PERGAMON

Pattern Recognition 35 (2002) 2425–2438

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

A rough-fuzzy approach for generating classification rules

Qiang Shen*, Alexios Chouchoulas

Centre for Intelligent Systems and their Applications, Division of Informatics, University of Edinburgh, 80 South Bridge, Edinburgh EH1 1HN, UK

Received 10 October 2000; received in revised form 14 June 2001; accepted 20 August 2001

Abstract

The generation of effective feature pattern-based classification rules is essential to the development of any intelligent classifier which is readily comprehensible to the user. This paper presents an approach that integrates a potentially powerful fuzzy rule induction algorithm with a rough set-assisted feature reduction method. The integrated rule generation mechanism maintains the underlying semantics of the feature set. Through the proposed integration, the original rule induction algorithm (or any other similar technique that generates descriptive fuzzy rules), which is sensitive to the dimensionality of the dataset, becomes usable on classifying patterns composed of a moderately large number of features. The resulting learned ruleset becomes manageable and may outperform rules learned using more features. This, as demonstrated with successful realistic applications, makes the present approach effective in handling real world problems.

Keywords: Pattern classification; Rough sets; Fuzzy sets; Feature selection; Rule induction

1. Introduction

Intelligent pattern recognition systems have been successful in many application areas. However, complex application problems, such as reliable monitoring and diagnosis of industrial plants and rapid detection and estimation of environmental changes, have emphasised the issue of knowledge acquisition and modelling. These problems are likely to present large numbers of features, not all of which will be essential for the task at hand. Inaccurate and/or uncertain values cannot be ruled out, either. Furthermore, such applications typically require convincing explanations about the inference performed. A method to allow automated generation of knowledge models of clear semantics is, therefore, highly desirable.

The most common approach to developing expressive and human readable representations of knowledge is the use of if-then production rules. Yet, real-life problem domains usually lack generic and systematic expert rules for mapping feature patterns onto their underlying classes. This paper, based on the novel ideas proposed by the authors as briefly summarised in Ref. [1], presents a methodology to generate classification rules in an automatic, efficient and domain-independent manner.

The work aims to induce low-dimensionality rulesets from historical descriptions of domain features (often of high dimensionality). In particular, an exhaustive fuzzy rule induction algorithm (RIA), as first reported in Ref. [2] is chosen to act as the starting point for this. It should be noted, however, that the flexibility of the system discussed here allows the incorporation of almost any rule induction algorithm that uses descriptive set representation of features [3]. The current RIA, provided with sets of continuous feature values, induces classification rules to partition the feature patterns into underlying categories. It is chosen for properties such as graceful handling of missing and inaccurate information or vague data, domain independence, incremental operation [4]. However, as with many RIAs, this algorithm exhibits high computational complexity due to its

* Corresponding author. Tel.: +44-131-650-2705; fax: +44-131-650-6516.

E-mail addresses: qiangs@dai.ed.ac.uk (Q. Shen), alexios@dai.ed.ac.uk (A. Chouchoulas).

0031-3203/02/$22.00
PII: S0031-3203(01)00229-1

generate-and-test nature. The effects of this become evident where patterns of high dimensionality need to be processed.

In order to speed up the RIA, a preprocessing step is required. This is particularly important for tasks where learned rulesets need regular updating to reflect the changes in the description of domain features. This step, herein implemented using rough set theory [5], reduces the dimensionality of potentially very large feature sets without losing information needed for rule induction, insofar as this is possible in the domain at hand. It has an advantageous side-effect in that it removes redundancy from the historical data. This also helps simplify the design and implementation of the actual pattern classifier itself, by determining what features should be made available to the system. In addition, the reduced input dimensionality increases the processing speed of the classifier, leading to better response times. Most significant, however, is the fact that rough set feature reduction (RSFR) preserves the semantics of the surviving features after removing any redundant ones. This is essential in satisfying the requirement of user readability of the generated knowledge model, as well as ensuring the understandability of the pattern classification process.

There are several existing approaches relevant to the task at hand, both from the point of view of applications and that of computational methods. For example, the FAPACS[1] algorithm documented in Refs. [6,7] is able to discover fuzzy association rules in relational databases. It works by locating pairs of attributes that satisfy an 'interestingness' measure that is defined in terms of an adjusted difference between the observed and expected values of relations. This algorithm is capable of expressing linguistically both the regularities and the exceptions discovered within the data. Romahi and Shen [8] have applied this approach to financial forecasting with promising results. Hayashi et al. [9] have documented modifications to the Fuzzy ID3 (itself an augmentation of Quinlan's original ID3 [10]) rule induction algorithm to better support fuzzy learning. In a similar attempt, Janikow [11] has proposed modifications to decision trees to combine traditional symbolic decision trees with approximate reasoning, offered by fuzzy representation. This approach redefines the methodology for knowledge inference, resulting in a method best suited to relatively stationary problems.

A common disadvantage of these techniques is their sensitivity to high dimensionality. This may be remedied using Principal Components Analysis (PCA), a well-known tool for data analysis and transformation [12,13]. However, although PCA is an efficient methodology, it irreversibly destroys the underlying semantics of the dataset. Further reasoning about the data is almost always humanly impossible, prohibiting the use of PCA as a dataset pre-processor for symbolic or descriptive fuzzy modelling. By implication, only purely numerical (non-symbolic) datasets may be processed by PCA.

Most semantics-preserving dimensionality reduction (feature selection) approaches tend to be domain specific, however, utilising well-known features of specific application domains. RSFR offers an alternative approach that preserves the underlying semantics of the data while allowing reasonable generality. The theoretical domain independence of RSFR allows it to be used with different rule induction algorithms as mentioned earlier, in addition to the specific RIA adopted herein. In light of this, the present work is developed in a highly modular manner.

The rest of this paper is organised as follows. Section 2 describes the proposed methodology that integrates dimensionality reduction and rule induction. It summarises the theoretical background of the currently used RIA and RSFR algorithms, and discusses important design and implementation issues involved. Section 3 provides in detail two problem cases, which both justify the need for the present work and set up the scene for the experimental investigations reported. Note that, to reflect the generality and utility of the approach, this paper presents substantial additional experimental results to those presented in Ref. [1]. Section 4 provides the results of applying the methodology to developing classifiers for these problems, supported by a comparison to the application of C4.5 to the same domains. Section 5 concludes the paper and proposes further work.

2. Rough-fuzzy rule induction

In essence, the proposed approach deals with patterns involving a large set of features, by applying a dimensionality reduction algorithm on the feature set to discover a smaller set of features that convey all the information with as little redundancy as possible. Patterns formed using the resulting dimensionally reduced feature set are then extracted from the original pattern descriptions and fed to a rule induction algorithm to generate a suitable ruleset. As shown in Fig. 1, a rough-fuzzy system following this approach integrates the following modules:

Feature reduction: reads a set of feature patterns and outputs another with reduced dimensionality, implemented with the RSFR algorithm.

Rule induction: reads a sequence of feature patterns and outputs a set of if-then rules connecting features and their implied classes, implemented with the RIA algorithm.

Fuzzy reasoner: a standard approximative reasoner that interprets the induced fuzzy ruleset and uses it to classify previously unseen feature patterns.

A summary of the RIA and RSFR algorithms is given below. Details about the fuzzy reasoner module are beyond

[1] FAPACS stands for *Fuzzy Automatic Pattern Analysis and Classification System*.

