalized Information Theory", Report, Center for Research in Fuzzy Mathematics and Computer Science, Creighton University, Omaha, Nebraska 68178, U. S. A., 1997.
- [12] L. Zadeh, "Fuzzy Sets", Information and Control, vol. 8, pp. 338-353, 1965.
- [13] G. J. Klir, "Multivalued Logics Versus Modal Logics: Alternative Frameworks for Uncertainty Modelling", Advances in Fuzzy Theory and Technology, Department of Electrical Engineering, Duke University, Durham, North Carolina, pp. 3-47, 1994.
- [14] T. Kohonen et al., "Self organization of a massive document collection", IEEE Trans. Neural Networks, vol. 11, pp. 54-585, 2000.
- [15] D. Shalvi and N. D. Claris, "Unsupervised neural network approach to medical data mining techniques", the Proc. IEEE Int. Joint Conf. Neural Networks, May 1998, pp. 171-176.
- [16] X. Hu, N. Cercone, and W. Ziarko, "Generation of Multiple Knowledge from Databases Based on Rough Sets Theory", Rough Sets and Data Mining, Kluwer, pp. 109-121, 1997.
- [17] Z. Pawlak, "Rough Classification", International Journal of Human-computer Studies, vol. 51, pp. 369-383, 1999.
- [18] M. Beynon, B. Curry, and P. Morgan, "Classification and Rule Induction Using Rough Set Theory", Expert Systems, vol. 17, no. 3, pp. 136-148, 2000.

References

- [19] N. Zhong, J. Dong and S. Ohsuga, "Rule Discovery by Soft Induction Techniques", *Neurocomputing*, vol. 36, pp. 171-204, 2001.
- [20] J. Stefanowski and D. Vanderpooten, "Induction of Decision Rules in Classification and Discovery-oriented Perspectives", *International Journal of Intelligent Systems*, vol. 16, pp. 13-27, 2001.
- [21] X. Hu and N. Cercone, "Discovering Maximal Generalized Decision Rules through Horizontal and Vertical Data Reduction", *Computational Intelligence*, vol. 17, no. 4, pp. 685-702, 2001.
- [22] J. Yang, "A Rule Induction-based Knowledge System for Retaining Wall Selection", *Expert System with Applications*, vol. 23, no. 3, pp. 273-279, 2002.
- [23] Y. Leung and D. Li, "Maximal Consistent Block Technique for Rule Acquisition in Incomplete Information Systems", *Information Sciences*, vol. 153, pp. 85-106, 2003.
- [24] J. Zhang et. al., "A New Heuristic Reduct Algorithm Based on Rough Sets Theory", *International Conference on Web-Age Information Management*, LNCS 2762, 2003, pp. 247-253.
- [25] S. Tsumoto, "Mining Diagnostic Rules from Clinical Databases Using Rough Sets and Medical Diagnostic Model", *Information Sciences*, vol. 162, pp. 65-80, 2004.
- [26] A. E. Hassanien and J. M. H. Ali, "Rough Set Approach for Generation of Classification Rules of Breast Cancer Data", *Informatica*, vol. 15, no. 1, pp. 23-38, 2004.
- [27] A. M. Radzikowska and E. E. Kerre, "A Comparative Study of Fuzzy Rough Sets", *Fuzzy Sets and Systems*, vol. 126, no. 2, pp. 137-155, 2002.

References

- [28] Y. Y. Yao, "Combination of Rough and Fuzzy Sets Based on α -level Sets", *Rough Sets and Data Mining: Analysis for Imprecise Data*, Kluwer Academic Publishers, Boston, pp. 301-321, 1997.
- [29] A. Nahamura and J. M. Gao, "On a KTB-model fuzzy logic", *Fuzzy Sets and Systems*, vol. 45, pp. 327-334, 1992.
- [30] Z. Pawlak, *Rough Sets: Theoretical Aspects of Reasoning about Data*, Kluwer, Boston, 1991.
- [31] A. Kusiak, *Data Mining: Computational Intelligence in Design and Manufacturing*, John Wiley, New York, 2000.
- [32] S. Mitra and Y. Hayashi, "Neuro-fuzzy Rule Generation: Survey in Soft Computing Framework," *IEEE Trans. On Neural Network*, vol. 11, no. 3, pp. 748–768, 2000.
- [33] L. X. Wang and J. M. Mendel, "Generating Fuzzy Rules by Learning from Examples", *IEEE Trans. Systems, Man, and Cybernetics*, vol. 22, pp. 1414–1427, 1992.
- [34] R. Rovatti and R. Guerrieri, "Fuzzy Sets of Rules for System Identification," *IEEE Trans. Fuzzy Syst.*, vol. 4, pp. 89–102, 1996.
- [35] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka, "Selecting Fuzzy If-Then Rules for Classification Problems Using Genetic Algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 260–270, 1995.
- [36] L. Wang and J. Yen, "Extracting Fuzzy Rules for System Modeling Using a Hybrid of Genetic Algorithms and Kalman Filter," *Fuzzy Sets Syst.*, vol. 101, pp. 353–362, 1999.

