Classification of Weather Data: A Rough Set Approach

By

Songqing Shan

A Thesis Submitted to the Faculty of Graduate Studies
in Partial Fulfillment of the Requirements for the Degree of

Master of Science
in Computer Engineering

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

*****

COPYRIGHT PERMISSION PAGE

Classification of Weather Data: A Rough Set Approach

BY

Songqing Shan

A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

Abstract

Meteorological volumetric radar data are used to detect thunderstorms. The classification of storm cells is a difficult problem due to the complex evolution of them, the high dimensionality of the weather data, and the imprecision and incompleteness of the data. This thesis investigates the classification theory and approaches of rough set, and use them to classify different types of storm events. The rough set classification strategies are compared with other ones to determine which approaches will best classify the volumetric storm cell data coming from the Radar Decision Support System database of Environment Canada. The criterion for comparison is the accuracy coefficient in the classification over a testing data. The results obtained with the rough set approach show that they are a little better than other ones, in terms of accuracy, for the volumetric storm cell classification.

Acknowledgements

I wish to express my gratitude and appreciation to Dr. J. F. Peters, Electrical & Computer Engineering Department, for his guidance, advice and encouragement throughout my investigation.

Also, I have had the pleasure of collaborating with Dr. Z. Suraj, Institute of Mathematics, Pedagogical University, and want to thank him for his help and suggestions.

And, of course, many thanks to my family for their support. Without them, none of this would have been realized.

This research was funded by National Research Council (NRC) Laboratory.

Table of Contents

List of Tables

List of Figures

Nomenclature

Notation	Explanation	Section
S	An information system	2.1
U	A non-empty finite set of objects called the universe	2.1
A, B, C	Sets of attributes in an information system	2.1 2.2
a_1, a_2, a_3	Individual Attribute in an information system	2.1
V_a	A value set of individual attribute a	2.1
V	The range of the attributes A	2.1
$\cup$	Set operator – union	2.1
U	Individual object in an information system	2.1
$A(u)$	Individual attribute value of the object u	2.1
D	A non-empty finite set of decision attributes	2.1
$d\ (U)$	Cardinality of the image $d(U)$	2.1
R (d)	Rank of decision attributes d	2.1
V_d	A value set of decision attributes d	2.1
X_I	The *i-th* decision class of the information system	2.1 2.2
$Ind_S(B)$, $Ind_S(C)$	Indiscernibility relation on objects in an information system relative to a set of attributes B or C.	2.2
R	Binary relation	2.2
X	Sets of objects in an information system	2.2
$[u]_B$	An equivalence class of the B-indiscernibility relation in information system	2.2
$\underline{B}X$	Lower approximation of set X relative to attributes in set B	2.2
$\overline{B}X$	Upper approximation of set X relative to attributes in set B	2.2
$BN_B(X)$, $BN_S(X)$	B-boundary of sets of objects in an information system Relative to a set of attributes B	2.2
$POS_B(d)$, $POS_B(C)$	B-positive region of an information system relative to a set attributes B	2.2
RED(S)	A set of reducts	2.3
M(S)	A discernibility matrix	2.3
C_{ij}	A element of the discernibility matrix	2.3
$a_1^*, \ldots, a_m^*$	Boolean variables corresponding to attributes	2.3
$F_{M(S)}$	Discernibility function of an information system	2.3
$\vee c_{ij}^*$	Disjunction of all elements of c_{ij}^*	2.3
c_{ij}^*	$c_{ij}^* = \{a^* : a \in c_{ij}\}$	2.3
F(B, V)	Set of conditional formulae of an information system	2.4
τ	Formulae in an information system	2.4
$\|\tau\|_S$, $\|\tau\|$	$\|a = v\| = \{u \in U: a(u) = v\}$ for $a \in B$ and $v \in V_a$; $\|\tau \vee \tau'\| = \|\tau\| \cup \|\tau'\|$; $\|\tau \wedge \tau'\| = \|\tau\| \cap \|\tau'\|$; $\|\neg\tau\| = U - \|\tau\|$.	2.4
φ	Individual conditional formulae	2.4
$\|\varphi\|$	Set of objects matching the decision rule	2.4
W	Subset of values of attributes	2.4

Notation	Explanation	Section
T	Template	2.4
$D_1, \ldots, D_m$	Descriptors	2.4
κ	A classifier	3.1
$\underline{D}_\kappa$	Classification function	3.1
$C(i,j)$	Confusion matrices	3.3
P( )	Probability function	3.3
KDD	Knowledge discovery from database	3.4
FR	Full reducts	3.5
OOR	Objected oriented reducts	3.5
GR	Genetic reducts	3.5
DR	Dynamic reducts	3.5
DT	Decomposition tree	3.5
RDSS	Radar decision support system	4.0
DBZ	Radar reflectivity factor	4.1
UTC	Universal time clock	4.2
VIL	Vertically integrated liquid	4.4
BWER	Bounded weak echo region	4.4
Aver	Average	tables

1 Introduction

Weather forecast plays an important role in today's daily life. Accurately forecasting severe weather events reduces loss of life and assets to a great extent. A prerequisite to short-term weather prediction is the ability to classify severe weather cells at different stages in their life cycles. Meteorologists rely on volumetric radar data to forecast weather phenomena. A radar data processing system gathers meteorological volumetric radar data by conducting a volume scan. Meteorologists use these radar data to detect thunderstorms. Radar subsystem exists that allows operational meteorologists to focus their attention on the regions of interest within the volumetric radar scan known as storm cells. When a storm is found, a number of parameters known as products or features are computed including its geographical location, volume, vertically integrated reflectivity, precipitation accumulation, maximum wind gust potentials, gradient profiles, and bounded weak echo regions. But it is difficult to classify detected storm cells into a specific type of storm event due to a number of confounding factors such as incomplete data, complex evolution of storm cells and high dimensionality of the data.

The principal object of this thesis is to identify patterns in the data that indicate, with a high degree of accuracy, the onset of a severe weather event using either the derived features of matched–cell files from RDSS [Wes99], or the raw data of the volume scans. Several approaches (including fuzzy clustering [Ale99], neural networks [LPP99], [APP99], genetic algorithms [LPPWV00], the support vector machine [RPP00]) have been attempted for classifying volumetric storm cells. But the primary focus so far has been to apply neural network classification

strategies, and techniques and results of these efforts can be found in reports of prior work [Ale99, LPP99, APP99, LPPWV00, RPP00]. In recent years, there has been considerable research concerning the application of rough set methods in solving classification problems [JA01, JFA01, AJJ01, AJ01, AC92, Ohrn, OKSS98, Paw91]. In this thesis, the rough set theory is used to construct weather data classification model and its approach is used to classify both four types of storm events: hail, heavy rain, tornado and wind or ten types of storm events: hail, heavy rain, tornado, wind, hail or rain, hail or tornado, hail or wind, rain or tornado, rain or wind, and tornado or wind. The rough set classification strategies are compared with other ones presented in [RPP00] to determine which approach would best classify the volumetric storm cell data coming from the Radar Decision Support System database of Environment Canada. The criterion for comparison is the accuracy coefficient in the classification over a testing data. The results obtained with the rough set approach show that it is a little better than the approaches presented in [APP99], [RPP00], in terms of accuracy, for the volumetric storm cell classification.

Rough sets and Pawlak information systems [Paw91] have recently gained rather substantial scientific interest, especially in the field of constructing models that describe or classify measurements. Rough set based data analysis starts from raw data represented by a data table, called an *information system*. Real-life data are frequently imperfect: incomplete, uncertain and vague. The information system contains data about *objects* of interest characterized in terms of some *attributes*, which represent some features of the objects, and each row represents an object by

all its *attribute values*. The *domain* of each attribute may be either symbolic or numerical. We assume that all attributes of input data are numerical. Numerical attributes, after discretization, become symbolic as well. Often we distinguish in the information system conditions and decision attributes. Such information system is called a *decision table*. The decision table describes decisions in terms of conditions that must be satisfied in order to carry out the decision specified in the decision table. For each object, there is a *decision value* associated with it. The set of all objects with the same decision value is called a *decision class*. The decision table is *inconsistent* if there exists two objects with all attribute values identical, but belongs to different decision classes. With every decision table we can associate a decision algorithm that is a set of *if* ... *then* decision rules. The decision rules can be also seen as a logical description of approximation of decisions, and consequently a decision algorithm can be viewed as a logical description of basic properties of the data. The decision algorithm can be simplified, what results in optimal description of the data. The optimal decision rules can be found and are used to classify new objects. Rough set theory has various classification strategies. In this thesis, Full Reducts, Object Oriented Reducts, Genetic Reuducts, Dynamic Reducts, and Decomposition Tree are used to classify the storm cells. The results of classifying the storm cells are presented.

The structure of this thesis is as follows. Chapter 1 (this section) introduces the subjects and structure of this thesis. Chapter 2 presents the preliminaries of the rough set theory. Chapter 3 gives the short description of investigated methods for the volumetric storm cell classification. Chapter 4 outlines radar data acquisition

and its derided features. In chapter 5, the methodology and experimental results are presented. Chapter 6 is dedicated to the presentation of the conclusions from the present work and future work.

2 Preliminaries of Rough Sets

Rough set theory was presented by Zdzilaw Pawlak [PAW91] in the early 1980's and investigated by some authors J.F.Peters, A. Skowron [JZL99, ZA94, ZJA01, JA01]. Rough set methods have been introduced as a means of dealing with the classificatory analysis of data (inputs) for an information system. The data in an information system can be acquired, for example, from sensor measurements or from human experts. The main goal of the rough set analysis is to approximate sets of data relative to the values of selected system attributes representing our problem domain knowledge. Our objective is to classify unseen objects that supply a partial picture of what we are trying to classify (e.g., storm cells) based on input data (e.g., volumetric radar data). In this thesis, severe weather storm cells are classified based on available radar data.

2.1 Information Systems and Decision tables

An information system is represented as a table, where each row presents a case, an event, a storm cell, or simply an object (see Table 2.1) and every column represents an attribute (a variable, an observation, a property, etc) that can be measured for each object. A human expert or user may also supply the attribute. This table is called an information system. Information systems (sometimes called data table, attribute systems, condition–action tables, knowledge representation system etc.) are used for representing knowledge.

More formally, an *information system* is a pair $S = (U, A)$, where U - is a non-empty, finite set of objects called the *universe*, A - is a non-empty, finite set of

attributes, i.e., a: $U \rightarrow V_a$ for $a \in A$, where U is a domain and V_a is called the *value set* of a. The set $V = \bigcup_{a \in A} V_a$ is said to be the *range* of A. An information system can be represented as a finite data table, in which the columns are labeled by attributes, the rows by objects and on the position corresponding to the row u and column a the value $a(u)$ appears. Each row in the table describes the information about some object in S.

Let us consider an information system $S = (U, A)$ such that $U = \{u_1, u_2, u_3\}$, $A = \{a_1, a_2, a_3\}$, where the values of the attributes are given in Table 2.1. In this case,

Table 2.1 an example information system

U/A	a_1	a_2	a_3
u_1	5	600	70
u_2	10	500	30
u_3	5	600	70

we assume that objects labeled by u_1, u_2, u_3 denote the storm cells and attributes a_1, a_2, and a_3 denote the features (actually have more features) of the storm cells, whereas entries of the first row in Table 2.1 5, 600, 70 denote height offset, volume, and orientation of storm cell u_1. The entries in other rows have the same meaning as that of the first row.

In many applications there is an outcome of classification that is known. This a posteriori knowledge is expressed by one distinguished attribute called decision attribute. Information systems of this kind are called decision systems. A *decision system* (a *decision table*) is any information system of the form $S = (U, A \cup \{d\})$,

where $d \notin A$ is a distinguished attribute called *decision*. The elements of A are called *conditional attributes* (*conditions*). The decision attributes may take several values. One can interpret a decision attribute as a sort of a classification of the universe of objects given by an expert (decision-maker, operator, physician, etc.). The cardinality of the image $d(U) = \{k: d(u) = k \text{ for some } u \in U\}$ is called the *rank* of d and is denoted by $r(d)$. We assume that the set V_d of values of the decision d is equal to $\{1, \ldots, r(d)\}$. Let us observe that the decision d determines a partition $\{X_1, \ldots, X_{r(d)}\}$ of the universe U, where $X_k = \{u \in U: d(u) = k\}$ for $1 \leq k \leq r(d)$. The set X_i is called the *i-th decision class* of S. Any decision system $S = (U, A \cup \{d\})$ can be represented by data table with the number of rows equal to the cardinality of the universe U and the number of columns equal to the cardinality of the set $A \cup \{d\}$. On the position corresponding to the row u and column a the $a(u)$ appears.

Let us extend Table 2.1 to form a small example information decision system, Table 2.2. Table 2.2 has the same three storm cells as in the Table 2.1, but one decision attribute d with several possible outcomes has been added. It may be noticed that u_1 and u_3 have exactly the same values of conditions, but have a different outcome (different value of the decision attribute). An information system (i.e. a decision table) expresses all the knowledge about the model. But the information system may be unnecessarily large partly because the same or indiscernible objects may be represented several times or partly because some of the attributes may be superfluous. Generally speaking, there may be redundant components in it. In the next sections, the notions related with above questions are discussed.

Table 2.2 an example information decision system

U/A	a_1	A_2	a_3	d
u_1	5	600	70	1
u_2	10	500	30	3
u_3	5	600	70	2

2.2 Indiscernibility, Equivalence Class and Set Approximation

In the given information system Table 2.1, we are not able to distinguish all single objects in terms of the available attributes. Namely, different objects can have the same values on considered attributes. Hence, any set of attributes divides the universe U into some classes that establish a partition [PAW91] of the set of all objects U. It is defined in the following way. Let $S = (U, A)$ be an information system. With any subset of attributes $B \subseteq A$ we associate a binary relation $\mathrm{Ind_S(B)}$, called an *indiscernibility relation*, which is defined by

$$\mathrm{Ind_S(B)} = \{ (u,u') \in U \times U \text{ for every } a \in B,\ a(u) = a(u') \}.$$

If u $\mathrm{Ind_S(B)}$ u', then we say that the objects u and u' are indiscernible with respect to attributes from B. In other words, we cannot distinguish u from u' in terms of attributes in B. $\mathrm{Ind_S(B)}$ is an equivalence relation. The notion of equivalence is recalled in the following. A binary relation $R \subseteq U \times U$ that is reflexive (i.e. an object is in relation with itself uRu), symmetric (if uRv then vRu) and transitive (if uRv and vRw then uRw) is called an equivalence relation. The equivalence class of an element $u \in U$ consists of all objects $v \in U$ such that uRv. The equivalence classes of the *B-indiscernibility* relation are denoted $[u]_B$.