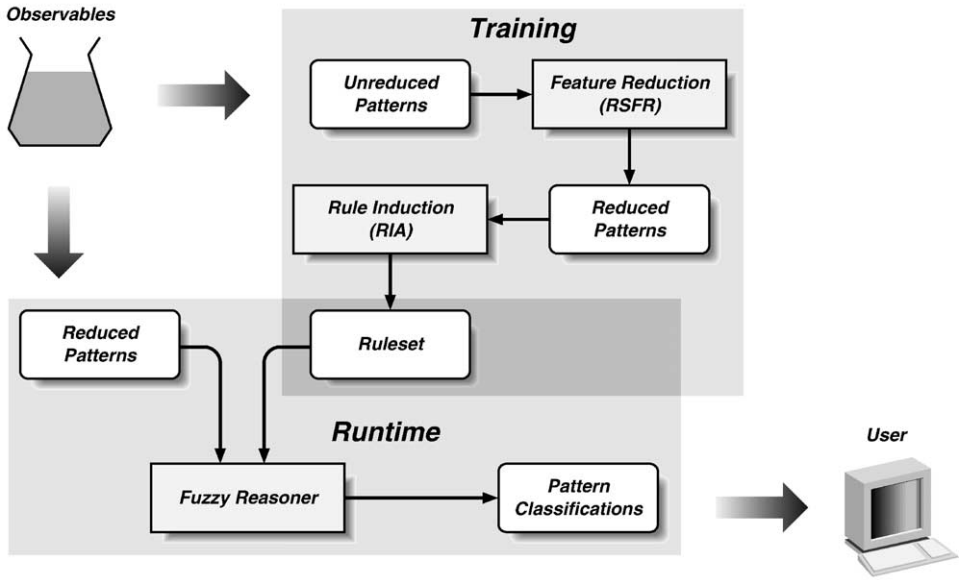

Fig. 1. Block diagram of the dimensionality-reducing rule induction approach.

the scope of this paper but can be easily found in the relevant literature (e.g. [14]).

2.1. Fuzzy rule induction

As reflected in Fig. 1, central to the present work is the creation of descriptive production rules from given feature patterns. This relies on the use of a data-driven rule induction method. The fuzzy RIA adopted to implement this task is described below.

2.1.1. The algorithm

The rule induction algorithm as presented in Ref. [2] extracts linguistically expressed fuzzy rules from real-valued examples. Although this RIA was proposed to be used in conjunction with neural network-based classifiers, it appears to be independent of the actual type of classifier used. Provided with training data, the RIA induces approximate relationships between the characteristics of the conditional attributes (pattern features) and their decision attributes (underlying classes). The premise attributes of the induced rules are represented by fuzzy variables, facilitating the modelling of the inherent uncertainty of the knowledge domain.

An additional input item needed by this algorithm is the fuzzification of the conditional attributes as feature patterns are in general assumed to be available in real numbers (though these numbers may not be accurately measured). For presentational simplicity, the terms *dataset* and *set of historical patterns* are hereafter used interchangeably, and so are the term *historical patterns* and the term *training data* (or examples).

A decision region is a set comprising examples' row numbers, for which a certain decision attribute y_i has a certain value c: $D_{y_i=c} = \{x\colon\ y_{xi} = c \wedge x \in \{1,\ldots,k\}\}$, where y_{xi} is the value of the decision attribute y_i as given by the example with number x, and k is the total number of examples in the dataset.

Next, the algorithm generates a set of all the possible combinations of fuzzy sets defined in the underlying ranges of the n conditional attributes. Each of the n attributes has its own value range divided up into a number of fuzzy sets. Such a range is hereafter called a fuzzy region for short. For a domain with n conditional attributes, each of which is represented by f_x fuzzy sets ($1 \leqslant x \leqslant n$), there will be $\prod_{i=1}^{n} f_i$ possible combinations, each referred to as a fuzzy set vector. Each vector represents an emerging pattern of rule conditions that may lead to a fuzzy rule, provided that dataset examples support it. Note that, in implementing this algorithm, it is infeasible and undesirable to implement directly this step. Alternative, computationally equivalent methods are employed to avoid the overhead of generating and storing this potentially vast set of combinations.

Based on this, it is possible to measure the evidence contributed by an example $\mathbf{x}$ towards the establishment of a fuzzy rule denoted by a fuzzy set vector $\mathbf{p} = \langle \mu_1, \mu_2, \ldots, \mu_n \rangle$ (as produced in the previous step). The t-norm operator is used to this end. Hence, $T^{\mathbf{p}}(\mathbf{x}) = \min(\mu_1(x_1), \mu_2(x_2), \ldots, \mu_n(x_n))$ (Fuzzy intersection or t-norm), where $x_1, x_2, \ldots, x_n$ are conditional attribute values. Sets of these values are formed for each decision $y_i = c$: $\mathbb{T}^{\mathbf{p}}_{y_i=c} = \{w\colon\ w = T^{\mathbf{p}}(\mathbf{x}_j)\ \forall j \in D_{y_i=c}\}$.

In a typical fuzzy region, no more than two fuzzy sets overlap for each underlying real element, while the region

is split into two or more fuzzy sets. This causes most of the memberships to evaluate to zero, in turn forcing $T^{\mathbf{p}}(\mathbf{x})$ to also evaluate to zero, due to the use of min(·).

For each decision $y_i = c$, the maximum element of $\mathbb{T}^{\mathbf{p}}_{y_i=c}$ is then evaluated (Fuzzy union or s-norm): $S^{\mathbf{p}}_{y_i=c} = \max\{x: x \in \mathbb{T}^{\mathbf{p}}_{y_i=c}\}$. This results in a multidimensional array of candidate rules describing the mapping between conditional and decision attributes. There is one array for each pair of decision attributes and decision attribute values (i.e. one for each decision region, as calculated above).

Fuzzy rules may not be generated directly from these arrays, as there are candidate rules for each possible decision $y_i = c$ and every possible combination of fuzzified conditional attributes. This implies that the candidate ruleset may comprise contradicting rules. A means of deciding which candidate rule best describes a given fuzzy set vector is needed.

A constant ε dubbed the *uncertainty margin* (or *tolerance*) is used to implement this control. A fuzzy rule concludes that class y_i has value c given attribute values matching the fuzzy premises in $\mathbf{p}$, if and only if the corresponding candidate rule is equal to at least ε more than the candidate rules supporting any competing classifications $y_i = c'$, where $c' \in Y_i$ (the set of possible values for decision class y_i) and $c' \neq c$. If no candidate rule can be decided upon (i.e. all values are within the ε neighbourhood), the whole rule is undecided for the fuzzy set vector in question. That is,

$$y_i = \begin{cases} c \text{ if } (\forall c' \in Y_i - \{c\}) \Rightarrow S^{\mathbf{p}}_{y_i=c} - S^{\mathbf{p}}_{y_i=c'} > \varepsilon, \\ - \text{ otherwise.} \end{cases} \quad (1)$$

The uncertainty margin introduces a trade-off to the rule generation step. The higher ε is, the less rules are generated. The accuracy of the generated rules in performing a certain application task is, of course, expected to be affected by changing ε. A careful selection of an appropriate ε value is therefore needed for a given problem at hand.

2.1.2. Algorithm modifications

The RIA summarised above is NP-hard and may become intractable, when inducing rules for datasets with many conditional attributes [4]. The most important problem both in terms of memory and runtime is dealing with the large numbers of combinations of fuzzy values. This may not be so significant when only a few attributes are involved, but with real applications such as the water treatment plant monitoring and algae population estimation (see Section 3), the algorithm may become intractable in terms of both time and space.