References

- [37] P. Mitra and S. K. Pal, "Staging of Cervical Cancer with Soft Computing", IEEE Trans. On Biomedical Engineering, vol. 47, no. 7, pp. 934–940, 2000.
- [38] Y. Jung-heurn, Y. Seung-moo, and J. Hong-Tae, "Structure Optimization of Fuzzy-Neural Network Using Rough Set Theory", the Proceedings of 1999 IEEE International Conference on Fuzzy Systems, Aug 1999, pp. 1666-1670.
- [39] B. S. Ahn, S. S. Cho, and C. Y. Kim, "The Integrated Methodology of Rough Set Theory and Artificial Neural Network for Business Failure Prediction," Expert Systems with Applications, vol. 18, pp. 65–74, 2000.
- [40] R. Yasdi, "Combining Rough Sets Learning and Neural Learning Method to Deal with Uncertain and Imprecise Information", Neurocomputing, vol. 7, pp. 61–84, 1995.
- [41] R. W. Swiniarski and L. Hargis, "Rough Sets as a Front End of Neural Networks Texture Classifiers", Neurocomputing, vol. 36, pp. 85–102, 2001.
- [42] R. R. Hashemi et. al., "A Hybrid Intelligent System for Predicting Bank Holding Structures", European Journal of Operational Research, vol. 109, pp. 390-402, 1998.
- [43] P. J. Lingras, "Rough Set Clustering for Web Mining," Proceedings of 2002 IEEE International Conference on Fuzzy Systems, Hawaii, May 2002, pp. 12-17.
- [44] S. K. Pal, S. Mitra, and P. Mitra, "Rough-Fuzzy MLP: Modular Evolution, Rule Generation, and Evaluation", IEEE Trans. On Knowledge and data engineering, vol. 15, no. 1, pp. 14-25, Jan/Feb 2003.
- [45] T. Kohonen, Self-organizing Maps, 3rd ed. Berlin, Germany: Springer, 2001.

References

- [46] C. C. Chan, C. Batur, and A. Srinivasasn, "Determination of Quantization Intervals in Rule Based Model for Dynamic Systems", Proceedings of the IEEE conference on Systems, Man and Cybernetics, 1991, pp. 1719-1723.
- [47] H. Liu, F. Hussain, C. L. Tan and M. Dash, "Discretization: An Enabling Technique", Journal of Data Mining and Knowledge Discovery, vol. 6, no. 4, pp. 393-423, 2002.
- [48] U. M. Fayyad and K. B. Irani, "Multi-interval Discretization of Continuous-valued Attributes for Classification Learning," Proceedings of the 13th International Joint Conference on *Artificial* Intelligence, Morgan Kaufmann, 1993, pp. 1022-1027.
- [49] R. Felix and T. Ushio, "Binary Encoding of Discernibility Patterns to Find Minimal Coverings", International Journal of Software Engineering and Knowledge Engineering, vol. 12, no. 1, pp. 1-18, 2002.
- [50] A. Skowron and C. Rauszer, "The Indiscernibility Matrices and Functions in Information Systems", Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory, Kluwer Academic Publishers, pp. 331-362, 1992.
- [51] J. A. Starzyk, N. Dale and K. Sturtz, "A Mathematical Foundation for Improved Reduct Generation in Information Systems", Knowledge and Information Systems Journal, pp. 131-147, 2000.
- [52] P. M. Murphy and D. W. Aha, UCI Repository of machine learning databases and domain theories, 1995. <http://www.ics.uci.edu/~mllearn/MLRepository.html>.
- [53] G. G. Towell and J. W. Shavlik, "Extracting Refined Rules from Knowledge-based Neural Networks", Machine Learning, vol. 13, no. 1, pp. 71-101, 1993.