The equivalence class notation can be used to partition the universe. The partitions can build new subsets of the universe. Subsets that are most often of interest have the same value of the outcome attribute. It may happen, however, that a concept cannot be defined in a crisp manner. They can only be roughly (approximately) defined. The idea of rough sets consists of the approximation of a set by a pair of sets, called lower and upper approximation of this set. Let $S = (U, A)$ be an information system, $B \subseteq A$ be a set of attributes, and let $X \subseteq U$ denotes a set of objects. Then the sets $\{u \in U: [u]_B \subseteq X\}$ and $\{u \in U:[u]_B \cap X \neq \phi\}$ are called *B-lower and B-upper approximation* of X in S, and denoted by $\underline{B}X$ and $\overline{B}X$, respectively. The set $BN_B(X) = \overline{B}X - \underline{B}X$, will be called the *B-boundary* of X. When $B = A$, we write also $BN_S(X)$ instead of $BN_A(X)$. Sets that are unions of some classes of the indiscernibility relation $Ind_S(B)$ are called definable by B. The set X is *B-definable* if and only if $\underline{B}X = \overline{B}X$. The set $\underline{B}X$ is the set of all elements of U which can be with certainty classified as elements of X, given the knowledge represented by attributes from B, $\overline{B}X$ is the set of elements of U which can be possibly classified as elements of X, employing the knowledge represented by attributes from B, and $BN_B(X)$ is the set of elements which can be classified neither to X nor to $-X$ given knowledge B. If $X_1, \ldots, X_{r(d)}$ are decision classes of S then the set $\underline{B}X_1 \cup \ldots \cup \underline{B}X_{r(d)}$ is called the *B-positive region* of S and denoted by $POS_B(d)$. If $C \subseteq A$ then the set $POS_B(C)$ is defined as $POS_B(d)$ where $d(u) = \{a(u):a \in C\}$ for $u \in U$ *is* an attribute representing the set C in of attributes and called the *B-positive* region of C in S. The *B-positive* region of C in S contains all objects in U which

can be classified perfectly without error into distinct classes defined by $Ind_S(C)$, based only on information in relation $Ind_S(B)$.

2.3 Attribute Reduction, Discernibility Matrix and Reducts Computing

Some attributes in an information system may be redundant and can be eliminated without losing essential classificatory information. The process of finding a smaller set of attributes (other than the original one), with the same or close classificatory power as the original set is called attribute reduction. As a result the original larger information system may be reduced to a smaller system containing attributes. Rough set theory makes it possible us to determine for a given information system the minimum number of attributes (reducts) needed to classify input data. That is, a reduct is a minimal set of attributes $B \subseteq A$ that can be used to discern all objects obtainable by all of the attributes of an information system. That is, $Ind_S(B) = Ind_S(A)$. In effect, a reduct is a subset B of attributes A of information system S that preserves the partitioning of the universe U. Hence, a reduct can be used to perform the same classifications as the whole attribute set A of the information. The intersection of all the reducts in S is called the *core* of S.

Let $S = (U, A)$ be an information system, and let us assume that $U = \{u_1, \ldots, u_n\}$, and $A = \{a_1, \ldots, a_m\}$. By $M(S)$ we denote an $n \times n$ matrix (c_{ij}), called the *discernibilty matrix* of S, such that $c_{ij} = \{a \in A : a(u_i) \neq a(u_j)\}$ for $i, j = 1, \ldots, n$. Intuitively an entry c_{ij} consists of all the attributes discerning objects u_i and u_j. Since $M(S)$ is symmetric and $c_{ii} = \varnothing$ for i = 1, …, n, $M(S)$ can be represented using only elements in the lower triangular part of $M(S)$, i.e., for $1 \leq j < i \leq n$. With every

discernibility matrix $M(S)$ one can uniquely associate a *discernibility function* $f_{M(s)}$, defined as follows. A *discernibility function* $f_{M(S)}$ for an information system S is a Boolean function of m prepositional variables $a_1^*, \ldots, a_m^*$ (where $a_i \in A$ for $i = 1, \ldots, m$) defined as the conjunction of all expressions $\vee c_{ij}^*$, where $\vee c_{ij}^*$ is the disjunction of all elements of $c_{ij}^* = \{a^* : a \in c_{ij}\}$, where $1 \leq j < i \leq n$ and $c_{ij} \neq \varnothing$. In the sequel we write a instead of a^*.

With the help of discernibility matrix, the steps of computing the set of all reducts in a given information system S are as follows:

(1) Compute the discernibility matrix for the system S.

(2) Compute the discernibility function $f_{M(S)}$ associated with the discernibility matrix M(S).

(3) Compute the minimal disjunctive normal form of the discernibility function $f_{M(S)}$ (The normal form of the function yields all the reducts).

The finding of minimal (with respect to cardinality) reducts for an information system is a combinatorial NP-hard problem [RSC92]. This means that computing reducts is a non-trivial task that cannot be solved by a simple-minded increase of computational resources. It is, in fact, one of the bottlenecks of the rough set methodology. Finding the reducts can be considered similarly as finding the minimal disjunctive normal form for a logical expression given in the conjunctive normal form [PAW91]. In general the number of reducts of a given information system can be exponential with respect to the number of attributes (i.e., any

information system S has at most m over [m/2] reducts, where reduct m = card(A)). Fortunately, existing procedures for reduct computation are efficient in many applications and for more complex cases one can apply some efficient heuristics (see e.g. [BSS94, NSS95, SA95, NgHo1997]).

2.4 Decision Rules and Templates

One of the important applications of rough sets is the generation of decision rules for a given information system for classification of known objects, or prediction of classes for new objects unseen during design. Using an original or reduced decision table, one can find the rules classifying objects through determining the decision attributes value based on condition attributes values. Rules express some of the relationships between values of the attributes described in the information systems.

Let $S = (U, A \cup \{d\})$ be a decision table and let $V = \bigcup_{a \in A} V_a \cup V_d$. Atomic formulae over $B \subseteq A \cup \{d\}$ and V are expressions of the form $a = v$; they are called *descriptors* over B and V, where $a \in \boldsymbol{B}$ and $v \in V_a$. The set F(B, V) of formulae over B and V is the least set containing all atomic formulae over B and V and closed under prepositional connectives: $\neg$ (negation), $\vee$ (disjunction) and $\wedge$ (conjunction). The meaning $\|\tau\|_S$ (or in short $\|\tau\|$) of the formulae τ in S is defined inductively as follows:

$$\|a = v\| = \{u \in U: a(u) = v\} \text{ for } a \in B \text{ and } v \in V_a;$$

$$\|\tau \vee \tau'\| = \|\tau\| \cup \|\tau'\|; \|\tau \wedge \tau'\| = \|\tau\| \cap \|\tau'\|;$$

$$\|\neg\tau\| = U - \|\tau\|.$$

The set F(A,V) is called the set of *conditional formulae* of S. A *decision rule* in S is any expression of the form $\varphi \Rightarrow d = v$, where $\varphi \in$ F(A, V), $v \in V_d$ and $\|\varphi\| \neq \varnothing$. The decision rule $\varphi \Rightarrow d = v$ is *true* in S if and only if $\|\varphi\| \subseteq \|d = v\|$; $\|\varphi\|$ is the set of objects *matching* the decision rule; $\|\varphi\| \cap \|d = v\|$ is the set of objects *supporting* the rule. The decision rule $\varphi \Rightarrow d = v$ is *minimal* in S if and only if it includes a minimal number of descriptors on its left hand side. The reducts (of all the various types) can be used to synthesize minimal decision rules. Once the reducts have been computed, the rules are easily constructed by overlaying the reducts over the originating decision table and reading off the values.

Let $S = (U, A)$ be an information system. The notion of a descriptor can be generalized by using terms of the form $(a \in W)$, where $W \subseteq V_a$ is a subset of values of a. By a *template* (a *pattern*) we mean the conjunction of descriptors, i.e. $\boldsymbol{T} = D_1 \wedge \ldots \wedge D_m$, where $D_1, \ldots, D_m$ are either simple or generalized descriptors. An object *satisfies* (*matches*) a template if and only if for every attribute a occurring in the template the value of this attribute on considered object is equal to v (belongs to W, in case of generalized template).

2.5 Rule Application and Classification of New Objects

The purpose of every classification system is correct recognition of objects for which it has been designed to deal with. Once we have achieved a decision system with properly classified examples, there are many concepts of how to classify new objects. The simplest way is to compare the attribute value vector of a new object

with value of vectors of a template in order to find an exact match. If there is such a match, the new object is stated to belong to the same class as its matched counterpart. If the table contains no object with the same value vector, the new object is said to be rejected.

There are two kinds of errors in the process of recognition new objects, one is rejection and the other is wrong classification i.e. the new object is assigned a class to which it actually does not belong. Wrong classification and rejection are usually two sides of a modal: if the decision system easily accepts new objects, the wrong classification may be high and vice versa. This is because rejection is the results of information overloading in the decision system when wrong classification is due to information shortage. Many concepts have been developed to improve correct recognition as well as to reduce rejection rate. One of them is to compute reducts and decision rules corresponding to these reducts to decrease the information overloading with classification structure remaining intact. When a set of rules have been induced from a decision system containing a set of training examples, they can be inspected to see if they reveal any novel relationships between attributes that are worth pursuing for further research. Furthermore, the rules can be applied to a set of unseen objects in order to estimate their classificatory power. One of application schemes is as follows.

(1) When a rough set classifier is presented with a new objects, the rule set is scanned to find applicable rules, i.e. rules whose predecessors match the object.

(2) If no rule is found (i.e. no rule "fires"), the most frequent outcome in the training data is chosen.

(3) If more than one rule fires, those may in turn indicate more than one possible outcome.

(4) A voting process is then performed among the rules that fire in order to resolve conflicts and to rank the predicted outcomes. A rule casts as many votes in favor of its outcome as its associated support count. The votes from all the rules are then accumulated and divided by the total number of votes cast in order to arrive at a numerical measure of certainty for each outcome. This measure of certainty is not really a probability, but may be interpreted as an approximation to such, if the model is well calibrated.

Another approach is to use special classification function that examines the similarity of new objects to already classified objects instead of their identical matching. Such functions are usually referred to as distances measures because they are based on some similarities (expressed often in terms of distances) of new objects to those present in the decision system. They can greatly reduce the rejection rate but must be chosen with special care to assure the correctness of the classification.

3 Rough Set Classification Strategies

3.1 Introduction to Classifier Evaluation

In practical applications, one of the main purposes of rough set data analysis is to induce rules from data represented as information or decision systems. Then the resulting decision rules can be used to classify new and unseen objects, i.e., they can be employed to realize *classifiers* (*decision algorithms*, i.e. sets of decision rules together with methods for conflict resolving when they classify new objects). Classifiers induced from empirical data can be evaluated with respect e.g. performance. By performance is meant assessment of how well the classifier does in classifying new cases, according to some specified performance criterion. A classifier κ can be treated as a realization of a function $\underline{d}_\kappa$: $U \rightarrow V_d$, where V_d denotes a set of decision values in a decision system $S = (U, A \cup \{d\})$, that, when applied to an object $u \in U$, assign a classification $\underline{d}_\kappa(u)$ to u. The true current classification of u is denoted $d(u)$. We assume in the following that κ is forced to make a classification when presented with an object. In the case that the classifier fails to recognize an object, a default classification is assumed invoked.

3.2 Partitioning the Examples

In supervised learning we are given a set of labeled example objects in a decision system S, and we want to construct a mapping $\underline{d}_\kappa$ that maps elements in U to elements in V_d, using only attributes contained in A. In practice S is almost always a finite and limited collection of possible examples, it is customary to randomly divide the examples in S into two disjoint subsets, a *training set* and a *test set*. The

training set is used to construct κ, while the test set is used to assess its performance. Under assumption that the two sets comprise independent samples, this ensures us that the performance estimate will be unbiased.

3.3 Confusion Matrices

A *confusion matrix* C is a card(V_d) × card(V_d) matrix with integer entries that summarizes the performance of a classifier κ, applied to the objects in a decision system S [Ohrn]. Without loss of generality we may assume that V_d is the set of integers $\{1, \ldots, r(d)\}$, as defined in Section 2.1. The entry $C(i,j)$ counts the number of objects that really belong to the decision class i, but were classified by κ as belonging to the decision class j, i.e., $C(i, j) = \text{card}\{\, u \in U : d(u) = i \text{ and } \underline{d}_\kappa(u) = j \,\}$. Of course, it is desirable for the diagonal entries to be as large as possible. Probabilities are easily estimated from the confusion matrix C by dividing an entry by the sum of the row or column the entry appears in, i.e.

$$P(d(u) = i / \underline{d}_\kappa(u) = j) = \frac{C(i,j)}{\sum_i C(i,j)},$$

$$P(\underline{d}_\kappa(u) = j / d(u) = \text{i}) = \frac{C(i,j)}{\sum_j C(i,j)},$$

$$P(d(u) = \underline{d}_\kappa(u)) = \frac{\sum_i C(i,i)}{\sum_i \sum_j C(i,j)}.$$

The last equality defines a so called the *accuracy* of the classifier. The accuracy is the proportion of correctly classified objects, and it is the most popular performance measure in the machine learning literature.