To improve the tractability of the RIA, it is convenient to treat the creation of fuzzy set vectors as the creation of a tree. In this context, a leaf node is one combination of membership functions and each arc represents one evaluation of a membership function. One branch of the tree from root to leaf consists of a candidate fuzzy rule's conditions and each arc of the tree represents a condition involving one variable.

The following observation is important to modify the algorithm. As the minimum membership is retained when applying the t-norm operator, any membership function that evaluates to zero implies immediately that t-norm will ultimately yield zero, regardless of the values of any other membership functions. Such a subtree is therefore useless and can be pruned if its root node evaluates to zero. This is shown in Fig. 2, where the leftmost branch of the right tree represents a membership evaluation yielding zero. The rest of the sub-tree is ignored. In this illustration, out of nine combinations in the complete tree, only six are used.

The gains with large numbers of conditional attributes are much greater. For instance, for the water treatment problem as reported in Section 3.1, using 38 conditional attributes and assuming that (for simplicity) each conditional attribute is broken up into five fuzzy sets, the RIA needs to generate 5^{38} combinations. If the pruning algorithm is used instead, and making the reasonable assumption that any real value may belong to at most two fuzzy sets, there is a *worst case* of 2^{38} combinations evaluated. The time needed is fifteen orders of magnitude less than that needed for the original algorithm which relies on full tree traversal. In so doing, the savings are significant, but the number of combinations is still far too large—this intractability is the reason for the most prominent disadvantage of this algorithm and many other descriptive fuzzy modelling techniques. It is also one of the motivations behind employing RSFR to remove some of the problems of rule induction over large datasets.

Additional space and time saving can also be facilitated by further improving the algorithm such that the generation of combinations, the evaluation of t-norm and the creation of candidate rules are merged in one step. This avoids temporary storage required for computing and evaluating fuzzy set vectors.

2.2. Rough set feature reduction

Although the RIA is potentially very useful, the NP-hard nature of the exhaustive search, even with tree-pruning improvements, prevents it from direct application to complex domains [4]. This is unfortunately true both in terms of computation time and the size of the resultant rulesets. The RSFR technique is thus employed to reduce the complexity by minimising redundancies contained within the set of feature patterns. It works by selecting those essential features that are most significant to the classification represented in the pattern set.

2.2.1. Basic concepts

The RSFR technique is herein explained in terms of the following notions: U, the set of all feature patterns in the dataset, along with their corresponding class labellings; A the set of all features; and B, the set of feature class labellings.

The value of feature $q \in A$ in pattern $x \in U$ is written as $f(x, q)$, which defines an equivalence relationship over U. Assume a subset of the set of features, $P \subset A$. Two patterns

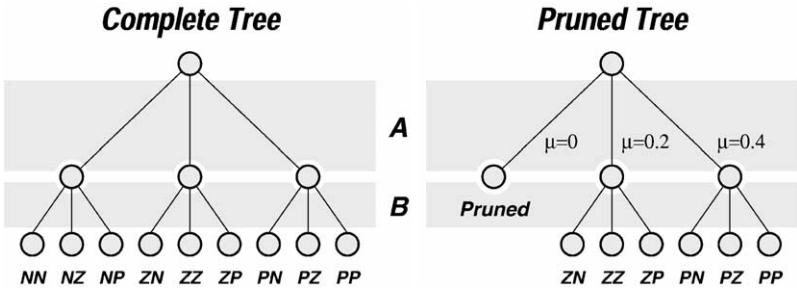

Fig. 2. RIA tree pruning. Arcs represent evaluations of membership functions. Each complete path to a leaf is one combination of fuzzy set vectors.

numbered x and y in U are *indiscernible* with respect to P if and only if $f(x,q) = f(y,q)\ \forall q \in P$. The indiscernibility relation for all $P \in A$ is written as IND(P). U/IND(P) is used to denote the partition of U given IND(P) and is calculated as:

$$U/\mathrm{IND}(P) = \otimes\{q \in P:\ U/\mathrm{IND}(q)\} \tag{2}$$

where

$$I \otimes J = \{X \cap Y:\ \forall X \in I,\ \forall Y \in J,\ X \cap Y \neq \emptyset\}. \tag{3}$$

A rough set approximates traditional sets using a pair of sets named the *lower* and *upper approximation* of the set in question. The lower and upper approximations of a set $P \subseteq U$ (given an equivalence relation IND(P)) are defined as:

$$\underline{P}Y = \cup\{X:\ X \in U/\mathrm{IND}(P),\ X \subseteq Y\}, \tag{4}$$

$$\bar{P}Y = \cup\{X:\ X \in U/\mathrm{IND}(P),\ X \cap Y \neq \emptyset\}. \tag{5}$$

Assuming P and Q are equivalence relations in U, the important concept *positive region* $\mathrm{POS}_P(Q)$ is defined as:

$$\mathrm{POS}_P(Q) = \bigcup_{X \in Q} \underline{P}X. \tag{6}$$

A positive region contains all patterns in U that can be classified in attribute set Q using the information in attribute set P. From this, the *degree of dependency* of a set Q of classification variables, $Q \subseteq B$ on a set of features P, where $P \subseteq A$, is defined as [5]:

$$\gamma_P(Q) = \frac{||\mathrm{POS}_P(Q)||}{||U||} \tag{7}$$

where $||S||$ denotes the cardinality of set S.

The degree of dependency $\gamma_P(Q)$ of a set P of conditional attributes (features) with respect to a set Q of decision attributes (pattern classes) provides a measure of how important P is in classifying the dataset examples into Q. If $\gamma_P(Q) = 0$, then classification Q is independent of the attributes in P, hence the decision attributes are of no use to this classification. If $\gamma = 1$, then Q is completely dependent on P, hence the attributes are indispensable. Values $0 < \gamma_P(Q) < 1$ denote partial dependency, which shows that only some of the attributes in P may be useful, or that the dataset was flawed to begin with. In addition, the complement of γ gives a measure of the contradictions in the selected subset of the dataset.

It is now possible to define the *significance* of a feature. This is done by calculating the change of dependency when removing the feature from the set of considered conditional attributes. Given P, Q and a feature $x \in P$:

$$\sigma_P(Q,x) = \gamma_P(Q) - \gamma_{P-\{x\}}(Q). \tag{8}$$

The higher the change in dependency, the more significant x is. Thus, feature selection involves removing features that have no significance to the pattern classification task at hand.

Central to the development of the RSFR is the notion of *feature reduct set* (or simply *reduct*). A reduct is defined as a subset R of the set of features C such that $\gamma_C(D) = \gamma_R(D)$, for the set of class labellings D. It is obvious that a dataset may have more than one feature reduct set for a given D. The set $\mathscr{R}$ of all feature reduct sets R is defined as:

$$\mathscr{R} = \{X:\ X \subseteq C,\ \gamma_C(D) = \gamma_X(D)\}. \tag{9}$$

RSFR *always* attempts to reduce the feature set while losing *no* information significant to the classification at hand. It searches for the feature reduct sets of least cardinality. That is, it seeks one or more elements in the set of *minimal reducts* $\mathscr{R}_{\min} \subseteq \mathscr{R}$:

$$\mathscr{R}_{\min} = \{X:\ X \in \mathscr{R},\ \forall Y \in \mathscr{R},\ ||X|| \leqslant ||Y||\}. \tag{10}$$

Clearly, RSFR preprocesses feature patterns without altering the feature values themselves, thus maintaining the semantics. Unlike statistical correlation-reducing approaches (e.g. the Principal Components Analysis [12,13]), RSFR dimensionality reduction does not require human intervention, or setting of variance thresholds. This property may also be a disadvantage however, since other techniques offer more aggressive dimensionality reduction, accepting that in some cases losing a little information may in fact prove to be advantageous (e.g. in noisy environments). It is, however, trivial to augment RSFR by adding a threshold to that effect. In terms of feature representation, RSFR is mainly intended for discrete domains. However, its dependence on nominal features does not give rise to the present problem of fuzzy

Algorithm 1 QUICKREDUCT($\mathbb{C}$, $\mathbb{D}$)
Input: $\mathbb{C}$, *the set of all feature attributes;* $\mathbb{D}$, *the set of class attribute.*
Output: R, *the attribute reduct,* $R \subseteq \mathbb{C}$

```
(1)  R ← {}
(2)  do
(3)    T ← R
(4)    foreach x ∈ (ℂ − R)
(5)      if γ_{R∪{x}}(𝔻) > γ_T(𝔻)
(6)        T ← R ∪ {x}
(7)    R ← T
(8)  until γ_R(𝔻) = γ_ℂ(𝔻)
(9)  return R
```

Fig. 3. The QUICKREDUCT algorithm.

