Performing Feature Selection with ACO

Jensen, Richard

Published in:
Swarm Intelligence and Data Mining

Publication date:
2006

Citation for published version (APA):
Jensen, R. (2006). Performing Feature Selection with ACO. In *Swarm Intelligence and Data Mining* (pp. 45-73). Springer Nature.

Performing Feature Selection with ACO

Richard Jensen

Department of Computer Science, The University of Wales, Aberystwyth, UK
rkj@aber.ac.uk

Summary. The main aim of feature selection is to determine a minimal feature subset from a problem domain while retaining a suitably high accuracy in representing the original features. In real world problems FS is a must due to the abundance of noisy, irrelevant or misleading features. However, current methods are inadequate at finding optimal reductions. This chapter presents a feature selection mechanism based on Ant Colony Optimization in an attempt to combat this. The method is then applied to the problem of finding optimal feature subsets in the fuzzy-rough data reduction process. The present work is applied to two very different challenging tasks, namely web classification and complex systems monitoring.

1 Introduction

Many problems in machine learning involve high dimensional descriptions of input features. It might be expected that the inclusion of an increasing number of features would increase the likelihood of including enough information to distinguish between classes. Unfortunately, this is not true if the size of the training dataset does not also increase rapidly with each additional feature included. This is the so-called curse of dimensionality. A high-dimensional dataset increases the chances that a data-mining algorithm will find spurious patterns that are not valid in general. It is therefore not surprising that much research has been carried out on dimensionality reduction [6, 18]. However, existing work tends to destroy the underlying semantics of the features after reduction.

The task of feature selection is to significantly reduce dimensionality by locating minimal subsets of features, at the same time retaining data semantics. The use of rough set theory (RST) [21] to achieve such data reduction has proved very successful. Over the past twenty years, rough set theory has become a topic of great interest to researchers and has been applied to many domains (e.g. classification [8], systems monitoring [29], clustering [12], expert systems [32]). This success is due in part to the following aspects of the theory: only the facts hidden in data are analysed, no additional information about the data is required (such as thresholds or expert knowledge), and it finds a minimal knowledge representation. Given a dataset with discretized attribute values, it is possible to find a subset (termed a *reduct*) of the original attributes using RST that are the most informative; all other attributes can be removed from the dataset with minimal information loss.

However, it is most often the case that the values of attributes may be both crisp and *real-valued*, and this is where traditional rough set theory encounters a problem. It is not possible in the theory to say whether two attribute values are similar and to what extent they are the same; for example, two close values may only differ as a result of noise, but in RST they are considered to be as different as two values of a different order of magnitude.

It is, therefore, desirable to develop these techniques to provide the means of data reduction for crisp and real-value attributed datasets which utilises the extent to which values are similar. This could be achieved through the use of *fuzzy-rough* sets. Fuzzy-rough set theory is an extension of crisp rough set theory, allowing all memberships to take values in the range [0,1]. This permits a higher degree of flexibility compared to the strict requirements of crisp rough sets that only deal with full or zero set membership. They encapsulate the related but distinct concepts of vagueness (for fuzzy sets [37]) and indiscernibility (for rough sets [21]), both of which occur as a result of imprecision, incompleteness and/or uncertainty in knowledge [9].

Ant Colony Optimization (ACO) techniques are based on the behaviour of real ant colonies used to solve discrete optimization problems [2]. These have been successfully applied to a large number of difficult combinatorial problems such as the quadratic assignment and the traveling salesman problems. This method is particularly attractive for feature selection as there seems to be no heuristic that can guide search to the optimal minimal subset (of features) every time. Additionally, it can be the case that ants discover the best feature combinations as they proceed throughout the search space. This chapter investigates how ant colony optimization may be applied to the difficult problem of finding optimal feature subsets, using fuzzy-rough sets, within web classification and systems monitoring programs.

The rest of this chapter is structured as follows. The second section describes the theory of rough sets and particularly focuses on its role as a feature selection tool. The extension to this approach, fuzzy-rough set feature selection, is detailed in the third section. Section 4 introduces the main concepts in ACO and details how this may be applied to the problem of feature selection in general, and fuzzy-rough feature selection in particular. The fifth section describes the experimentation carried out using the crisp ACO-based feature selector. The application of the fuzzy-rough techniques to web content classification and complex system monitoring is detailed in section 6. Section 7 concludes the chapter, and proposes further work in this area.