3.4 The KDD Process Using Rough Sets

This subsection outlines the steps that constitute the process of knowledge discovery from databases (KDD) using rough sets approach [OKSS98]. The overall KDD process may be broken up into several steps and phases that are iterated in so called a waterfall-like cycle [FPS96] shown as Figure 3.1. From a data source containing raw data, all or portions of this is selected for further processing. The selected raw data is then typically pre-processed and transformed in some way, before being passed on to the data-mining algorithm itself. The output patterns from the computational mining procedure are then post-processed, interpreted and evaluated, hopefully revealing new knowledge previously buried in the data. Along the way, backtracking on each of the steps will in practice inevitably occur.

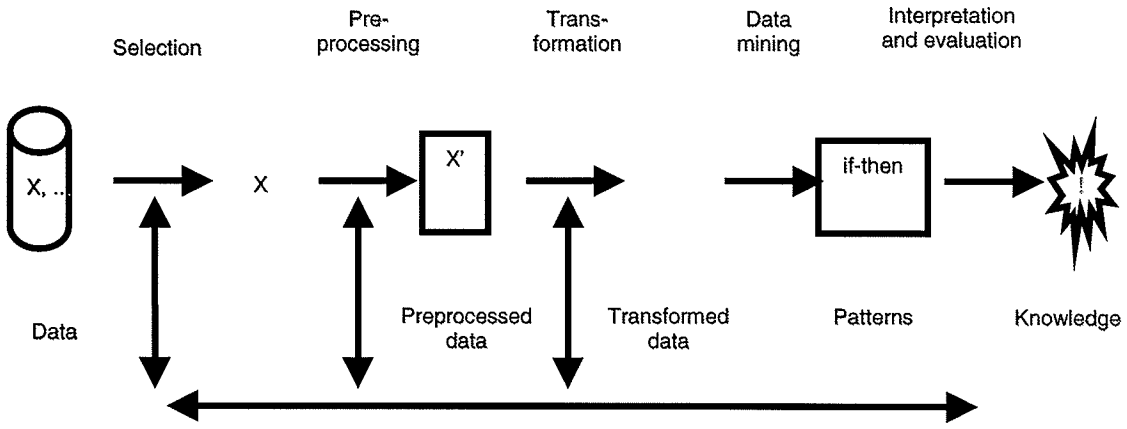

Figure 3.1 a scheme of the overall KDD process

In a rough set framework, the KDD process has usually the following structure: A decision table is read and pre-processed by an attribute discretization

algorithm. After discretization of data we obtain a coarser view of the world through making the attributes' value sets smaller. For numerical attributes, as it is in our experiments described in this thesis, we can introduce intervals which in turn may be given linguistic labels and be treated as qualitative rather than quantitative entities. The decision table's reducts or approximations thereof are subsequently calculated through a reduction process. The reduct set might then be filtered down according to some criterion (e.g. according to their support basis or cost), and then overlaid the transformed data in order to generate a set of decision rules. The rule set might in turn also be subjected to some filtering scheme. The processing pipeline is displayed graphically in Figure 3.2.

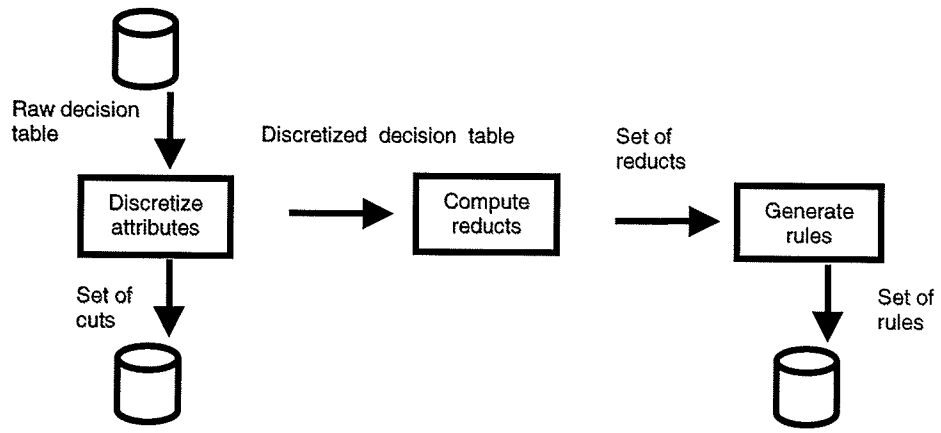

Figure 3.2 KDD process pipeline using rough sets

Fig. 3.3 illustrates how a small example KDD process can be executed by using e.g. the Rosetta system or the RSES system. For more information refer to [OKSS98], and [KPPS].

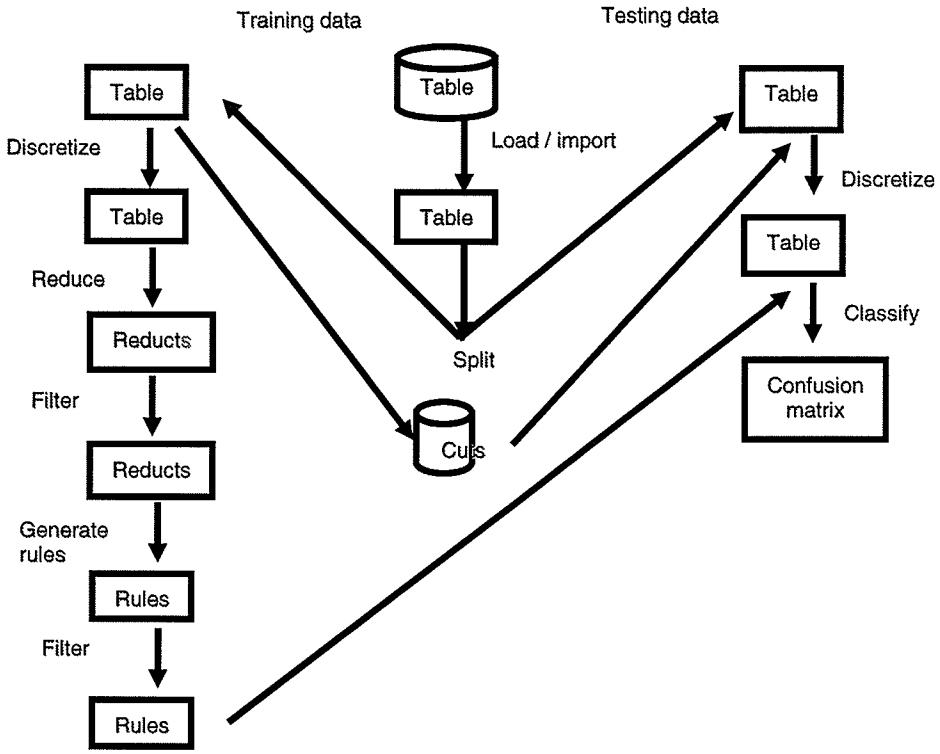

Figure 3.3 KDD process example using rough sets

3.5 Description of Methods Classifying Unseen Objects

We used the following four approaches for generating reducts and in the consequence for generating rules used further for classification of unseen, testing objects (data). The methods for generating reducts and then decision rules are following:

(1) Full reducts (FR).

(2) Object-oriented reducts (OOR).

(3) Genetic reducts (GR).

(4) Dynamic Reducts(DR).

Full reducts (FR) allows us to generate decision rules on the base of reducts relative to the system as a whole, i.e., minimal attribute subsets that preserve our ability to discern all relevant objects from each other. For further information refer to [Paw91]. Object Oriented Reducts (OOR) allows us to create decision rules using the reducts relative to a fixed object from the universe, i.e., minimal attribute subsets that preserve our ability to discern that object from the other relevant objects. For further information refer to [Paw91]. Genetic Reducts (GR) employs a genetic algorithm to compute full reducts or reducts related to each object from the universe. It usually allows computing approximate solutions only. For further information refer to [Wrob95]. Dynamic Reducts (DR) allows to compute reducts for random subsets of the universe of a given decision system and to select the most stable reducts, i.e., reducts that occur in most of the subsystems. This method employs an algorithm to compute full reducts or reducts related to each object from the universe or genetic reducts. These reducts, called dynamic reducts, are usually inconsistent for the original table, but the rules generated from them are more tolerant to noise and other abnormalities. For further information refer to [BSS94]. Decomposition Tree (DT) is based on the template notion (see Section 2.4). The template induces in natural way the split of original information system into two distinct sub tables containing objects that satisfy or not satisfy the template, respectively. Decomposition tree is a binary tree, whose every internal node is labeled by some template and external node (leaf) is associated with a set of objects matching all templates in a path from the root to a given leaf. For further information refer to [NgHo1999]. In the following sections, the notation *x-y*, where

$x, y \in \{\text{FR, OOR, GR, DR, DT}\}$, means that we use the combination of two methods x and y, described above.

4 Data Acquisition and Derived Features

Before performing a meaningful analysis of data, we are concerned about where the data came from and what was measured. The data studied in this thesis was acquired by conventional radar weather observation stations[1]. This acquired data is referred to as *raw radar data*. In order to get meaningful insight into the weather event from the raw radar data, post-processing analysis must be performed. Environment Canada currently uses software developed by InfoMagnetics Technologies Corporation to perform this post-processing. The software, known as Radar Decision Support System (RDSS), reads in the raw radar data and generates matched-cell files based on a continuity of statistical parameters of the raw data over a period of time and across a region of space. Although the intention of the creators of RDSS[2] is not to make the matched-cell files readily accessible to further post-processing, the data extracted from these matched-cell files provides the researchers with the wealthy information so that they apply advanced pattern recognition and further post-processing techniques to this data for the purpose of identifying more subtle relationships between the derived features and severe weather events.

---

[1] The data studied in this thesis was obtained from Environment Canada's radar stations in Broadview, SK, and Vivian, MB, between May 1997 and September 1999.

[2] RDSS was designed as a real-time weather radar interpreter, with the ability to read in historical, previously saved, raw radar data or matched-cell files viewing in the RDSS workspace. The RDSS workspace is a graphical user interface (GUI) that displays visualizations of weather events and tables of values of various derived feathers.

4.1 Weather Radar Data Acquisition

Radar is an acronym that stands for RAdio Detection And Ranging. Weather radars consist of a parabolic dish (it looks like a satellite dish) mounted on a tower of up to five stories tall. The radar emits a pulsed beam of microwave radiation (analogous to a radio signal, only it is pulsed rather than continuous, and the signal has a shorter wavelength than radio signal) in a particular direction, then receives energy reflected back from particles (usually water droplets, ice crystals or dust), then emits another burst, then receives again. The radar very rapidly switches from sending out the signal to listening for any returned signal to sending out the signal again in quick succession. When a burst of radar microwave radiation encounters a particle in the lower atmosphere, some of that energy is absorbed by the particle, the rest is scattered in all directions and some of it is scattered back in the direction of the source. The bigger the particle, or the better its scattering characteristics, the more energy is returned to the source. The energy received back at the source is measured, and the reflectivity factor (reflectivity is a measure of the fraction of radiation reflected by a given surface; defined as a ratio of the radiant energy reflected to the total that is incident upon that surface. Reflectivity factor z = the sum (over i) of (N_i * D_i^6), where N_i is the number of drops of diameter D_i in a pulse resolution volume. Note that z may be expressed in linear or logarithmic units. *The radar reflectivity factor is simply a more meteorologically meaningful way of expressing the radar reflectivity.) is calculated. The time between the transmitted and received signal is measured, and the distance to the particle is calculated. The weather radar data is collected with the above procedure. There are

two forms of weather radar data currently being gathered by weather offices around the world: conventional and Doppler radar. Conventional radar and Doppler radar are similar with respect to the fact that they both return information about how much of the transmitted signal is reflected back to the source and what is the distance range of the particle. Conventional radar differs from Doppler radar in that Doppler radar also uses the shifted frequency of the reflected signal to determine the radial velocity of the particle, toward or away from the radar station. Conventional radar has no information about radial velocity, thus movement of weather events has to be determined by applying post-processing analysis to the acquired raw radar data, such as with RDSS. Vivian, MB uses WSR81 C-band conventional radar with a 1.1-degree beam width and with a 5 cm wavelength and RDSS for post processing.

Radar gathers weather data by scanning 360° around the azimuth at a prescribed number of angles of elevation, which is known as a volume scans. This is illustrated in Figure 4.1 a, b, c for three angles of elevation respectively. In Figure 4.1 a, the radar station is at the center of the surface, at the vertex to the triangle. The triangle represents the beam of the antenna. The beam sweeps 360° azimuthally for one angle of elevation. There are two storm regions of dense particles represented by the ellipsoid volumes. As the beam sweeps through these volumes in Figure 4.1 b, it will record higher dBZ (a logarithmic expression for reflectivity factor, referenced to (1 mm^6 / 1m^3). dBZ = 10log(z/1 mm^6 m^3)) values than from the empty space around them. Figure 4.1 c demonstrates how, at a certain elevation, one storm is no longer detected. Also observable is the conical

region above the radar site from which no readings are taken. Each volume scan takes a certain amount of time to complete, which for Vivian and Broadview is five minutes. Within each volume scan, there is an entry for each angle of azimuth and elevation giving the dBZ value and the range (distance to reflector). In Vivian and Broadview stations, radars measure reflectivity using 1 km bins for a 120 km radius around the radar, 2km bins for 120-240 km. Because the volume scans occur every five minutes, there will be storm events that are recorded in several volume scans. Over the course of time, these storm cells may move, may join with another storm cell, may split into a number of storm cells, may increase in intensity or may dissipate and cease to be a storm. This activity will be related by the distribution of dBZ values within a volume scan, and how those values change from one volume scan to the next. In order to extract that information, post-processing analysis has to be applied the raw radar data contained in the volume scans. Figure 4.2 contains a screen shot from RDSS displaying a two-dimensional slice of volumetric radar scan. It also maintains a collection of derived products for each storm cell. This process is described in more detail in [WES99] and also explained on the website www.mit.edu. RDSS defines a storm cell to be a spatial cluster of high dBZ values. A storm cell snapshot is a collection of features of a storm cell, observed at a specific instance in time.