Algorithm 2 QUICKREDUCT II($\mathbb{C}$, $\mathbb{D}$)
Input: $\mathbb{C}$, *the set of all feature attributes;* $\mathbb{D}$, *the set of class attributes.*
Output: R, *the attribute reduct,* $R \subseteq \mathbb{C}$

```
(1)  R ← 𝔻
(2)  do
(3)    T ← R
(4)    foreach x ∈ R
(5)      if γ_{R−{x}}(𝔻) = γ_T(𝔻)
(6)        T ← R − {x}
(7)    R' ← R
(8)    R ← T
(9)  until γ_R(𝔻) < γ_ℂ(𝔻)  ∨  R = R'
(10) return R'
```

Fig. 4. The QUICKREDUCT II algorithm.

knowledge modelling. This is because the real feature values are ultimately fuzzified for use by the RIA, becoming ordered symbolic values anyway.

2.2.2. Quick reduct search algorithm

The calculation of all possible subsets of a given set is an NP-hard task. In order to reduce computational complexity and memory requirements, the reduct subset search space is herein treated as a tree traversal. Each node of the tree represents the addition of one conditional or feature attribute to an initially empty reduct. Instead of generating the whole tree and picking the best path on it, the path is chosen progressively using the following heuristic: the next feature chosen to be added to the reduct is the feature that adds the most to the reduct's dependency. The search ends when the dependency reaches one, or when no more features are left. In so doing, it converts an otherwise exhaustive evaluation of all feature combinations into a best-first tree search. This is dubbed the QUICKREDUCT algorithm. For conciseness, this algorithm is summarised in pseudocode (see Fig. 3).

However, the version of QUICKREDUCT shown here is not guaranteed to yield a minimal reduct of the dataset provided. Thus, it has been necessary to update the algorithm, forming QUICKREDUCT II (see Fig. 4). The second version of the quick reduct search algorithm works by *removing* features from the full set of conditional attributes C, as long as the value of γ does not change. QUICKREDUCT II supersedes QUICKREDUCT II and is used herein.

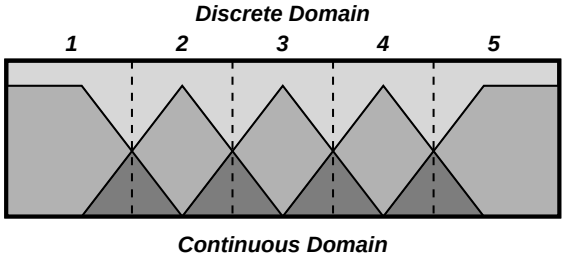

Fig. 5. Precategorisation of a continuous region into five discrete areas, by exploiting fuzzification information.

2.2.3. Precategorisation

This is required as a preprocessing stage before running the RSFR, mapping real values of features into ordered discrete symbolic values. The work required here must split the domain into no less discrete values than needed, as this will result in too much loss of information, and hence in rulesets with reduced accuracy. Splicing the domain into more values than necessary will, on the other hand, tax the induction process and is likely to produce less effective feature reductions.

As the rule induction algorithm employed describes pattern features using fuzzy terms (in an effort to cope with uncertainty in given patterns), in principle, any standard fuzzifier may be employed for this task. In practice, for convenience, the method used to implement the real-to-symbolic value mapping may take advantage of the fuzzification information used by the induction algorithm itself. It does not matter to the RIA what actual membership a pattern's certain feature has when considered belonging to a certain fuzzy value, as long as its membership is higher by at least ε than the membership in any of other fuzzy values covering the underlying value range of that feature. That is, the RIA essentially deals with ranges of values based on which fuzzy set provides the highest membership for a learning example.

Having taken notice of this, the precategorisation task is implemented here by partitioning the real-valued domain into discrete symbolic values (encoded as integers), one for each range in the domain:

$$P = \{[x_0, x_1), [x_1, x_2), \ldots, [x_{n-1}, x_n)\} \quad \text{so that}$$

$$\forall x,\ x_i \leqslant x < x_{i+1},\ i = 0, 1, \ldots, n-1 \Rightarrow \forall k\ \exists j,$$

$$\mu_j(x) \geqslant \mu_k(x) \tag{11}$$

where P is the set of all discrete values in ascending order of the value of feature x, n denotes the number of such symbolic values and is equal to the cardinality of P, and j and k indicate membership functions defined within the range $[x_i, x_{i+1})$. The process is illustrated in a simpler manner in Fig. 5. The required information on fuzzification may be provided by a human expert, by a clustering method [15], or by a genetic algorithm to search for the most suitable fuzzy

set definition [3] (though this may alter the semantics of the resulting fuzzy sets).

3. Problem cases

To emphasise the generality of the presented approach and its independence from any specific domain, two application case studies will be given later. To ease the presentation of experimental results, this section provides a brief description of each the two application problems.

3.1. Urban water treatment plant monitoring

The Water Treatment database, as archived in the UCI Machine Learning Database Repository [16] comprises a set of historical data obtained over a period of 521 days, with one series of measurements per day. Thirty eight different input attribute (feature) values are measured per day, with one set of such measurements forming one datum. All measurements are real-valued.

The goal is to implement a fuzzy reasoning system that, given this dataset of past measurements and without the benefit of an expert in the field at hand, will monitor the plant's status.

The domain was chosen because of its realism. A large plant is likely to involve a number of similar conditional attributes, not all of which will be essential in determining the operational status. Interrelations between attributes are unavoidable as the plant is a single system with interconnections. Also, unknown values cannot possibly be ruled out (due to, for instance, instrument failure or other event hindering the measurement). As such, there is a fair amount of redundancy in measurements obtained from the plant, as shown in Fig. 6.

The thirty eight conditional attributes account for the following five aspects of the water treatment plant's operation (see Fig. 7 for an illustration of this):

(1) Input to plant (9 attributes)
(2) Input to primary settler (6 attributes)
(3) Input to secondary settler (7 attributes)
(4) Output from plant (7 attributes)
(5) Plant performance (9 attributes).

The status of the plant is represented by classifying each day in one of 13 different categories, some representing normal operation of varying types, others pointing out faults in various parts of the plant. The categories are shown in Fig. 8. Because of the efficiency of the actual plant the measurements were taken from, all faults appear for short periods (usually single days) and are dealt with immediately. This does not allow for a lot of training examples of faults, which is a clear drawback if a monitoring system is to be produced.