2 Rough Feature Selection

Rough set theory [10, 20, 21] is an extension of conventional set theory that supports approximations in decision making. It possesses many features in common (to a certain extent) with the Dempster-Shafer theory of evidence [30] and fuzzy set theory [35]. The rough set itself is the approximation of a vague concept (set) by a pair of precise concepts, called lower and upper approximations, which are a classification of the domain of interest into disjoint categories. The lower approximation is a description of the domain objects which are known with certainty to belong to the subset of interest, whereas the upper approximation is a description of the objects which possibly belong to the subset.

Rough Set Attribute Reduction (RSAR) [3] provides a filter-based tool by which knowledge may be extracted from a domain in a concise way; retaining the information content whilst reducing the amount of knowledge involved. The main advantage that rough set analysis has is that it requires no additional parameters to operate other than the supplied data [11]. It works by making use of the granularity structure of the data only.

2.1 Theoretical Background

Central to RSAR is the concept of indiscernibility. Let $I = (\mathbb{U}, \mathbb{A})$ be an information system, where $\mathbb{U}$ is a non-empty set of finite objects (the universe) and $\mathbb{A}$ is a non-empty finite set of attributes such that $a : \mathbb{U} \to V_a$ for every $a \in \mathbb{A}$. V_a is the set of values that attribute a may take. For a decision table, $\mathbb{A} = \{\mathbb{C} \cup \mathbb{D}\}$ where $\mathbb{C}$ is the set of input features and $\mathbb{D}$ is the set of class indices. Here, a class index $d \in \mathbb{D}$ is itself a variable $d : \mathbb{U} \to \{0, 1\}$ such that for $a \in \mathbb{U}, d(a) = 1$ if a has class d and $d(a) = 0$ otherwise.
With any $P \subseteq \mathbb{A}$ there is an associated equivalence relation $IND(P)$:

$$IND(P) = \{(x, y) \in \mathbb{U}^2 | \forall a \in P, a(x) = a(y)\} \tag{1}$$

The partition of $\mathbb{U}$, generated by *IND(P)* is denoted $\mathbb{U}$/*IND(P)* (or $\mathbb{U}/P$) and can be calculated as follows:

$$\mathbb{U}/IND(P) = \otimes\{a \in P : \mathbb{U}/IND(\{a\})\}, \tag{2}$$

where

$$A \otimes B = \{X \cap Y : \forall X \in A, \forall Y \in B, X \cap Y \neq \emptyset\} \tag{3}$$

If $(x, y) \in IND(P)$, then x and y are indiscernible by attributes from P. The equivalence classes of the P-indiscernibility relation are denoted $[x]_P$.
Let $X \subseteq \mathbb{U}$. X can be approximated using only the information contained within P by constructing the P-*lower* and P-*upper* approximations of X:

$$\underline{P}X = \{x \mid [x]_P \subseteq X\} \tag{4}$$

$$\overline{P}X = \{x \mid [x]_P \cap X \neq \emptyset\} \tag{5}$$

Let P and Q be equivalence relations over $\mathbb{U}$, then the positive, negative and boundary regions can be defined as:

$$\begin{aligned} POS_P(Q) &= \bigcup_{X \in \mathbb{U}/Q} \underline{P}X \\ NEG_P(Q) &= \mathbb{U} - \bigcup_{X \in \mathbb{U}/Q} \overline{P}X \\ BND_P(Q) &= \bigcup_{X \in \mathbb{U}/Q} \overline{P}X - \bigcup_{X \in \mathbb{U}/Q} \underline{P}X \end{aligned}$$

The positive region contains all objects of $\mathbb{U}$ that can be classified to classes of $\mathbb{U}/Q$ using the information in attributes P. The boundary region, *BND*$_P$*(Q)*, is the set of objects that can possibly, but not certainly, be classified in this way. The negative region, *NEG*$_P$*(Q)*, is the set of objects that cannot be classified to classes of $\mathbb{U}/Q$.

An important issue in data analysis is discovering dependencies between attributes. Intuitively, a set of attributes Q depends totally on a set of attributes P, denoted $P \Rightarrow Q$, if all attribute values from Q are uniquely determined by values of attributes from P. If there exists a functional dependency between values of Q and P, then Q depends totally on P. In rough set theory, dependency is defined in the following way:

For $P,\ Q \subset \mathbb{A}$, it is said that Q depends on P in a degree k ($0 \leq k \leq 1$), denoted $P \Rightarrow_k Q$, if

$$k = \gamma_P(Q) = \frac{|POS_P(Q)|}{|\mathbb{U}|} \tag{6}$$

If $k = 1$, Q depends totally on P, if $0 < k < 1$, Q depends partially (in a degree k) on P, and if $k = 0$ then Q does not depend on P.