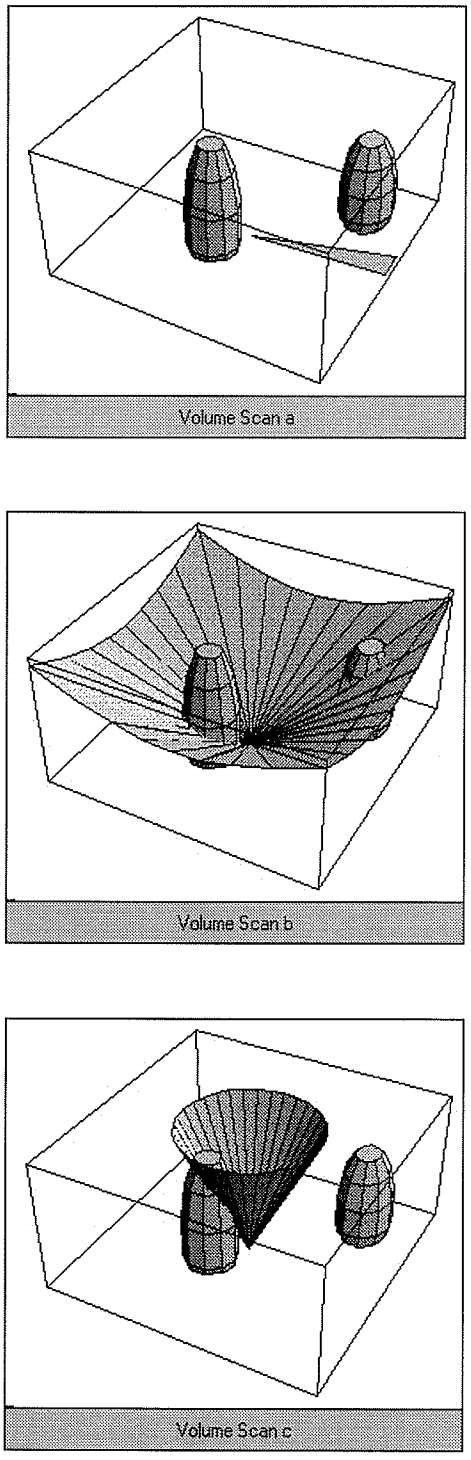

Figure 4.1 Volume scans in the vicinity of two storm events

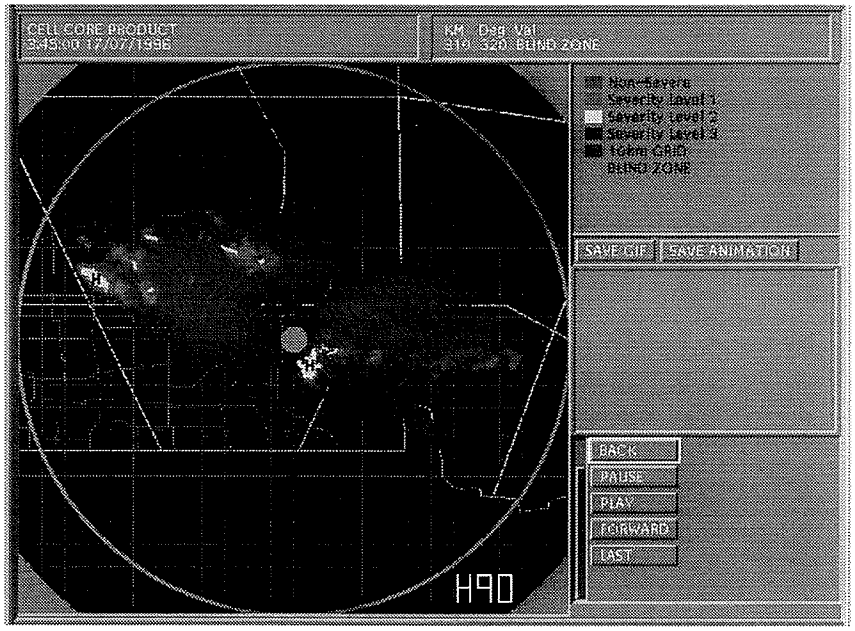

Figure 4.2 RDSS-generated two-dimensional slice of a volume scan

The above is further simplified as follow: the received signal is actually treated as though it reflects from a pulse resolution volume (a discrete radar sampling volume) of some density of particles. This density yields the radar reflectivity factor, z. The radar reflectivity factor is usually expressed in decibels as dBZ, referenced to "a unit of z per volume". Large dBZ values imply a high density of particles per unit volume, which increases the probability of heavy rain, hail or snow.

4.2 Raw Radar Data Post-processing

RDSS reads in the raw radar data and generates matched-cell files based on a continuity of statistical parameters of the raw data over a period of time and across a region of space. The key to understanding what RDSS does is to realize the difference between information gained from a direct measurement and information gained by manipulating the values obtained by direct measurement using proven or accepted (mathematical) relationships. A prime example is the difference between directly measuring the length of the hypotenuse of a right-angle triangle, or by calculating the length by applying the Pythagorean theorem to the lengths of the sides. The directly measured data is the raw radar data. This is the dBZ value of the reflectivity at a measured distance (range) for each angle of azimuth and elevation within the volume scan. These are called the *direct features* of a weather event. By applying a meteorological analysis to the dBZ values in the volume scans, RDSS is able to derive additional features of the weather event. These *derived features* give the meteorologist a different view of the information contained in the raw data, enabling them to make more accurate predictions about weather. Keep in mind that the derived features cannot yield more information than that contained in the raw data. In order to get "more information" a statistical analysis must be applied, relating a time-varying spatial distribution of dBZ values to what is most likely to occur. The likelihood of an event's occurrence may be determined empirically by relating ground observations to historical volume scans or it may be derived analytically through some known relationships of matter.

Thus, given the raw data, RDSS is able to derive some exact features of a weather event, and also some probabilistic features.

The first derived feature (sometimes called *product*) is the identification of storm cells based on dBZ values exceeding a chosen threshold. Indeed, since RDSS was intended as a real-time observation tool, if reflectivity values did not exceed a certain dBZ value, the forecast was "clear sky". Once storm cells are identified, RDSS can keep track of their growth and movement over time. The subsequent calculation of storm height, location and velocity are examples of derived exact features, and the probability of hail would be a derived probabilistic feature. There are numerous features derived by RDSS, some of them two-dimensional, but this thesis has been primarily concerned with 22 features, as listed in Table 4.1 and given in [APP99]. The boldface items are each a feature, where **R+** is the range of positive real numbers and **N+** is the range of positive real integers. A range of {0, 1, 2, and 3} indicates that the feature takes on one of those values. These derived features will be discussed more detail in Section 2.4.

As RDSS calculates the derived features of a weather event, it identifies the growth, movement and dissolution of storm cells. This process can identify several storm cells in a single volume scan, and then track the lifetimes of these storm cells from the first volume scan, where it is identified, to the last volume scan, where it has dissolved to the point where the dBZ values have fallen below the threshold. This information is grouped together by RDSS and output as a ***matched-cell file***. Table 4.2 shows the file naming convention used by RDSS for matched cell files. The first three letters of the filename identify the weather radar station, WVJ for

Vivian and WIK for Broadview. The first set of numbers give the year, month, day and UTC time of the start of the matched-cell file (UTC – universal time clock, or Greenwich mean time). The second and third sets of numbers give the latitude and longitude, respectively, as degrees-minutes-seconds.

Table 4.1 List of groups describing 22 derived features

Derived Features	**Range**
Height offset (km)	**R+**
Extend	{**N+, N+**}
Core Volume	**N+**
Core Height	**N+**
Supercell Severity	{0,1,2,3}
Wind Gust Severity	{0,1,2,3}
Hail	{0,1,2}
Core Tilt Angle	**R+**
Supercell Flag	{0,1}
Join Count	{0,1}
Split Count	{0,1}
Core Tilt Vector	{**R, R, R**}
Velocity Set Flag	{0,1}
Velocity (km/h)	{**R, R, R**}
Core Size	{**N+, N+**}
Orientation	**R+**

Table 4.2 Filename convention for RDSS matched-cell files

Matched-Cell Filenames, as saved by RDSS	**Radar Location**
WVJ_MATCHED_CELL_199707120605_502308N_992202W	Vivian,
WVJ_MATCHED_CELL_199707242025_493110N_980447W	Vivian,
WIK_MATCHED_CELL_199807081915_504027N_104503W	Broadview

Table 4.3 shows the first few lines of a matched-cell file, displayed as text. The files are saved as unsigned characters; an ASCII text viewer cannot interpret most of the file. It just so happens that the range of ASCII text characters is a

subset of the range of unsigned character values, so whatever information saved in the matched-cell file that was intended as text is viewable, but not all of the saved data in the file is intended ASCII. A key to understanding these files can also be found as the last three pages of the RDSS user's manual. All of the fields are delimited with the | character, and are easily accessible. These files are used to extract the data for further analysis. In order to make the matched-cell files readily accessible to further post-processing, we extract the data from these matched-cell files by the modified project EC-NRC program files [DIE00].

Table 4.3 Partial listing of the contents of a matched-cell file

WVJ_MATCHED_CELL_199707120605_502308N_992202W
1|6|3|13|9|Vivian|WVJ|CELL SNAPSHOT|13.000000|_49.883057|-96.449997|0.290000|6371.100098|18|6:05:00 12/07/1997|47|7.000000|50|10|2|-207.047821|55.939224|6.243376|5|6|75|8.000000|8|0|0.000000|0.000000|0|0.000000|0.000000|0|0.000000|0.000000|15|-207.133331|57.533333|14|-207.000000|57.500000|23|-205.869568|57.130436|7|-206.142853|57.285713|8|-206.000000|57.250000|0|0|0|3.500000|4.000000|104.036240|1|0|0|0.297347|-0.074337|0.951871|0|0.000000|0.000000|0.000000|-206.000000|57.166668|3.572447|2.138433|2.895067|6|3|17|9|Vivian|WVJ|CELL CORE PRODUCT|13.000000|_49.883057|-96.449997|0.290000|6371.100098|18|6:05:00 12/07/1997|47|7.000000|50|10|-207.047821|55.939224|6.243376|25|26|0|0|0|25|26|_____

4.3 Analysis Data Acquisition

The severe weather events associate with the weather features. Having a list of desired features and an observed events table, we generate data for further analysis by cross-referencing the list of the events table with all of the matched–cell files based on the time and location of the events. The cross-reference process is realized by Matlab® program and output is a text file. The data for further analysis are generated by C++ program StormIT [LPP99] that uses the Matlab® program's output text file as input. The analysis data output is a text file of 23 columns whose first 22 columns are the 22 features listed in Table 4.1 and 23rd column is the event classification. The output files can be imported into the ROSSETA system [ROSET], the RSES system [RSES] and other platforms for the subsequent analyses.

4.3.1 Cross-reference of Matched-cell and Ground-truth

In order to corroborate a list of features (Table 4.1) to a weather condition, one must have reference to a list of observed events. Such a list is called a ground-truth table. Ground-truth tables are compiled by Environment Canada as they receive reports from weather stations and volunteers. Table 4.4 is a part of the Ground-truth table. The ground-truth events are subdivided into categories, such as hail, rain, tornado, and wind and Table 4.4 belongs to hail. Indeed, one of the biggest hindrances to this project is the lack of ground-truth observations compared to the number of matched-cell files indicating that there may be an observable event. We cross-reference the list of the events table with all of the matched–cell files based on the

time and location of the events. This means choosing a suitable "window" in time and space to which the both the ground-truth event and matched cell files belong. Usually, one or more matched-cell files pertain to each ground-truth observation. Sometimes, the same matched–cell file may be associated with more than one ground-truth event – hail & rain, for example. Matching the ground-truth event to the specific cell-snapshot within the matched- cell file could minimize this.

Table 4.4 Ground-truth Event Table (Hail)

The latitude/longitude for multiple locations is the average of each location's latitude/longitude.

Y	M	D	TIME (CDT) unless otherwise indicated	LOCATION	REGION	LAT	LONG	COMMENTS
1997	05	7	18:15	McCreary	Dauphin	50.77	-99.49	pea
1997	06	1	16:20	Snowflake	Brandon	49.04	-98.66	Marble ("dime to nickel"). Quite a bit fell during the 10 minutes, winds were strong at the time
1997	06	1	18:05	Winnipeg - Fort Richmond	Red River	49.79	-97.14	1 cm
1998	05	13	20:00	Killarney		49.18	-99.66	pea size
1998	05	13	15:45	Ste.Rita/Elma		49.88	-96.59	golfball size
1998	05	13	16:20	St Adolphe		49.67	-97.11	Egg size.
1999	05	01	19:03 - 19:10	Altona		49.10	-97.56	Severe hail; slightly larger than quarters (30 mm)
1999	05	03	19:40	3 miles W of #8 on highway 17		50.39	-97.11	Marble hail (12 mm) covering ground completely; also rain
1999	05	04	11:00 - 12:00	1 mi. E of Argyle on hwy 322		50.18	-97.45	Severe hail (40 mm); lasted 1 hour; also funnel

The cross-reference of matched-cell filenames with ground-truth tables can be done manually for small sets, but Matlab® programs are used for larger sets of data. The basic procedure is as follows:

(1) Have the ground-truth information listed with the latitude and longitude of the event, the date and time of the event and the storm classification integer. Table 4.5 gives an example.

Table 4.5 Ground-truth information list

Latitude	Longitude	Date	Time	Storm Type
51.23	-101.35	26-May-99	17:35	1
50.13	-97.53	27-May-99	23:45	2
50.13	-97.53	27-May-99	23:45	4
49.2	-98.88	06-Jun-99	0:01	3

(2) Have a directory listing of all filenames that pertain to the time range of ground-truth events (months or years). Copy these files into the Data directory.