In order to reduce redundancies and increase the number of examples of faults, it was necessary to collate most of the fault cases into two major categories, so that each category is well represented in the dataset. The result is a binary monitoring problem: 507 samples for acceptable (OK) performance and the remaining 14 samples for malfunctions.

3.2. Algae population estimation

Concern for environmental issues has increased in recent years. Waste production influences humanity's future. The alga, an ubiquitous single-celled plant, can thrive on industrial waste, to the detriment of water clarity and human activities. To avoid this, biologists need to isolate the chemical parameters of these rapid population fluctuations.

The task of this application problem is to estimate the populations of seven different species of alga based on eleven attributes of the river sample [17]:

- the time of year the sample was taken, given as a season,
- the size of the river,
- the flow rate of the water, and
- eight chemical concentrations, including nitrogen in the form of nitrates, nitrites, ammonia, phosphate, the pH of the water, oxygen and chloride.

To derive the rules required for estimation, training samples were taken from different European rivers over the period of one year. These samples were analysed to quantify the presence of the chemicals and water pH. The algae population distributions for each of the species involved were determined in the samples.

It is relatively easy to locate relations between one or two of these quantities and a species of algae. However, the process of identifying relations between different chemical elements and the population of different algae species requires expertise in chemistry and biology and involves well-trained personnel and microscopic examination that cannot be automated given the state of the art. Thus, the process becomes expensive and slow, even for a subset of the quantities involved here. There are complex relations at work between the attributes of this application domain, be they conditional or decision: algae may influence one another, as well as be influenced by the concentration of chemicals. As such, there is expected to be some redundancy in the data. An important reason for the present development is utilising the RSFR technique.

The dataset available for training includes 200 instances. The first three features of each instance (season, river size and flow rate) are represented as fuzzy linguistic variables. Chemical concentrations and algae population estimates are represented as continuous quantities. The dataset includes a few samples with missing values. Of the 200 instances, two exhibiting mostly unknown values were removed from the dataset because of their extremely low quality.

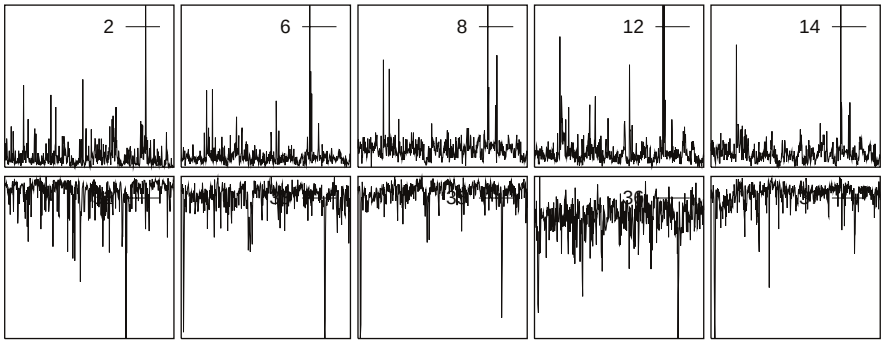

Fig. 6. Two groups of attributes, plotted over time. There is an obviously high degree of redundancy in the dataset.

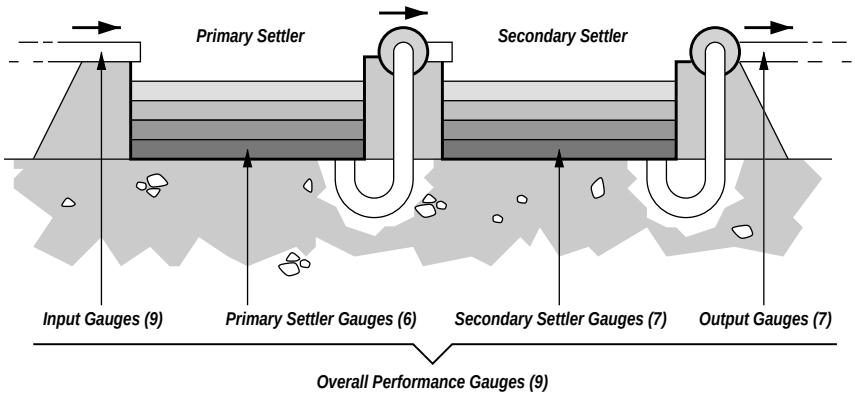

Fig. 7. Schematic diagram of the water treatment plant, indicating the number of measurements sampled at various points.

Value	Number	Description
1	275	Normal situation
2	1	Secondary settler, type 1 problems
3	1	Secondary settler, type 2 problems
4	4	Secondary settler, type 3 problems
5	114	Good performance
6	3	Solids overload, type 1
7	1	Secondary settler, type 4 problems
8	1	Storm, type 1
9	65	Normal situation, low influent
10	1	Storm, type 2
11	53	Normal situation
12	1	Storm, type 3
13	1	Solids overload, type 2

Fig. 8. The thirteen possible states of the water treatment plant.

To generalise given training samples, attributes of numerical values are preprocessed to become symbolic. As the first three conditional attributes are already represented in fuzzy terms, no such preprocessing is required for them. Matters differ for the eight chemical concentrations. As with all concentrations, these exhibit an exponential distribution (as shown in Fig. 9). To ease processing, samples were converted to a logarithmic scale defined by $f(x) = \log(x + 1)$, where x is the numerical measurement of an attribute.[2]

As can be expected, the distributions of the algae are also exponential. This, coupled with the fact that the decision attributes representing algae population counts are numerical, suggests the use of a similar treatment as above. The conditional attributes were thus transformed by $g(x) = \lfloor \log(x + 1) \rfloor$, where x is the numerical measurement of the algae community's population and $\lfloor \cdot \rfloor$ is the *floor* operator.[3] This quantisation is required because the proposed approach can only distinguish between discrete classes.

[2] Concentrations are non-negative real numbers, hence it is necessary to add an arbitrary constant to avoid the logarithm of zero.

[3] Yielding the maximum integer less than or equal to the floor operator's operand.

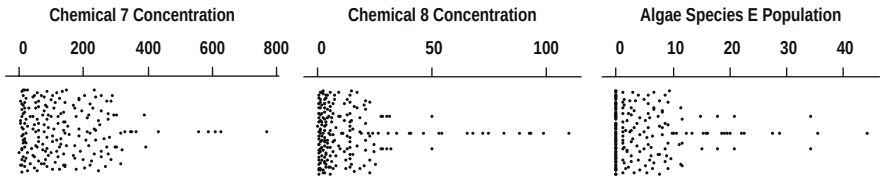

Fig. 9. Density plots for three of the algae dataset attributes: two of the eight chemical concentration distributions (left and middle), and the distribution of population values for one species of alga.

4. Experimental results

This section first provides separate results for the two problem cases used. Next, a comparison to the application of Quinlan's C4.5 algorithm [10] for both domains follows. Finally, a discussion on the impact of fuzzification on the process is given.

The results of a rule induction run are shown below using a pair of graphs: one shows the classification error plotted against ε (labelled tolerance); the other is a graph of the number of induced rules against ε. As described in Section 2.1, there is a trade-off between the number of generated rules and the classification error as decided by ε, the uncertainty margin or tolerance. To have an efficient runtime performance, the induced ruleset should be generated as small as possible (without sacrificing much classification accuracy). This makes it important to have a right choice for ε. However, the selection of ε is, in general, an application-specific task. A good choice for the uncertainty margin which provides a balance between a resultant ruleset's size and accuracy can be found by experiment. Results provided throughout this section demonstrate this.