By calculating the change in dependency when an attribute is removed from the set of considered conditional attributes, a measure of the significance of the attribute can be obtained. The higher the change in dependency, the more significant the attribute is. If the significance is 0, then the attribute is dispensable. More formally, given *P,Q* and an attribute $a \in P$,

$$\sigma_P(Q, a) = \gamma_P(Q) - \gamma_{P-\{a\}}(Q) \tag{7}$$

2.2 Reduction Method

The reduction of attributes is achieved by comparing equivalence relations generated by sets of attributes. Attributes are removed so that the reduced set provides the same predictive capability of the decision feature as the original. A *reduct* is defined as a subset of minimal cardinality R_{min} of the conditional attribute set $\mathbb{C}$ such that $\gamma_R(\ \mathbb{D}) = \gamma_{\mathbb{C}}(\mathbb{D})$.

$$R = \{X : X \subseteq \mathbb{C}, \gamma_X(\mathbb{D}) = \gamma_{\mathbb{C}}(\mathbb{D})\} \tag{8}$$

$$R_{min} = \{X : X \in R, \forall Y \in R, |X| \leq |Y|\} \tag{9}$$

The intersection of all the sets in R_{min} is called the *core*, the elements of which are those attributes that cannot be eliminated without introducing more contradictions to the dataset. In RSAR, a subset with minimum cardinality is searched for.

The problem of finding a reduct of an information system has been the subject of much research. The most basic solution to locating such a subset is to simply generate *all* possible subsets and retrieve those with a maximum rough set dependency degree. Obviously, this is an expensive solution to the problem and is only practical for very simple datasets. Most of the time only one reduct is required as, typically, only one subset of features is used to reduce a dataset, so all the calculations involved in discovering the rest are pointless.

To improve the performance of the above method, an element of pruning can be introduced. By noting the cardinality of any pre-discovered reducts, the current possible subset can be ignored if it contains more elements. However, a better approach is needed - one that will avoid wasted computational effort.

```
QuickReduct($\mathbb{C}$,$\mathbb{D}$).
$\mathbb{C}$, the set of all conditional features;
$\mathbb{D}$, the set of decision features.
    $R \leftarrow \{\}$
    do
        $T \leftarrow R$
        $\forall x \in (\mathbb{C} - R)$
            if $\gamma_{R \cup \{x\}}(\mathbb{D}) > \gamma_T(\mathbb{D})$
                $T \leftarrow R \cup \{x\}$
        $R \leftarrow T$
    until $\gamma_R(\mathbb{D}) == \gamma_{\mathbb{C}}(\mathbb{D})$
    return $R$
```

Fig. 1. The QuickReduct Algorithm

The QuickReduct algorithm given in Fig. 1 (adapted from [3]), attempts to calculate a reduct without exhaustively generating all possible subsets. It starts off with an empty set and adds in turn, one at a time, those attributes that result in the greatest increase in the rough set dependency metric, until this produces its maximum possible value for the dataset. Other such techniques may be found in [23].

Determining the consistency of the entire dataset is reasonable for most datasets. However, it may be infeasible for very large data, so alternative stopping criteria may have to be used. One such criterion could be to terminate the search when there is no further increase in the dependency measure. This will produce exactly the same path to a reduct due to the monotonicity of the measure [3], without the computational overhead of calculating the dataset consistency.

The QuickReduct algorithm, however, is not guaranteed to find a *minimal* subset as has been shown in [4]. Using the dependency function to discriminate between candidates may lead the search down a non-minimal path. It is impossible to predict which combinations of attributes will lead to an optimal reduct based on changes in dependency with the addition or deletion of single attributes. It does result in a close-to-minimal subset, though, which is still useful in greatly reducing dataset dimensionality. However, when maximal data reductions are required, other search mechanisms must be employed. Although these methods also cannot ensure optimality, they provide a means by which the best feature subsets might be found.