(3) Each matched-cell filename has the date, time and geographical location embedded in it. For every ground-truth event, you look for all the matched-cell filenames that are "close enough" in time and space to the ground-truth event. When you find one, you put that filename and classifier in one text file. This text file contains the matched-cell filenames and storm classification. The first line of the text file is an integer indicating how many subsequent lines of filenames follows. These rows of filenames and storm classifiers are two-column tab delimited. For example, the contents of this text file could be shown as Figure 4.3, or view it as two columns given by Table 4.6.

```
5
WIK_MATCHED_CELL_199807111015_511145N_1052407W 1
WIK_MATCHED_CELL_199807111050_512336N_1051509W 1
WIK_MATCHED_CELL_199807081915_504027N_1045035W 2
WIK_MATCHED_CELL_199807050310_500015N_1034313W 3
WIK_MATCHED_CELL_199807052320_491414N_1025112W 3
```

Figure 4.3 an example of the input file

Table 4.6 Content of the input file

Column 1: integer in row 1, Filename otherwise	Storm Classification based on Ground-truth
5	
WIK_MATCHED_CELL_199807111015_511145N_1052407W	1
WIK_MATCHED_CELL_199807111050_512336N_1051509W	1
WIK_MATCHED_CELL_199807081915_504027N_1045035W	2
WIK_MATCHED_CELL_199807050310_500015N_1034313W	3
WIK_MATCHED_CELL_199807052320_491414N_1025112W	3

The storm classification is of the set {1,2,3,4} where 1-hail, 2- rain, 3-tornado or 4-wind. This file listing shows first that there are 5 rows of data following the first line, and then lists each matched-cell filename and its storm classifier, separated by a 'tab' character. We see that there are three different storm classes applied to five different matched-cell files. Each matched-cell file will have a number of cell snapshots within it. If, for instance, there were an average of 5 cell snapshots per matched-file, then the output would have 25 rows of 23 columns. The 23rd column contains the storm classification as applied to the matched-cell file in Table 4.6. Because very few filenames will have embedded times and locations exactly matching a ground-truth event, "close enough" is a variable adjusted by the researcher, but may reasonably be 10 minutes, 0.25° latitude and 0.45° longitude. This means that any matched-cell filename that is within ±10 minutes, ±0.25° latitude and ±0.45° longitude of the ground-truth event will be assigned the storm classification number of that ground-truth event. Converting to kilometers, 0.5° latitude is approximately 55.66km and 0.9°

longitude is ~64km, which means that any matched-cell file pertaining to an event within a ~30km radius of the ground-truth event will be classified to that ground-truth event, given it is within the corresponding time frame. In a time frame of 20 minutes, there may be several matched-cell files within a 30km radius, so they will all be classified to that ground-truth event. Also, there may be another ground-truth observation at nearly the same time and place, but for a different storm type. Thus filenames that were previously categorized as one storm type will now be added to the list as another storm type. This is reasonable, since a region may get heavy rain and nearby hail. For two ground-truth observations of the same event, one will report rain and the other hail. However, the closer the ground-truth event's can be aligned to matched-cell files, the better the subsequent classification strategies will perform. (When matching ground-truth event's, remember that the filenames have UTC time format and the ground-truth event may use a local time.)

(4) Having listed all the filenames and classifiers, separated by a tab, count the rows and put that integer number on the first row. Save the list with a name of your choice, for instance, *gtelist.txt*.

(5) If you have only one list, open the file *matchedCellFilesList.txt* and put 1 on the first row, and then the filename of the list file, in this case *gtelist.txt*. If you've generated several ground-truth list files, put the number of files on the first line, and list the filenames on the lines below. For example, the contents of *matchedCellFilesList.txt* are shown Figure 4.4. Save these files,

matchedCellFilesList.txt and *gtelist.txt*, wik98.txt and wvj97.txt in the /DataKey directory. Ensure that the \Data directory contains the matched-cell files. These files are ready to be used by the program StormIT.exe.

```
3
..\DataKey\gtelist.txt
..\DataKey\WIK98.txt
..\DataKey\WVJ97.txt
```

Figure 4.4 the contents of *matchedCellFilesList*

4.3.2 Automatically Cross-Referencing by Matlab Programs

The process described in section 4.3.1 above is suitable for small data sets, or for optimizing the storm type assignments manually. However, the algorithm lends itself to simple implementation as a program, and one has been developed for the Matlab® environment. For the analysis of RDSS data from May – September 1999, there are 4641 matched-cell files and 150 ground-truth observations. A time and distance range of ±10 minutes, ±0.25° latitude and ±0.45° longitude of the ground-truth event are used, but are variable. To do this cross-reference manually would be a formidable task. When the ground truth events and matched-cell files are cross-referenced using the program, there are 92 files assigned storm types from the 150 ground truth events. When only the unique assignments are extracted – meaning that duplicate occurrences of a filename with the same storm type are removed – there are 78 entries for the input data file *gtelist.txt*. The key m-files

(Matlab® program scripts) to perform the ground truth events cross-referencing are listed and described in Table 4.7.

Table 4.7 M-files used to ground-truth filename lists

M-file name	Description
Getfilenames.m	Reads in a directory listing of all of the data files in master data directories and stores the names in a variable named *filenames*.
Readgrndtruthdat.m	Reads in the ground-truth data: [lat,long,timestamp,type] from a text file named *dat2import.txt*, and creates a variable *grndtruthdat*, which lists info column wise as: datenums lat long type.
Matchgrndtruth.m	Takes the variable *filenames* and parses each filename to obtain the embedded date, time, latitude and longitude and saves it in a variable, *RDSSdat*. For each ground-truth event listed in *grndtruthdat*, all members of *RDSSdat* that are within the time-space window are indexed and assigned the storm type of the ground-truth event. They are stored as variable *FileIdx*. The index points to the row of *filenames* where the matched-cell filename is stored. Only unique classifications are needed, so just the unique data rows of *FileIdx* are extracted to variable *UniqueFiles*. The filenames indexed by *UniqueFiles* are written to a text file with the storm type integer. That text file is the input file, *gtelist.txt*, to get named in *matchedCellFilesList.txt*
Writenames.m	Writes the input file list, *gtelist.txt*

There are some specific details relating to the locations of the files. Firstly, it requires that the RDSS matched-cell files (obtained from Environment Canada, and referred to as the master files) are located on hard-drive C at C:\EC_DATA\ in one of four subdirectories, \S0, \S1, \S2, or \S3. These subdirectories relate to the storm severity rating, {0, 1, 2, 3}. The data will be received from Environment Canada in compressed zip files, which should be extracted to the corresponding directories. There do not need to be any more subdirectories under \S0, \S1, \S2, or \S3 since

the filenames are all unique. This structure allows easy access to storm files of a particular severity. It is also assumed that the main project directory is C:\EC within which will be other subdirectories pertinent to the project. To run the m-files to generate the cross-reference output file, at the Matlab® command prompt enter each m-file name in the order given in Table 4.7, or simply type **ec01** and enter, which runs all four programs. If we want to change the time-space window, find the variables dt, dlat & dlong in the m-file, *Matchgrndtruth.m* and change their value. The ground-truth tables come from Environment Canada in Excel spreadsheet format. The ground-truth events need to be saved in a text file named *dat2import.txt* as four columns of data: latitude, longitude, timestamp and type. Latitude and longitude should be in decimal format, and the timestamp needs to be in "DD-MMM-YYYY HH:MM" format (i.e. 26-May-1999 17:35). The storm type integer is in the fourth column.

To summarize, the files and data should be organized and the program should be run as follows:

(1) Have a directory C:\EC for work and C:\EC_DATA for the master matched-cell files.

(2) Within C:\EC have a directory for the StormIT feature extraction program (will be discussed in 4.3.3), i.e.: C:\EC\STORMIT

(3) Within C:\EC\STORMIT have three subdirectories:

C:\EC\STORMIT\DATA the location of the RDSS files (copied from C:\EC_DATA)

C:\EC\STORMIT\DATAKEY the location of

matchedCellFilesList.txt and *gtelist.txt*, etc.

C:\EC\STORMIT\RUNS location of *StormIT.exe* program file.

4.3.3 Generating the Output Data Files

There is a collection of C and C++ source files grouped together in a MS Visual C++ project named *StormIT* (for Storm Information Technology) by previous researchers. When run, the program, named *StormIT.exe*, will prompt for a number, 2 or 4, to indicate the number of storm classes to consider. If the user chooses 2, only hail and tornado occurrences will be considered. If the user chooses 4, then hail, rain, tornado and wind will be considered. Once the user makes that selection, the program looks for the text file *matchedCellFilesList.txt* first in the directory ..\DataKey\, then read in input files, such as *gtelist.txt*, *WIK98.txt* and *WVJ97*.txt, in the directory given by the relative path ..\DataKey, and then accesses the matched-cell files in a different directory and outputs all of the data to the screen. In order to get the data into a text file, the screen output must be piped into a text file. StormIT.exe runs from a DOS command prompt on Windows95/98/NT systems. Open a DOS command box, go to C:\EC\STORMIT\RUNS and Use the following command at the prompt: ***stormIT > output.txt.*** The screen will pause while the program waits for your choice of 2 or 4, but you will not get any visual prompt. The prompt line gets copied into the output file. Just enter 2 or 4 (usually 4) and press enter. The output will be written into the text file, *output.txt*, in the current directory. After the program *stormIT* stops, open *output.txt* and delete the first line, which will be the prompt for you to press 2 or 4, and then save the file. The derived features are now saved as 23 tab-delimited columns in the text file.

In order to run properly, *StormIT* and the input and data files must be organized with a particular directory structure (as described previously in section 4.3.2). Inside the parent directory C:\EC\STORMIT, create a directory with an arbitrary name (such as \runs) and copy *stormIT.exe* into that directory. Next, in the same parent directory, create two more subdirectories, specifically named \Data and \DataKey. The matched-cell data files pertaining to the analysis will go in the \Data directory, and the input files will go into the \DataKey directory.

4.4 Derived Features

RDSS considers a storm cell to be a spatial cluster of high radar reflectivity values. High values are considered to be any value over a set threshold. A cluster of high reflectivity values is considered to be a collection of samples that exceed this threshold and where each samples is physically adjacent to some other sample that belongs to the same cluster. The above is defined as storm cell reflectivity core by RDSS. RDSS maintains a collection of storm cells, obtained from volumetric radar scans, that meet a minimum reflectivity threshold (47dBZ) that is indicative of storm severity. Within RDSS, a storm cell snapshot is considered to be a collection of features, measurements, and data products that occur within the immediate vicinity of a storm cell reflectivity core, for a specific moment in time. A matched storm cell is a set of cell snapshots that have been identified as belonging to the same reflectivity core as it exists over time, plus additional rate of change measurements and event indicators (mergers and splits) that can be produced from the associations made. RDSS groups storm cells that are likely to be the same cell

at different durations of the storm, which is called a matched group. RDSS provides the meteorologist with a graphical view of a single storm cell or an entire matched group of storm cells, along with useful information associated with them.

Table 4.8 lists the twenty-two derived features that are used to identify a storm cell for classification. The table shows the data type, the description and the C++ variable used in program StormIT for the features. The height offset and extent features give the size of the storm cell in kilometers. The core volume feature gives the volume of the cell core. The core height feature gives the height of the core. The supercell severity, wind gust severity and hail occurrence features are all heuristically determined. The supercell severity and wind gust severity denote the severity of a supercell and its wind gust potential. The supercell severity and wind gust severity are determined by the size of the cell, volume and vertically integrated liquid (VIL) using heuristically determined thresholds. They may take on values from {0,1,2,3} where 3 indicate the greatest severity. Cells that contain bounded weak echo regions (BWER) will normally have a supercell severity value of 3. A BWER is the region beneath the extensive mid-level overhang echo. A BWER identifies the location of strong or intense updraft. Figure 4.5 shows a two dimensional view of a BWER. Cells that contain either a high downdraft potential and a descending core, or a high velocity, a moderate downdraft potential and a descending core, will be assigned a wind gust severity value of 3. The hail occurrence feature may take on values from {0,1,2}. It estimates the probability of hail larger than 0.75 inches in diameter: a value of 2 implies high probability and a value of 0 implies very low probability. The core tilt angle feature gives the tilt

angle of the storm cell, calculated by fitting a vector through the slice centers and measuring the angle separating the vector from the vertical vector fitted through the sliced centroilds. The supercell flag is a Boolen flag denoting if the cell is supercell or not (0,1). The join count denotes the number of cells RDSS has computed to have joined to make this cell. The split count feature is a Boolean value denoting whether or not a cell split occurred to form this cell. The tilt vector represents the tilt vector fitted through the sliced centroids. The velocity set flag determines if the next field (the velocity vector) is valid or not. The velocity vector denotes the velocity of the storm in km/h. The core size is the size of a slice of the cell. The orientation feature denotes the angle between the horizontal and the major axis of an ellipse fitted to the cell core. An important point to keep in mind is that the affirmation of heuristic criteria by a cell does not necessarily determine the cell's storm type. For example, a cell may be labeled as hail even though its hail probability is zero. Such inconsistencies may occur due to poor sampling or class labeling of cells. Figure 4.6 illustrates some of the above information, as computed by RDSS. These features are all scalar values, although some derived features returned by RDSS, such as vertically integrated liquid (VIL), vertically integrated reflectivity (VIZ), overhang and any of the calculated gradients are two-dimensional arrays of values. These arrays span an *x-range* and *y-range* for a given fixed height, *z*. To date, the two-dimensional derived features are an untapped potential for further pattern recognition analysis.

The sampling rate of cells is once every five minutes with the reflectivity values used spread out in time over the five minutes interval as the radar adjusts its

azimuth. This is sufficient to capture the dominant features in most heavy rain and wind events; hail and tornado events have substantially shorter life cycles and features may not be clearly depicted. Table 4.9 describes the temporal and spatial sample rates of the various types of storms. After the cells have been collected and matched cell files have been generated, they are cross-referenced according to a list of weather events verified by meteorologists or ground observers. We can corroborate a list of these features to a weather condition and identify relationships between the derived features and severe weather.