4.1. The urban water treatment plant problem

Running the RSFR algorithm on the water treatment plant dataset provided a significant reduction, with RSFR selecting merely two conditional features from the total of 38, namely the conductivity of the water in the primary settler and *any* other attribute. Input flow was arbitrarily chosen as the second conditional attribute because there is no intuitively obvious relation between it, the primary settler conductivity attribute and the operational status of the plant.

The rule induction algorithm was then executed using the selected attributes to generate a ruleset. The results are illustrated in Fig. 10. The left plot gives the resulting classification error, while the right plot shows the size of the ruleset generated as a function of the uncertainty margin (labelled 'tolerance'). An additional number of rule induction runs were performed, using randomly selected groups of eleven conditional attributes each. The minimum, average and maximum values for the classification error and ruleset size of these extra runs are shown as the vertical lines on the graphs, to ease comparison.

Note that in these graphs, 'undecidable' answers by the RIA are considered wrong answers, this giving slightly more conservative results. In this case, top classification accuracy is around 96.5% with 'undecidable' counted as a wrong answer, or 97.1% with 'undecidable' counted as an acceptable admission of ignorance. These results were obtained using ten-fold cross-validation of the learned rulesets.

Clearly, the rulesets induced using the set of attributes chosen by the RSFR method are of better quality than the average randomly selected ones, which implies that the latter lose some important information in the course of attribute reduction and rule induction. Detailed investigation revealed that, on top of information loss incurred by choosing a subset of the original attributes, inexpert fuzzification is also responsible for the error during the rule induction phase. The fuzzification of certain conditional attributes is less successful than others, leading to the further removal of useful information. The use of a human expert in designing the fuzzification of the application would prove valuable in reducing the classification error of the resultant ruleset. Accuracy is also reduced because of the contents of the dataset, as there is very little information on plant faults present. Nevertheless, a classification rate of around 97% is very encouraging.

Even given the reduction in ruleset quality, though, using the set of attributes discovered by the RSFR algorithm could be several orders of magnitude faster than generating numerous rulesets in the hope of finding the optimum one. Running the RIA on all 38 conditional attributes would be computationally prohibitive; the reduced, two-attribute set requires a fraction of a second per example. This was timed on a common desktop workstation at the time of writing. Additionally, it is important to note that this particular RIA can be trained incrementally. The algorithm offers linear complexity with respect to the number of examples in the dataset—its NP-hard complexity is only with respect to the number of attributes.

Although the choice of the RIA used in this application was partially meant to emphasise the savings of dimensionality reduction, even much more efficient rule induction algorithms would benefit from this. As stated previously, however, the benefits do not limit themselves to the training stage; they extend to the runtime use of the system. By reducing the dimensionality of the data, the dimensionality of

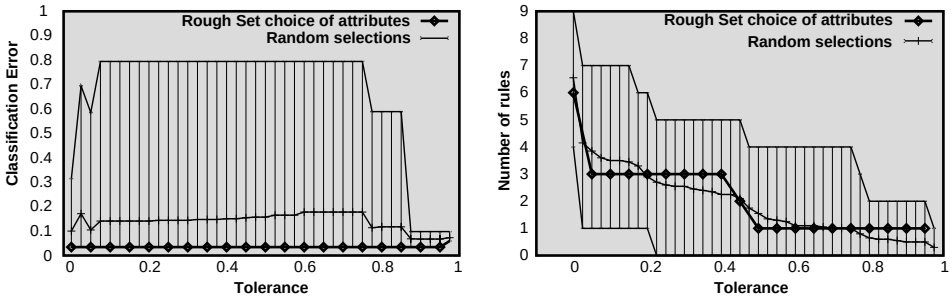

Fig. 10. Impact of dimensionality reduction.

the rule set is also decreased. This results in less measured features, which is very important for dynamic systems where observables are often restricted. This in turn leads to less connections to instrumentation and faster system responses in emergencies.

4.2. Algae population estimation

For convenience, each of the seven alga species were processed separately by the RIA in order to provide seven different rulesets. Each ruleset models the behaviour of one species. The separate rulesets can be merged trivially, to form a single ruleset. Alternatively, the RIA can be applied to all seven to produce directly a single, unified ruleset. This latter choice is, of course, a more inelegant and inflexible solution than having separate algae models. Therefore, the following results are shown with respect to individual algae species.

It is, first of all, interesting to investigate what effects dimensionality reduction may have on the runtime performance of this particular application. To show whether feature reduction has an impact on overall accuracy, the RIA algorithm was used to induce a ruleset from the entire, unreduced algae dataset [17]. The results are shown on the top row of Fig. 11. Then, RSFR was employed to reduce the dimensionality of the dataset, producing another ruleset from these reduced patterns. This resulted in a seven-feature dataset selected from the original, eleven-feature one. The results of testing the ruleset induced from this dataset are illustrated on the bottom row of Fig. 11.

The exact selected features were different for each alga species, although certain ones were present in all seven reduct sets, namely the Season and Concentrations 1, 4 and 7. There is a certain drop in accuracy (approximately 10%) after dimensionality reduction, which may indicate that the attribute reduction process has removed some of the necessary information.

To show that RSFR performs as claimed, it is desirable to prove two further points: that the RSFR algorithm truly finds a minimal reduct of the dataset; and that adding further attributes to this reduct does not produce better results.

To this end, two further groups of experiments were conducted. In the first, numerous datasets of six features each were randomly generated from the original, eleven-attribute algae dataset. Rulesets were induced from these, and the average estimation error of all runs was plotted, as shown on the right graph of Fig. 12 (where the left graph is the reduced dataset error for Fig. 11, copied here to ease comparison). Two empirical conclusions can be drawn from these results: first, not all features contribute the same information; second, the results obtained from random sets of features are worse than those obtained from the reduct set. The latter conclusion demonstrates that RSFR does indeed locate a relatively high-quality reduct.

In the second group of experiments, the four remaining conditional attributes were added to the seven-feature reduct one at a time. The aim was to show that more attributes do not necessarily imply higher accuracy. Rulesets were induced from these artificially produced feature sets, and the results were averaged. As shown on the right graph of Fig. 13 (again, the canonical, reduced results from Fig. 11 are shown on the left graph for comparison), error increased by adding an arbitrary feature to the reduct. This leads to the conclusion that the reduct indeed leads to an accuracy loss that is acceptably low.

4.3. Comparison with C4.5

C4.5 [10] is a widely accepted and powerful algorithm providing a good machine learning benchmark [18]. The decision trees it generates can be interpreted very quickly by the monitoring system. However, C4.5's decision tree for the Water Treatment Plant problem involves a total of three features from the dataset, as compared to two chosen by the RSFR algorithm. The runtime efficiency of the rules induced by such an approach may therefore be reduced, whilst the difficulty in performing effective monitoring and diagnosis would be increased due to the requirement of additional measurements, although, in this example, the difference is not very pronounced.