Table 4.8 derived features listed with data type, description and variable

Feature	Type	Description	C++ Variable
1. z location	Real	Height offset [km]	TheZValue
2. x size	Integer	x extent of cell [km]	TheXValue
3. y size	Integer	y extent of cell [km]	TheYValue
4. volume	Integer	Cell core volume [km^3]	TheCoreVolume
5. height	Integer	Height of cell core	TheCoreHeight
6. supercell severity	0-3	Heuristic	TheSupercellSev
7. wind gust severity	0-3	Heuristic	TheWindGustSev
8. hail probability	0-2	Heuristic	TheHailProb
9. cell tilt angle	Real	core fitted angle	TheTiltAngle
10. supercell flag	0-1	Heuristic	IsSupercell
11. join count	Integer	Number of cells joined	TheJoinCount
12. split count	0-1	Has cell split ?	TheSplitCount
13. tilt vector x–coord	Real	centroid parameter	TheTiltX
14. tilt vector y–coord	Real	centroid parameter	TheTiltY
15. tilt vector z–coord	Real	Centroid parameter	TheTiltZ
16. velocity set flag	0-1	Next field available	IsVelocityFlagSet
17. x velocity	Real	[km/h]	TheVelX
18. y velocity	Real	[km/h]	TheVelY
19. z velocity	Real	[km/h]	TheVelZ
20. a size	Integer	Cell core size	TheCoreX
21. b size	Integer	Cell core size	TheCoreY
22. orientation	Real	Vector	TheOrientation

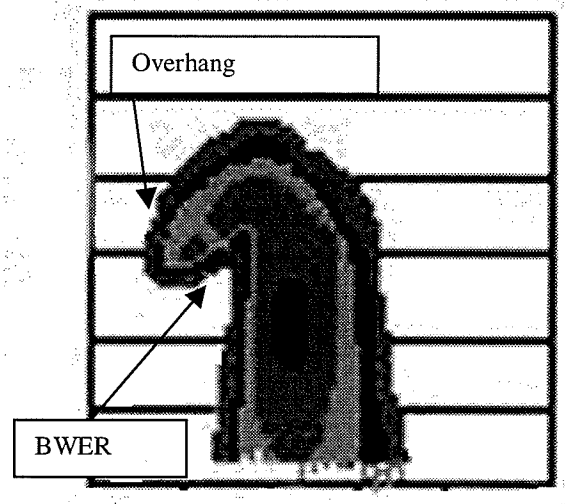

Figure 4.5 Two-dimensional vertical slice of a storm cell with a BWER

Figure 4.6 RDSS-generated storm cell information

Table 4.9 Temporal and spatial sampling rates

Weather Event	Temporal Scale (min)	Spatial Scale (km)
Heavy Rain	5-60	~ 10
Hail	5-10	~ 5
Tornado	5-10	~0.5X0.5
Gust Front	5-30	~ 1X30
Shear Zone	5-30	~ 1X10
Mesocyclone	10-30	~ 4
Convergence/Divergence	10-60	~5X20

5 Methodology and Experimental Results

5.1 Methodology

There are a considerable number of identical cells (that is, the 22 features used to describe them have the same values) with two different labels. This happened for about 25% of the cells. There are 146 cells with two different labels and 431 cells with only one label. This indicates the uncertainty of the label assignments. To solve this problem we have added six more classes to the four originals (hail, rain, tornado, wind) in order to cover those cells with two labels. For both the 4-classes decision and the 10-classes decision, the data have been randomly re-sampled into the training set and the testing set. 75% of the data will form the training set and remaining 25% form the test set due to following two reasons: (1) our data set is not large (enough large training data set is needed to discovery the hidden patterns that represent the knowledge in the whole data); (2) we have the same resample conditions in the paper [RPP00] so that we can compare the results obtained with different methods. This re-sampling was performed 5 times over the whole data set in order to obtain 5 groups of training and testing data sets for each type of classification. The experiments have also been tried on the data (50% of the data for the training data and 50% of the data for the testing data) and the similar results can be gained individually. But the stability of the results for 75% and 25% seems to be a bit better than that of the results for 50% and 50%. It indicates that enough data set should be provided. The mean pattern distribution for five groups for the 4-class and 10-class decision is shown in Table 5.1 and 5.2, respectively.

Table 5.1 Object distributions of the data with 4 decision values

Decision class number	Class name	Number of objects	Number of training objects	Number of testing objects
1	Hail	166	126	40
2	Rain	54	36	18
3	Tornado	265	207	58
4	Wind	92	64	28
	Total	**577**	**433**	**144**

Table 5.2 Object distributions of the data with 10 decision values

Decision class Number	Class name	Number of objects	Number of training objects	Number of testing objects
1	Hail	150	112	38
2	Rain	22	17	5
3	Tornado	207	155	52
4	Wind	52	46	6
5	Hail or Rain	20	16	4
6	Hail or Tornado	10	6	4
7	Hail or Wind	0	0	0
8	Rain or Tornado	33	25	8
9	Rain or Wind	33	23	10
10	Tornado or Wind	50	33	17
	Total	**577**	**433**	**144**

5.2 Experimental results

In this section, the results of experiments, which were performed independently for the 4-classes decision and 10-classes decision over the 5 randomly re-sampled groups, are presented. The results presented here include:

(1) The average over the 5 groups for the training and testing confusion matrices, respectively.

(2) The performance evaluation of the used classifiers. The performance evaluation is represented by two coefficients: the average accuracy for the Rosetta system, and accuracy for the RSES system.

(3) The classification error for the training and testing data in the case of the results get from the Rosetta system.

The labels of the confusion matrices are:

Actual: the actual value of decision for tested objects.

Predicted: the predicted value of decision by classifier for tested objects.

1: Hail
2: Rain
3: Tornado
4: Wind
5: Hail or Rain
6: Hail or Tornado
7: Hail or Wind
8: Rain or Tornado
9: Rain or Wind
10: Tornado or Wind

5.2.1 ROSETTA Analysis

This section will give the results from ROSETTA analysis with confusion matrix that summarizes the performance of a classifier k, applied to the objects in an information system S. According to definition of confusion matrix in section 3.3, we assume that v_d is the set of integers {1, 2, 3, 4} for 4-classes decision or {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} for 10-classes decision. The entry C(i, j) counts the number of objects that really belong to class i, but were classified by k as belonging to class j. Accuracies have been estimated from the confusion matrix C by the formula in section 3.3 and listed in the last column and last row separately.

Table 5.3 gives the classification results of the training set with 4-decision values by object oriented reduct method. The accuracy for the class 2 is only 0.417. The reason for this low accuracy is mainly due to the uncertain label for this class. The total objects of the class 2 (including 36 training objects and 18 testing objects) are 54 among which there are 32 objects with two labels and there are only 22 objects with one label. The proportion of the number for the objects with two labels and total number of the objects is 59%. The accuracy for the class 4 is 0.781. The total objects of the class 4 (including 64 training objects and 28 testing objects) are 92 among which there are 40 objects with two labels and there are 52 objects with one label. The proportion of the number for the objects with two labels and total number of the objects is 43%. The accuracy for the class 4 is higher than that for the class 2. Similarly, for class 1 the proportion of the number for the objects with two labels and total number of the objects is 9% and for class 3 the proportion is 21%. Therefore, the accuracies for classes 1 and 3 are higher. This fact indicates

that the uncertainty of the label assignments causes the inaccuracy of the classification results. This kind of uncertainty also exists in tables 5.4 – 5.9.

Table 5.3 OOR confusion matrix for 4-decision training table

		Predicted				
A		**1**	**2**	**3**	**4**	
c	**1**	126	0	0	0	1.0
t	**2**	4	15	8	9	0.417
u	**3**	3	0	204	0	0.986
a	**4**	0	0	14	50	0.781
l		0.947	1.0	0.903	0.847	0.912

Table 5.4 gives the classification results of the training set with 4-decision values by GR-OOR method. The purpose of combining the GR method with OOR method is to find a better method to classify the objects and improve the classification performance. The classification results for the classes 1 and 3 remain the same. The classification results for the classes 2 and 4 have changed. But the whole probability remains 0.912.

Table 5.4 GR-OOR confusion matrix for 4-decision training table

		Predicted				
A		**1**	**2**	**3**	**4**	
c	**1**	126	0	0	0	1.0
t	**2**	4	17	8	7	0.472
u	**3**	3	0	204	0	0.986
a	**4**	0	2	14	48	0.75
l		0.947	0.895	0.903	0.873	0.912

Table 5.5 provides us with the classification results of the training set with 4 decision values by FR method. The classification results for the classes 1 and 3

remain the same. The classification results for the classes 2 and 4 have improved. But the whole probability remains 0.912.

Table 5.5 FR confusion matrix for 4-decision training table

		Predicted				
A		**1**	**2**	**3**	**4**	
c	**1**	126	0	0	0	1.0
t	**2**	4	19	8	5	0.528
u	**3**	3	0	204	0	0.986
a	**4**	0	4	14	46	0.719
l		0.947	0.826	0.903	0.902	0.912

Table 5.6 gives the classification results of the testing set with 4-decision values by OOR method. The probabilities for each class are less than that of the correspondence in table 5.3. The reasons for this kind of low probability are mainly due to the small amounts of the data and incompleteness of the data. The same situations exist in the tables 5.7 – 5.9.

Table 5.6 OOR confusion matrix for 4-decision testing table

		Predicted				
A		**1**	**2**	**3**	**4**	
c	**1**	32	0	7	1	0.8
t	**2**	6	5	5	2	0.278
u	**3**	5	0	52	1	0.897
a	**4**	0	0	12	16	0.571
l		0.744	1.0	0.684	0.8	0.729

Table 5.7 gives the classification results of the testing set with 4-decision values by GR-OOR method. This method improves the results for classes 1 and 3.

Table 5.7 GR-OOR confusion matrix for 4-decision testing table

		Predicted				
Actual		**1**	**2**	**3**	**4**	
	1	35	0	4	1	0.875
	2	6	5	5	2	0.278
	3	4	0	54	0	0.931
	4	2	2	10	14	0.5
		0.745	0.714	0.740	0.824	0.75

Table 5.8 gives the classification results of the testing set with 4-decision values by FR method. The accuracies for each class are less than that of the correspondence in table 5.5. It indicates that the rules hidden in the training data cannot completely represent knowledge in the testing data.

Table 5.8 FR confusion matrix for 4-decision testing table

		Predicted				
Actual		**1**	**2**	**3**	**4**	
	1	23	0	6	1	0.575
	2	6	5	3	4	0.278
	3	5	1	42	2	0.724
	4	5	4	5	11	0.393
		0.590	0.5	0.75	0.611	0.563

Table 5.9 gives the classification results of the testing set with 4-decision values by DR-OOR method. The probability for the class 2 is 0.0. It reaffirms that affects of the uncertainty of the label assignments.

Table 5.9 DR-OOR confusion matrix for 4-decision testing table

		Predicted				
Actual		**1**	**2**	**3**	**4**	
	1	26	0	13	1	0.65
	2	7	0	9	2	0.0
	3	3	0	55	0	0.948
	4	0	0	17	11	0.393
		0.722	Undefined	0.585	0.786	0.639

Table 5.10 gives the classification results of the training set with 10-decision values by OOR method. The accuracies have been improved due to solution of the uncertainty of the label assignments. Tables 5.11 – 5.15 have the similar effects.

Table 5.10 OOR confusion matrix for 10-decision training table

		Predicted										
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
Actual	**1**	112	0	0	0	0	0	0	0	0	0	1.0
	2	0	17	0	0	0	0	0	0	0	0	1.0
	3	0	0	155	0	0	0	0	0	0	0	1.0
	4	0	0	0	46	0	0	0	0	0	0	1.0
	5	0	0	0	0	16	0	0	0	0	0	1.0
	6	0	0	0	0	0	6	0	0	0	0	1.0
	8	0	0	0	0	0	0	0	25	0	0	1.0
	9	0	0	0	0	0	0	0	0	23	0	1.0
	10	0	0	0	0	0	0	0	0	0	33	1.0
		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0

Table 5.11 gives the classification results of the testing set with 10-decision values by OOR method.

Table 5.11 OOR confusion matrix for 10-decision testing table

		Predicted										
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
Actual	**1**	24	0	12	2	0	0	0	0	0	0	0.632
	2	0	5	0	0	0	0	0	0	0	0	1.0
	3	7	0	38	0	0	0	0	2	0	5	0.731
	4	1	0	3	2	0	0	0	0	0	0	0.333
	5	0	0	0	0	4	0	0	0	0	0	1.0
	6	0	0	0	0	0	4	0	0	0	0	1.0
	8	0	0	0	0	0	0	0	8	0	0	1.0
	9	0	0	0	0	0	0	0	0	10	0	1.0
	10	2	0	3	0	0	0	0	0	0	12	0.706
		0.706	1.0	0.679	0.5	1.0	1.0	0	0.8	1.0	0.706	0.743

Table 5.12 gives the classification results of the testing set with 10-decision values by GR-OOR method.

Table 5.12 GR-OOR confusion matrix for 10-decision testing table

Actual	Predicted										
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
1	22	1	7	3	0	0	1	1	1	2	0.579
2	0	5	0	0	0	0	0	0	0	0	1.0
3	10	1	35	2	0	0	2	2	0	0	0.673
4	0	0	4	2	0	0	0	0	0	0	0.333
5	0	0	0	0	4	0	0	0	0	0	1.0
6	0	0	0	0	0	4	0	0	0	0	1.0
8	0	0	0	0	0	0	8	0	0	0	1.0
9	0	0	0	0	0	0	0	10	0	0	1.0
10	3	0	2	0	0	0	0	0	12	0	0.706
	0.629	0.714	0.729	0.286	1.0	1.0	0	0.727	0.769	0.923	0.708

Table 5.13 gives the classification results of the testing set with 10-decision values by FR method. Udf means the decision value is undefined.

Table 5.13 FR confusion matrix for 10-decision testing table

ACTUAL	Predicted											
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Udf**	
1	13	0	4	2	0	0	0	0	0	0	19	0.342
2	0	5	0	0	0	0	0	0	0	0	0	1.0
3	5	3	27	1	0	0	0	1	0	1	14	0.519
4	0	0	2	3	0	0	0	0	0	0	1	0.5
5	0	0	0	0	4	0	0	0	0	0	0	1.0
6	0	0	0	0	0	4	0	0	0	0	0	1.0
8	0	0	0	0	0	0	0	8	0	0	0	1.0
9	0	0	0	0	0	0	0	0	10	0	0	1.0
10	0	0	0	0	0	0	0	0	0	15	2	0.882
	0.722	0.625	0.818	0.5	1.0	1.0	0	0.889	1.0	0.938	0.0	0.618

Table 5.14 gives the classification results of the testing set with 10-decision values by DR-FR method.

Table 5.14 DR-FR confusion matrix for 10-decision testing table

		Predicted										
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
A C T u a l	**1**	22	2	11	1	0	0	0	0	1	1	0.579
	2	0	5	0	0	0	0	0	0	0	0	1.0
	3	7	3	31	4	0	4	0	0	0	3	0.596
	4	0	0	3	3	0	0	0	0	0	0	0.5
	5	0	0	0	0	4	0	0	0	0	0	1.0
	6	0	0	0	0	0	4	0	0	0	0	1.0
	8	0	0	0	0	0	0	0	8	0	0	1.0
	9	0	0	0	0	0	0	0	0	10	0	1.0
	10	2	0	0	0	0	0	0	0	0	15	0.882
		0.710	0.5	0.689	0.375	1.0	0.5	0	1.0	0.910	0.789	0.708

Table 5.15 gives the classification results of the testing set with 10-decision values by DR-OOR method.

Table 5.15 DR-OOR confusion matrix for 10-decision testing table

		Predicted										
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
A c t u a l	**1**	23	0	14	1	0	0	0	0	0	0	0.605
	2	0	2	3	0	0	0	0	0	0	0	0.4
	3	10	0	41	0	0	0	0	1	0	0	0.788
	4	2	0	4	0	0	0	0	0	0	0	0.0
	5	0	0	0	0	4	0	0	0	0	0	1.0
	6	0	0	2	0	0	2	0	0	0	0	0.5
	8	0	0	5	0	0	0	0	3	0	0	0.375
	9	0	0	6	0	0	0	0	0	4	0	0.4
	10	2	0	6	0	0	0	0	0	0	9	0.529
		0.62222	1.0	0.506	0.0	1.0	1.0	0	0.75	1.0	1.0	0.611

5.2.2 RSES Analysis

This section provides the results from RSES analysis with shorted confusion matrix that summarizes the performance of a classifier k, applied to the objects in an information system S. The performance evaluation is represented by two coefficients (the average accuracy and the best accuracy). Table 5.16 gives the

classification results of the training set and the testing set with 4-decision values by FR method.

Table 5.16 FR shorted confusion matrix for 4-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**0.897**	**0.912**	**0.615**	**0.652**
D = 1	0.959	0.952	0.625	0.833
D = 2	1.000	1.000	0.483	0.455
D = 3	0.839	0.877	0.654	0.607
D = 4	0.863	0.870	0.498	0.583

Table 5.17 gives the classification results of the training set and the testing set with 4-decision values by OOR method.

Table 5.17 OOR shorted confusion matrix for 4-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**0.904**	**0.912**	**0.594**	**0.622**
D = 1	0.958	0.952	0.642	0.636
D = 2	1.000	1.000	0.501	0.625
D = 3	0.855	0.877	0.641	0.633
D = 4	0.867	0.870	0.475	0.533

Table 5.18 gives the classification results of the training set and the testing set with 4-decision values by GR-FR method.

Table 5.18 GR-FR shorted confusion matrix for 4-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**0.893**	**0.895**	**0.606**	**0.644**
D = 1	0.944	0.952	0.640	0.806
D = 2	1.000	1.000	0.531	0.455
D = 3	0.845	0.836	0.655	0.607
D = 4	0.851	0.878	0.476	0.583

Table 5.19 gives the classification results of the training set and the testing set with 4-decision values by GR-OOR method.

Table 5.19 GR-OOR shorted confusion matrix for 4-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**0.893**	**0.895**	**0.603**	**0.652**
D = 1	0.946	0.942	0.801	0.833
D = 2	1.000	1.000	0.531	0.455
D = 3	0.843	0.843	0.652	0.607
D = 4	0.855	0.878	0.469	0.583

Table 5.20 gives the classification results of the training set and the testing set with 4-decision values by DT method.

Table 5.20 DT shorted confusion matrix for 4-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**1**	**1**	**0.659**	**0.741**
D = 1	1	1	0.602	0.727
D = 2	1	1	0.398	0.330
D = 3	1	1	0.806	0.917
D = 4	1	1	0.521	0.545

Table 5.21 gives the classification results of the training set and the testing set with 10-decision values by FR method.

Table 5.21 FR shorted confusion matrix for 10-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**1.000**	**1.000**	**0.751**	**0.789**
D = 1	1.000	1.000	0.577	0.583
D = 2	1.000	1.000	0.931	1.000
D = 3	1.000	1.000	0.720	0.818
D = 4	1.000	1.000	0.802	0.923
D = 5	1.000	1.000	1.000	1.000
D = 6	1.000	1.000	1.000	1.000
D = 8	1.000	1.000	0.921	0.857
D = 9	1.000	1.000	0.971	0.857
D = 10	1.000	1.000	1.000	1.000

Table 5.22 gives the classification results of the training set and the testing set with 10-decision values by OOR method.

Table 5.22 OOR shorted confusion matrix for 10-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**1.000**	**1.000**	**0.751**	**0.789**
D = 1	1.000	1.000	0.577	0.583
D = 2	1.000	1.000	0.931	1.000
D = 3	1.000	1.000	0.720	0.818
D = 4	1.000	1.000	0.802	0.923
D = 5	1.000	1.000	1.000	1.000
D = 6	1.000	1.000	1.000	1.000
D = 8	1.000	1.000	0.921	0.857
D = 9	1.000	1.000	0.971	0.857
D = 10	1.000	1.000	1.000	1.000

Table 5.23 gives the classification results of the training set and the testing set with 10-decision values by GR-FR method.

Table 5.23 GR-FR shorted confusion matrix for 10-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**1.000**	**1.000**	**0.754**	**0.780**
D = 1	1.000	1.000	0.582	0.583
D = 2	1.000	1.000	0.988	1.000
D = 3	1.000	1.000	0.729	0.795
D = 4	1.000	1.000	0.788	0.923
D = 5	1.000	1.000	1.000	1.000
D = 6	1.000	1.000	1.000	1.000
D = 8	1.000	1.000	0.921	0.857
D = 9	1.000	1.000	0.971	0.857
D = 10	1.000	1.000	1.000	1.000

Table 5.24 gives the classification results of the training set and the testing set with 10-decision values by GR-OOR method.

Table 5.24 GR-OOR shorted confusion matrix for 10-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**1.000**	**1.000**	**0.751**	**0.780**
D = 1	1.000	1.000	0.589	0.583
D = 2	1.000	1.000	0.931	1.000
D = 3	1.000	1.000	0.716	0.795
D = 4	1.000	1.000	0.788	0.923
D = 5	1.000	1.000	1.000	1.000
D = 6	1.000	1.000	1.000	1.000
D = 8	1.000	1.000	0.921	0.857
D = 9	1.000	1.000	0.971	0.857
D = 10	1.000	1.000	1.000	1.000

Table 5.25 gives the classification results of the training set and the testing set with 10-decision values by DT method.

Table 5.25 DT shorted confusion matrix for 10-decision training/testing table

Accuracy				
	Training Table		Testing Table	
	Average	The best	Average	The best
All Classes	**1.000**	**1.000**	**0.820**	**0.877**
D = 1	1.000	1.000	0.648	0.706
D = 2	1.000	1.000	1.000	1.000
D = 3	1.000	1.000	0.798	0.897
D = 4	1.000	1.000	0.891	0.917
D = 5	1.000	1.000	1.000	1.000
D = 6	1.000	1.000	1.000	1.000
D = 8	1.000	1.000	0.786	0.571
D = 9	1.000	1.000	0.971	1.000
D = 10	1.000	1.000	1.000	1.000

5.2.3 Classification Analysis Summary

Tables 5.26 – 5.29 summarize the classification results with 4-decision classes and 10-decision classes for the Rosetta system and RSES system respectively. The analysis results for the 4 decision values are summarized for the Rosetta system in Table 2.6 and GR-OOR has the best outcome. The analysis results for the 4 decision values are summarized for the RSES system in Table 5.28 and DT method has the best outcome. The analysis results for the 10 decision values are summarized for the Rosetta system in Table 5.27 and OOR method has the best outcome. The analysis results for the 10 decision values are summarized for the RSES system in Table 5.29 and DT method has the best outcome. The classification based on the DT-method was found to be consistently better than that

of other methods mentioned. DT-method is also a bit better than the approaches presented in paper [RPP00] in terms of the accuracy coefficient.

Table 5.26 summarizes the Rosetta system's classification results of the training set and the testing set with 4-decision values by FR method, OOR method, DR-FR method, DR-OOR method and GR-OOR method. The average accuracies and the best accuracies are presented respectively.

Table 5.26 Summary of 4-decision results for the Rosetta system

	FR		OOR		DR-FR		DR-OOR		GR-OOR	
	Aver	**Best**	**Aver**	**Best**	**Aver**	**Best**	**Aver**	**Best**	**Aver**	**Best**
Training Accuracy	0.912	0.927	0.912	0.920	0.895	0.910	0.893	0.905	0.912	0.914
Testing Accuracy	0.563	0.596	0.729	0.763	0.611	0.641	0.639	0.671	0.750	0.811

Table 5.27 summarizes the Rosetta system's classification results of the training set and the testing set with 10-decision values by FR method, OOR method, DR-FR method, DR-OOR method and GR-OOR method. The average accuracies and the best accuracies are presented respectively.

Table 5.27 Summary of 10-decision results for the Rosetta system

	FR		OOR		DR-FR		DR-OOR		GR-OOR	
	Aver	**Best**	**Aver**	**Best**	**Aver**	**Best**	**Aver**	**Best**	**Aver**	**Best**
Training Accuracy	1.000	1.000	1.000	1.000	0.952	1.000	0.952	1.000	0.963	1.000
Testing Accuracy	0.618	0.649	0.743	0.781	0.708	0.743	0.611	0.642	0.708	0.735

Table 5.28 summarizes the RSES system's classification results of the training set and the testing set with 4-decision values by FR method, OOR method, GR-FR method, GR-OOR method and DT method. The average accuracies and the best accuracies are presented respectively.

Table 5.28 Summary of 4-decision results for the RSES system

	FR		OOR		GR-FR		GR-OOR		DT	
	Aver	**Best**	**Aver**	**Best**	**Aver**	**Best**	**Aver**	**Best**	**Aver**	**Best**
Training Accuracy	0.897	0.912	0.904	0.912	0.893	0.895	0.893	0.895	1.000	1.000
Testing Accuracy	0.615	0.652	0.594	0.622	0.606	0.644	0.603	0.652	0.659	0.741

Table 5.29 summarizes the RSES system's classification results of the training set and the testing set with 10-decision values by FR method, OOR method, GR-FR method, GR-OOR method and DT method. The average accuracies and the best accuracies are presented respectively.

Table 5.29 Summary of 10-decision results for the RSES system

	FR		OOR		GR-FR		GR-OOR		DT	
	Aver	**Best**	**Aver**	**Best**	**Aver**	**Best**	**Aver**	**Best**	**Aver**	**Best**
Training Accuracy	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Testing Accuracy	0.751	0.789	0.751	0.789	0.754	0.780	0.751	0.780	0.820	0.877

6 Conclusions and Future Work

6.1 Conclusion

Our main objective is to find the best method classifying unseen objects connected with the weather data. Results of our experiments are presented in Tables 5.26-5.29. In order to rank those methods in a reasonable way we used the results gained in confusion matrices presented in Tables 5.3-5.25. The result indicates that the classification based on the decision tree method and the reducts with object related parameter is always better than other methods mentioned in this thesis. Moreover, the applicability of rough set approach for classifying new objects from volumetric storm cell data seems sometimes to be a little better than the approaches presented in [RPP00], and [APP99] in terms of accuracy coefficient. Although our approach based on rough set theory looks quite promising, as demonstrated by results of our experiments, more experiments with the data is needed. Therefore, more radar data and ground truth information need to acquire so that they provide the researchers with rich source of data. One should consider new representations of the data that will facilitate more discretization of the data. So, instead of using whole numbers like 0, 1, 2, 3, one might consider fractional or real-values in the interval [0,3]. The idea would be to strive for greater refinement, and care in the representation of data. It is desirable that meteorologists introduce refinements in the classification of the weather data. For example, it would be very helpful if the decision class "hail mixed with rain" (instead of "hail or rain") is introduced at the source of the data, not as a result of data mining. More radar observations using the refined observation system are needed to improve the weather data classification. The fact

that training accuracy is 1 and yet testing accuracy for some of the decision classes is very low provides an indication of what is commonly known as "over-fitting" to the validation set [CMB95]. More work needs to be done in selection of the training set (so that over-fitting does not occur). The method of cross-validation can be used by dividing the training set at random into S distinct segments. Then carry out training with S-1 segments and test the performance of the network by using the remaining segment and observing the resulting accuracy. The training process should then be repeated S times for each of the possible S choices for the test segment which is left out of the training process. This is known as the " leave out one" method. Finally, in the event that meteorologists introduce new decision-classes and/or new features of the radar observations, then the classification system needs to be recalibrated and tested.

6.2 Future work

The analysis of the information system is based on the attributes of the objects in the universe. Therefore, it is important that the derived features can describe accurately the storm cells. The program extracts the derived features of each internal cell snapshot, and the time embedded in the matched-cell filename is the time of the first cell snapshot within the cell. These brought out a potential inaccuracy. By comparing the time of the ground-truth event with the time in the filename, a ground truth event is matched to a matched-cell file. However, there are cell snapshots within the matched-cell file that may be at a time outside the chosen time window, and they will still be associated to the storm type of the ground truth event. For instance, a matched-cell file may begin at 17:00 and extend to 17:50 or

later. With 17:00 in the filename and a time window of ±10 minutes, this file will be associated to a ground truth event that occurred at 16:50 in the same region. This means that the derived features of the cell snapshot at 17:50 are going to be assigned a storm type of the ground truth event from 16:50. Obviously this is going to degrade performance. It is therefore a recommendation that the output of the derived features shall be improved. In addition, a user interface is needed so that a meteorologist can use a "trained" system to classify new weather data. This user interface will make it possible for several things to occur automatically: (i) preprocessing new raw data, (ii) classifying the processed data using the set of rules already obtained (reading in the system), (iii) computing the probability that the classification is correct, (iv) plot showing recent history for similar data vs. estimated correctness of the classification. Furthermore, it will be important to develop methodology for classification unseen objects with using the generalized rough membership function [PPSSR2001].