In terms of classification performance, C4.5 obtains an accuracy of around 96.8%. For easy comparison, the classi-

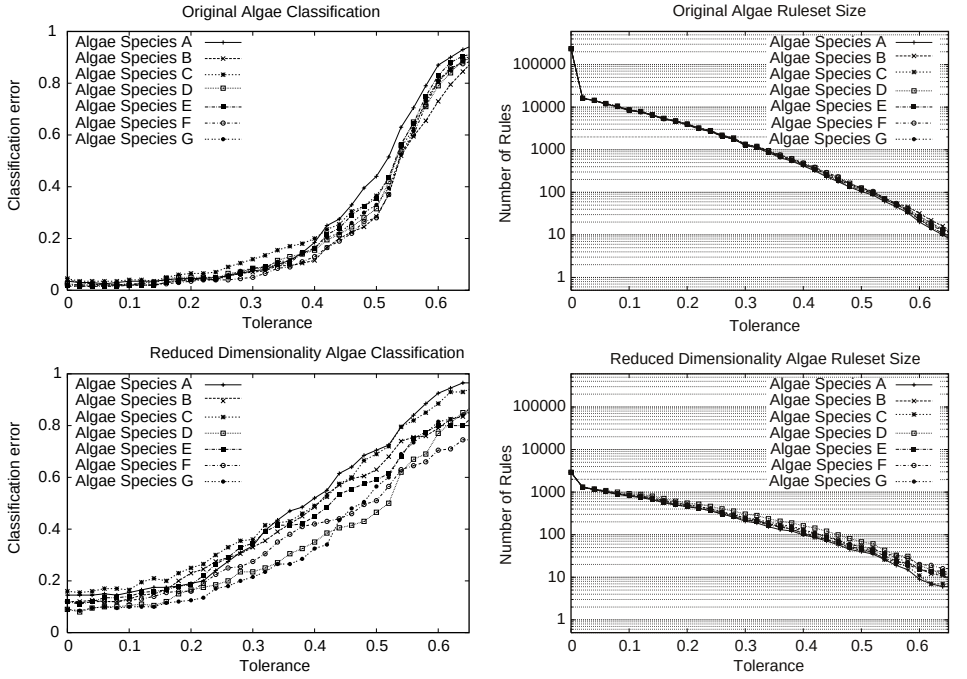

Fig. 11. Algae estimation accuracy before (top) and after (bottom) dimensionality reduction. Note that ruleset size is given on a logarithmic scale.

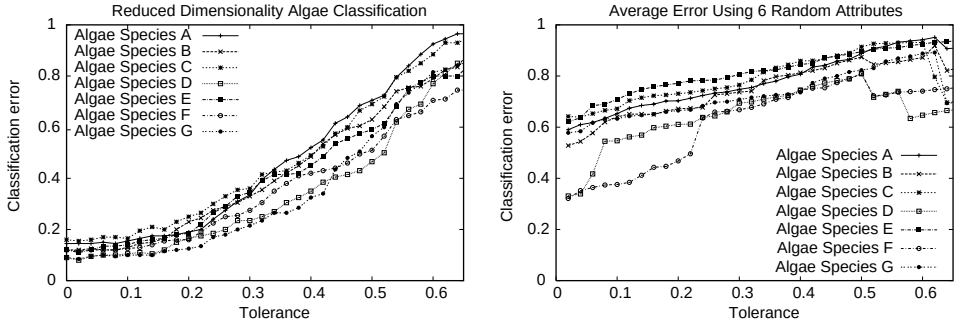

Fig. 12. Comparison of estimation error after training on the reduct set of seven attributes (left), and random sets of six attributes (right).

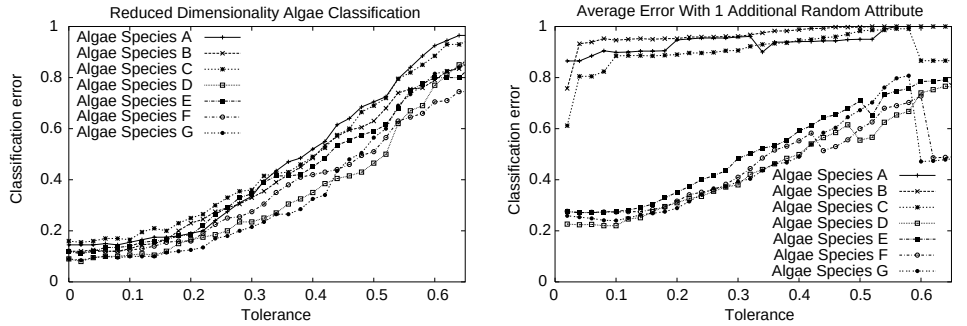

Fig. 13. Comparison of estimation error after training on the reduct set of attributes (left), and the reduct set plus one random attribute (right).

Method	Classification Accuracy	Number of Features
RSFR+RIA		
('undecidable' is wrong)	96.5%	2
('undecidable' is acceptable)	97.3%	2
C4.5	96.8%	3

Fig. 14. Comparison of classification accuracies and the number of features required by the algorithms, in the context of the Water Treatment Plant problem.

Algae Species	RSFR+RIA Error	RSFR+RIA Attributes	C4.5 Error	C4.5 Attributes
a	15%	7	18%	11
b	12%	7	19%	10
c	16%	7	24%	9
d	8%	7	13%	11
e	11%	7	14%	10
f	12%	7	15%	10
g	9%	7	16%	11

Fig. 15. Comparison with C4.5 in the context of the Algae problem.

fication performances of the two approaches are summarised in Fig. 14. Also included in the table are the numbers of features selected by RSFR and C4.5.

For the Algae Population Estimation problem, the system described herein is able to provide an estimation accuracy that clearly surpasses that of C4.5, all the while using a significantly smaller set of conditional attributes (as shown in Fig. 15, where undecidables are treated as misclassifications). Although C4.5 offers superior training speed, the higher number of features involved in the final system can be undesirable, inasmuch as the cost, complexity and time requirements of obtaining each set of measurements is proportional to the number of measurements in each set.

4.4. Impact of fuzzification

It is clear that the integrated rough-fuzzy approach works very well. This shows that real-world problems do contain a lot of redundancy which, once removed, allows highly accurate rulesets of low-arity rules to be induced.

However, the present work requires good fuzzification. In running the experiments reported above, no attempt to optimise the fuzzification was made, either via an automated membership function tuning tool [3] or via a domain expert. The fuzzification was instead performed by a basic statistical count of the dataset. Yet, better fuzzification generally leads to better classification. Therefore, with a better partition of the underlying numerical value range, the integrated rough-fuzzy learning method should be able to produce a more accurate ruleset for a given application.

Deviations from the expected results are partially blamed on inexpert fuzzification of the domain. This is because the

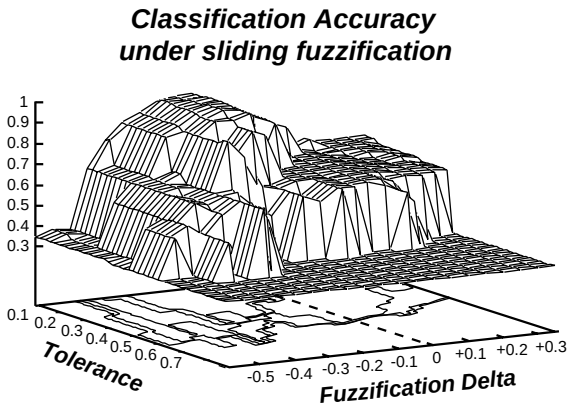

Fig. 16. Implications of shifting membership functions to the left or right along the x-axis. Note: classification *accuracy* is shown here rather than classification error.

rulesets induced by the integrated rule induction system are, as with the original RIA, based on the evaluation of fuzzy set membership functions. This makes the choice and configuration of these functions crucial for correct results. The fuzzification used, for instance, for the monitoring task at hand was performed using statistical analysis and intuition rather than human expertise in the specific domain. However, the system produces acceptable rulesets quickly, even with inexpertly fuzzified datasets.