7 References

[JAJ01] J.F. Peters, A. Skowron, J. Stepaniuk, Information granule decomposition, *Fundamenta Informatica*, 2001 [to appear].

[JSM01] J.F. Peters, S. Ramanna, M. Borkowsky, A. Skowron, Z. Suraj, Sensor, filter and fusion models with rough Petri nets, *Fundamenta Informatica*, vol. 34, 2001, 1-19.

[JA01] J.F. Peters, A. Skowron, A rough set approach to knowledge discovery, *International Journal of Intelligent Systems*, 2001 [to appear].

[JFA01] J.F. Peters, A. Skowron, A rough set approach to reasoning about data. *International Journal of Intelligent Systems*, vol. 15, 2001, 1-2.

[JLS01] J.F.Peters, L.Han, S.Ramanna, Rough neural computing in signal analysis, *Computational Intelligence*, vol. 1, no. 3, 2001, 493-513.

[AJJ01] A. Skowron, J. Stepaniuk, J.F. Peters, Discovering patterns in information granules. In: S. Hirano, M. Inuiguchi, S. Tsumoto (Eds.), Rough Sets and Granular Computing. Berlin: Physica Verlag, 2001 [to appear].

[LJS99] L. Han, J.F. Peters, S. Ramanna, R. Zhai, Classifying faults in high voltage power systems: A rough-fuzzy neural computational approach. In: N.Zhong, A. Skowron, S. Ohsuga (Eds.), New Directions in Rough Sets, Data Mining, and Granular-Soft Computing, Lecture Notes in Artificial Intelligence, vol. 1711. Berlin: Springer-Verlag, 1999, 47-54.

[WLJ01] W. Pedrycz, L. Han, J.F. Peters, S. Ramanna, R. Zhai, Calibration of software quality: Fuzzy neural and rough neural approaches. *Neurocomputing*, vol. 36, 2001, 149-170.

[SLJA] S.K. Pal, L. Polkowski, J. Peters, A. Skowron, Rough neurocomputing: An Introduction. In: S. Pal, A. Skowron (Eds.), Rough Neurocomputing, ch. 7. Berlin: Springer Verlag [to appear].

[JS99] J.F. Peters, S. Ramanna, A rough sets approach to assessing software quality: Concepts and rough Petri net models. In: S.K. Pal and A.

Skowron (Eds.), *Rough-Fuzzy Hybridization: New Trends in Decision Making*. Berlin: Springer-Verlag, 1999, 349-380.

[AJ01] A. Skowron, J. Stepaniuk, J.F. Peters, Extracting patterns using information granules. In: S. Hirano, M. Inuiguchi, S. Tsumoto (Eds.), *Proc. of Int. Workshop on Rough Set Theory and Granular Computing* (RSTGC'01), Matsue, Shimane, Japan, 20-22 May 2001, 135-142.

[LRJ99] L. Han, R. Menzies, J.F. Peters, L. Crowe, High Voltage Power Fault-Detection and Analysis System: Design and Implementation. In: *Proc. Canadian Conf. on Electrical and Computer Engineering* (CCECE'99), Edmonton, Alberta, May 1999.

[AR02] A. Skowron, R.W. Swiniarski, Information granulation and pattern recognition. In: L. Polkowski, A. Skowron, Rough-Neuro Computing. Berlin: Physica-Verlag, 2002 [to appear].

[JZL99] J. Komorowski, Z. Pawlak, L. Polkowski, A. Skowron, Rough sets: A tutorial. In: S.K. Pal, A. Skowron (Eds.), Rough Fuzzy Hybridization: A New Trend in Decision-Making. Berlin: Springer-Verlag, 1999, 3-98.

[ZA94] Z. Pawlak, A. Skowron, Rough membership functions. In: R. Yager, M. Fedrizzi, J. Kacprzyk (Eds.), Advances in the Dempster-Shafer Theory of Evidence, NY, John Wiley & Sons, 1994, 251-271.

[ZJA01] Z. Pawlak, J.F. Peters, A. Skowron, Z. Suraj, S. Ramanna, M. Borkowski, Rough measures: Theory and Applications. In: S. Hirano, M. Inuiguchi, S. Tsumoto (Eds.), Rough Set Theory and Granular Computing, Bulletin of the International Rough Set Society, vol. 5, no. 1 / 2, 2001, 177-184.

[WJ98] W. Pedrycz, J.F. Peters, Learning in fuzzy Petri nets. In: J. Cardoso and H. Scarpelli (Eds.), *Fuzziness in Petri Nets*. Berlin: Physica Verlag, a division of Springer Verlag, 1998.

[JW99] J.F. Peters, W. Pedrycz, Computational Intelligence. In: J.G. Webster (Ed.), *Encyclopedia of Electrical and Electronic Engineering*. 22 vols. NY, John Wiley & Sons, Inc., 1999.

[JLS99] J.F. Peters, L. Han, S. Ramanna, Approximate time rough software cost decision system: Multicriteria decision-making approach. In: *Proc. Eleventh Int. Symposium on Methodologies for Intelligent Systems*, Warsaw, Poland, 8-11 June 1999 [to appear].

[JAZ98] J.F. Peters, A. Skowron, Z. Suraj, S. Ramanna, A. Paryzek, Modeling real-time decision-making systems with rough fuzzy Petri nets. In: *Proc. of 6th European Congress on Intelligent Techniques & Soft Computing* (EUFIT98), vol. 1, 7-10 Sept. 1998, Aachen, Germany, 985-989.

[ARS] A. Skowron, R.W. Swiniarski, Information granulation and pattern recognition. In: [1].

[CMB95] C.M. Bishop, Neural Networks for Pattern Recognition. UK: Oxford University Press, 1995.

[ZJA02] Z. Pawlak, J.F. Peters, A. Skowron, S. Ramanna, M. Borkowski, Rough membership functions on finite and infinite sets. *Fundamenta Informatica*, 2002 [in preparation].

[ZJAZS] Z. Pawlak, J.F. Peters, A. Skowron, Z. Suraj, S. Ramanna, M. Borkowski, Properties and Applications of Rough measures and Integrals. In: S. Hirano, M. Inuiguchi, S. Tsumoto (Eds.), Rough Sets and Granular Computing. Berlin: Physica Verlag [to appear].

[ZJAZS] Z. Pawlak, J.F. Peters, A. Skowron, Z. Suraj, S. Ramanna, M. Borkowski, Rough measures and Integrals. In: S. Hirano, M. Inuiguchi, S. Tsumoto (Eds.), **LNCS** [to appear].

[AJJ02] A. Skowron, J. Stepaniuk, J.F. Peters, Hierarchy of information granules. In: In: H.D. Burkhard, L. Czala, H.S. Nguyen, P. Starke (Eds.), *Proc. of the Workshop on Concurrency, Specification and Programming*, Oct. 2001, Warsaw, Poland [to appear].

[AJJ03] A. Skowron, J. Stepaniuk, J.F. Peters, Rough granules in spatial reasoning. In: *Proc. Joint 9th International Fuzzy Systems Association (IFSA) World Congress and 20th North American Fuzzy Information*

Processing Society (NAFIPS) Int. Conf., Vancouver, British Columbia, Canada, 25-28 June 2001, 1355-1361.

[SJJ00] Skowron, J. Stepaniuk, J.F. Peters, Approximation of information granule sets. In: W. Ziarko, Y. Yao (Eds.), *Rough Sets and Current Trends in Computing* (RSCTC'2000). Banff, Alberta, Canada, 2000, 33-39.

[AC92] A. Skowron, C. Rauszer, The discernibility matrices and functions on information systems. In: R. Slowinski (Ed.), Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory. Dordrecht: Kluwer Academic Publishers, 1992, 331-362.

[ZP98] Z. Pawlak, Reasoning about data: A rough set perspective. In: L. Polkowski, A. Skowron (Eds.), Rough Sets and Current Trends in Computing (RSCTC'98), Lecture Notes in Artificial Intelligence 1424. Berlin: Springer-Verlag, 1998, 25-34.

[RL01] R.W. Swiniarski, L. Hargis, Rough sets as a front end of neural-networks texture classifiers, Neurocomputing, vol. 36, 2001, 85-102.

[WJ98] W. Pedrycz, J.F. Peters (Eds.), *Computational Intelligence in Software Engineering*. Singapore: World Scientific Publishing Co. Pte. Ltd., 1998, 485 pp. ISBN 981-02-3503-8.

[BSS94] Bazan, J., Skowron, A., Synak, P., *Dynamic reducts as a tool for extracting laws from decision tables*, Proc. Symp. On Methodologies for Intelligent Systems, Charlotte, NC, USA, Oct. 16-19, 1994, LNAI, 869, Springer, 346-355.

[APP99] Alexiuk, M., Pizzi, N., Pedrycz, W., *Classification of Volumetric Storm Cell Patterns*, Proc. of the 1999 IEEE Canadian Conference on Electrical and Computer Engineering, Edmonton, 1999.

[ALE99] Alexiuk, M. D., Pattern Recognition Techniques as Applied to the Classification of Convective Storm Cells, M.Sc. Thesis. University of Manitoba, Fall 1999.

[DIE00] Dietrich, J., *Report on Project EC-NRC*, spring 2000.

[FPS96] Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., *The KDD process for extracting useful knowledge from volumes of data*, Comm. ACM, 39/11 (1996), 27-34.

[BUS97] Grzymala-Busse, J.W., *A new version of the rule induction system LERS*, Fundamenta Informaticae 31 (1997), 27-39.

[KPPS] Komorowski, J., Pawlak, Z., Polkowski, L., Skowron, A., *A Rough Set Perspective on Data and Knowledge*. In L. Polkowski and A. Skowron (Eds.). Rough Sets in Knowledge Discovery 1: Methodology and Applications. Physica-Verlag, Heidelberg, 1998.

[OHRN] Øhrn, A., *Discernibility and Rough Sets in Medicine: Tools and Applications*, Ph. D. Thesis, Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway, 1999.

[OKS98] Øhrn, A., Komorowski, J., Skowron, A., Synak, P., *The Design and Implementation of a Knowledge Discovery Toolkit Based on Rough Sets – The Rosetta system.* In: L. Polkowski, A. Skowron (Eds.), Rough Sets in Knowledge Discovery 1: Methodology and Applications, Physica-Verlag, Heidelberg, 1998, 376-399.

[LPP99] Li, P. C., Pizzi, N., Pedrycz, W., *Classification of Hail and Tornado Storm Cells Using Neural Networks*, Project EC-NRC Research Reports, 1999.

[LPPWV] Li, P.C., Pizzi, N., Pedrycz, W., Westmore, D., and Vivanco, R., Severe *Storm Cell Classification Using Derived Products Optimized by Genetic Algorithm*, Proc. of the 2000 IEEE Canadian Conference on Electrical and Computer Engineering, Halifax, 2000.

[PAW91] Pawlak, Z., *Rough sets – theoretical aspects of reasoning about data*, Kluwer Academic Publ., Dordrecht 1991.

[NGH99] Nguyen, S. H., *Data regularity analysis and applications in data mining*, Ph. D. Thesis, Faculty of Math., Comp. Sci. and Mechanics, Warsaw University, Warsaw 1999.

[PPSSR] Pawlak, Z., Peters, J., F., Skowron, A., Suraj, Z., Ramanna, S., *Rough Measures: Theory and Applications* (in preparation)

[RPP00] Ramirez, L., Pedrycz, W., Pizzi, N., Storm Cell Classification With the Use of Support Vector Machines (draft version).

[ROSET] The ROSETTA WWW homepage, http://www.idi.ntnu.no/~aleks/rosetta/

[RSES] The RSES WWW homepage, http://logic.mimuw.edu.pl/~rses/

[WES99] Westmore, D., *Radar Decision Support System: User Manual*, InfoMagnetics Technologies Corporation Technical Document, 1999.

[WRO95] Wroblewski, J., *Finding minimal reducts using genetic algorithms.* In P.P. Wang (Ed.), Proc. Of the International Workshop on Rough Sets Soft Computing at Second Annual Joint Conference on Information Sciences (JCIS'95), Wrightsville Beach, NC, Sep. 28 – Oct. 10, 1995, 186-189.

[SRC92] Skowron, A., Rauszer, C.: The *discernibility matrices and* functions in information systems. In: R. Slowinski (ed.): Intelligent Decision Support: Handbook of Applications and Advances of Rough Sets Theory, Kluwer Academic Publishers 1992, Dordrecht, 331-362.

[NSS95] Nguyen, S.H., Skowronn, A.: Quantization of real value attributes. Proceedings of the Second Joint Annual Conference on Information Sciences, Wrightsville Beach, NC, September 28 – October 1, 1995, 34-37.

[SA95] Skowron, A.: *Extracting laws from decision tables: a rough set approach.* Computational Intelligence, Vol. 11-2, 1995, 371-388.

[SPK96] Skowron, A., Polkowski, L., Komorowski, J.: *Learning tolerance relations by Boolean descriptors: automatic feature extraction from data tables.* Proceedings of the Fourth International Workshop on Rough Sets, Fuzzy Sets, and Machine Discovery (RSFD-96), Tokyo, Japan, November 6-8, a996, 11-17.

[NGH97] Nguyen, S.H.: Discretization of real-valued attributes: Boolean reasoning approach. Ph. D. Thesis, Faculty of Mathematics, Computer Science and Mechanics, Warsaw University, 1997.

[LB991] Li, Ben, "Classification User Documentation", *Project EC-NRC Research Reports*, 1999.

[LB992] Li, Ben, "Description of Algorithms for Pattern Recognition", *Project EC-NRC Research Reports*, 1999.

[LB993] Li, Ben, editor, "Short Description of Algorithms for Generation of Storm Cell Products", *Project EC-NRC Research Reports*, 1999.