To demonstrate the importance of correct fuzzification, the rough-fuzzy approach was also applied to a simple domain, the Iris dataset [19]. Numerous rule induction runs were performed and the rulesets were evaluated. For each run, the entire group of fuzzy set membership functions was shifted to the left or right along the x-axis. The results were plotted as a three-dimensional graph (see Fig. 16). This is essentially a family of accuracy/tolerance graphs. Classification accuracy is shown, rather than classification error, in an effort to demonstrate the effects of correct fuzzification as a hill. The third axis is the shift factor of the fuzzy membership functions. The dashed line marks the original, unshifted fuzzification information from the work of Lozowski [2]. The highest point in the graph is very close to that fuzzification. Shifting the membership functions to the left or right produces less accurate rulesets.

Finally, it is worth indicating that there are various approaches to optimising fuzzy rule sets automatically. A few aim at optimising the definition of the membership functions rather than the product ruleset [20]. Such systems could be applied here to improve fuzzification information.

5. Conclusion

Feature pattern-based if-then rules offer an expressive and human readable form of modelling knowledge effective for classification. Automated generation of such rules is essential to the practical success of most intelligent pattern

classifiers that rely on the use of such rules. This paper has presented an approach which integrates a potentially powerful fuzzy rule induction algorithm with a rough set-assisted feature reduction method. Unlike transformation-based techniques, this approach maintains the underlying semantics of the feature set. This is very important to ensure the resulting models are readily interpretable by the user and the inference performed explainable to the user. Through the integration, the original rule induction algorithm (or any other similar technique that generates fuzzy rules), which is sensitive to the dimensionality of the set of feature patterns, becomes usable on patterns involving a moderately large number of features.

The work has been applied to several real problem-solving tasks. In addition to the application results on monitoring of an industrial water treatment plant, as reported in Ref. [1] where the initial ideas of this research were first described, this paper has provided the results of utilising the approach for algae population estimation. Although the application problems encountered are complex, the resulting learned rulesets are manageable and may outperform rules learned using more features.

The performance of the present approach can be improved further. In particular, as indicated previously, fuzzification plays a very significant role in obtaining high quality rulesets. An optimising pre-processor for domain fuzzification would be very helpful. Techniques for fuzzy set optimisation, typically by the use of a genetic algorithm, to search for the most suitable fuzzy set definitions, have been proposed [3]. Investigation into the addition of such a technique to the rough-fuzzy learning mechanism, whilst minimising the loss of descriptiveness of the learned rules, is currently on-going at Edinburgh [3]. This can be especially beneficial in situations where no human experts are available to provide initial categorisation of the domain.

The ruleset generated by the RIA was not processed by any post-processing tools so as to allow its behaviour and capabilities to be revealed fully. By enhancing the induced ruleset through post-processing, performance can be expected to improve. This conjecture is supported by existing work in processing various types of learned ruleset, as reported in the literature [20–22]. Finally, it is also very interesting to modify the QuickReduct algorithm by allowing the inclusion of more than one feature at a time in the emerging minimal reduct. This will help boost the efficiency of the rule generation process.

References

[1] Q. Shen, A. Chouchoulas, Combining rough sets and data-driven fuzzy learning, Pattern Recognition 32 (12) (1999) 2073–2076.

[2] A. Lozowski, T.J. Cholewo, J.M. Zurada, Crisp rule extraction from perceptron network classifiers, Proceedings of the International Conference on Neural Networks, Volume of Plenary, Panel and Special Sessions, 1996, pp. 94–99.

[3] Q. Shen, J.G. Marin-Blazquez, A. Tuson, Tuning fuzzy membership functions with neighbourhood search techniques: a comparative study. Proceedings of the Third IEEE International Conference on Intelligent Engineering Systems, 1999, pp. 337–342.

[4] A. Chouchoulas, Q. Shen, Rough set-aided rule induction for plant monitoring, Proceedings of the 1998 International Joint Conference on Information Science (JCIS'98), Vol. 2, 1998, pp. 316–319.

[5] Z. Pawlak, Rough Sets: Theoretical Aspects of Reasoning About Data, Kluwer Academic Publishers, Dordrecht, 1991.

[6] W.H. Au, K.C.C. Chan, An effective algorithm for discovering fuzzy rules in relational databases, Proceedings of the Seventh IEEE International Conference on Fuzzy Systems, 1998, pp. 1314–1319.

[7] K. Chan, A.A. Wong, Apacs: a system for automatic analysis and classification of conceptual patterns, Comput. Intelligence 10 (1990) 119–131.

[8] Y. Romahi, Q. Shen, Dynamic financial forecasting with automatically induced fuzzy associations, Proceedings of the Ninth International Conference on Fuzzy Systems, Vol. 1, 2000, pp. 493–498.

[9] I. Hayashi, T. Maeda, A. Bastian, L.C. Jain, Generation of fuzzy decision trees by fuzzy id3 with adjusting mechanism of and/or operators, Proceedings of the Seventh IEEE International Conference on Fuzzy Systems, 1998, pp. 681–685.

[10] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[11] C.Z. Janikow, Fuzzy decision trees: issues and methods, IEEE Trans. Systems Man Cybernet.—Part B: Cybernetics 28 (1) (1998) 1–14.

[12] P. Devijver, J. Kittler, Pattern Recognition: A Statistical Approach, Prentice Hall, Englewood Cliffs, NJ, 1982.

[13] B. Flury, H. Riedwyl, Multivariate Statistics: A Practical Approach, Prentice Hall, Englewood Cliffs, NJ, 1988.

[14] E. Cox, The Fuzzy Systems Handbook: a Practitioner's Guide to Building, Using and Maintaining Fuzzy Systems, Academic Press, Inc, New York, 1994.

[15] R. Krishnapuram, J. Keller, The possibilistic C-means algorithm: insights and recommendations, IEEE Trans. Fuzzy Systems 4 (3) (1996) 385–393.

[16] Various, The Machine Learning Database Repository of the University of California, Irvine, 1995.

[17] ERUDIT, European Network for Fuzzy Logic and Uncertainty Modelling in Information Technology, Protecting rivers and streams by monitoring chemical concentrations and algae communities (3rd international competition), 1999.

[18] T.M. Mitchell, Machine Learning, McGraw-Hill, New York, 1997.

[19] R.A. Fisher, Iris plants database, 1936.

[20] A.F. Gómez-Skarmeta, F. Jiménez, Generating and tuning fuzzy rules using hybrid systems, Proceedings of the Seventh IEEE International Conference on Fuzzy Systems, 1997, pp. 247–251.

[21] W. Mees, Detection of defects in a fuzzy knowledge base, Proceedings of the Eighth IEEE International Conference on Fuzzy Systems, 1999, pp. 204–209.

[22] M.W. Kim, J.G. Lee, C. Min, Efficient fuzzy rule generation based on fuzzy decision tree for data mining, Proceedings of the Eighth IEEE International Conference on Fuzzy Systems, 1999, pp. 1223–1228.

About the Author—QIANG SHEN is a senior lecturer in the Division of Informatics at the University of Edinburgh, where he leads the Approximate and Qualitative Reasoning group. His research interests include fuzzy and imprecise modelling; model-based inference; pattern recognition; and knowledge refinement and reuse. Dr. Shen has published over 110 peer-refereed papers in academic journals and conferences on topics within Artificial Intelligence and related areas.

About the Author—ALEXIOS CHOUCHOULAS is a Ph.D. student in the Division of Informatics at the University of Edinburgh, working in the Approximate and Qualititative Reasoning Group. His research interests include rough and fuzzy set theory; pattern recognition; and information retrieval and filtering. He has published around 15 peer-refereed articles in these areas. Much of the work reported in this paper was carried out in an externally funded research project, to which Chouchoulas was appointed as a research associate.