3 Fuzzy-Rough Feature Selection

The selection process described previously based on crisp rough sets (RSAR) can only operate effectively with datasets containing discrete values. However, most datasets contain real-valued features and so it is necessary to perform a discretization step beforehand. This is typically implemented by standard fuzzification techniques. As membership degrees of feature values to fuzzy sets are not exploited in the process of dimensionality reduction, important information has been lost. By employing *fuzzy-rough* sets, it is possible to use this information to better guide feature selection.

A fuzzy-rough set is defined by two fuzzy sets, fuzzy lower and upper approximations, obtained by extending the corresponding crisp rough set notions. In the crisp case, elements that belong to the lower approximation (i.e. have a membership of 1) are said to belong to the approximated set with absolute certainty. In the fuzzy-rough case, elements may have a membership in the range [0,1], allowing greater flexibility in handling uncertainty.

3.1 Fuzzy Equivalence Classes

Fuzzy equivalence classes [9, 19] are central to the fuzzy-rough set approach in the same way that crisp equivalence classes are central to classical rough sets. For typical applications, this means that the decision values and the conditional values may all be fuzzy. The concept of crisp equivalence classes can be extended by the inclusion of a fuzzy similarity relation S on the universe, which determines the extent to which two elements are similar in S. The usual properties of reflexivity ($\mu_S(x,x) = 1$), symmetry ($\mu_S(x,y) = \mu_S(y,x)$) and transitivity ($\mu_S(x,z) \geq \mu_S(x,y) \wedge \mu_S(y,z)$) hold.

Using the fuzzy similarity relation, the fuzzy equivalence class $[x]_S$ for objects close to x can be defined:

$$\mu_{[x]_S}(y) = \mu_S(x,y) \tag{10}$$

The following axioms should hold for a fuzzy equivalence class F:

- $\exists x,\ \mu_F(x) = 1$
- $\mu_F(x) \wedge \mu_S(x, y) \leq \mu_F(y)$
- $\mu_F(x) \wedge \mu_F(y) \leq \mu_S(x, y)$

The first axiom corresponds to the requirement that an equivalence class is non-empty. The second axiom states that elements in y's neighbourhood are in the equivalence class of y. The final axiom states that any two elements in F are related via the fuzzy similarity relation S. Obviously, this definition degenerates to the normal definition of equivalence classes when S is non-fuzzy. The family of normal fuzzy sets produced by a fuzzy partitioning of the universe of discourse can play the role of fuzzy equivalence classes [9].

3.2 Fuzzy Lower and Upper Approximations

The fuzzy lower and upper approximations are fuzzy extensions of their crisp counterparts. Informally, in crisp rough set theory, the lower approximation of a set contains those objects that belong to it with certainty. The upper approximation of a set contains the objects that possibly belong. From the literature, the fuzzy P-lower and P-upper approximations are defined as [9]:

$$\mu_{\underline{P}X}(F_i) = inf_x max\{1 - \mu_{F_i}(x), \mu_X(x)\} \quad \forall i \tag{11}$$

$$\mu_{\overline{P}X}(F_i) = sup_x min\{\mu_{F_i}(x), \mu_X(x)\} \quad \forall i \tag{12}$$

where $\mathbb{U}/P$ stands for the partition of the universe of discourse, $\mathbb{U}$, with respect to a given subset P of features, and F_i denotes a fuzzy equivalence class belonging to $\mathbb{U}/P$. Note that although the universe of discourse in feature reduction is finite, this is not the case in general, hence the use of *sup* and *inf* above. These definitions diverge a little from the crisp upper and lower approximations, as the memberships of individual objects to the approximations are not explicitly available. As a result of this, the fuzzy lower and upper approximations are redefined as [14]:

$$\mu_{\underline{P}X}(x) = \sup_{F \in \mathbb{U}/P} min(\mu_F(x), \inf_{y \in \mathbb{U}} max\{1 - \mu_F(y), \mu_X(y)\}) \tag{13}$$

$$\mu_{\overline{P}X}(x) = \sup_{F \in \mathbb{U}/P} min(\mu_F(x), \sup_{y \in \mathbb{U}} min\{\mu_F(y), \mu_X(y)\}) \tag{14}$$

The tuple $< \underline{P}X, \overline{P}X >$ is called a fuzzy-rough set. For this particular feature selection method, the upper approximation is not used, though this may be useful for other methods.

For an individual feature, a, the partition of the universe by $\{a\}$ (denoted $\mathbb{U}/IND(\{a\})$) is considered to be the set of those fuzzy equivalence classes for that feature. For example, if the two fuzzy sets N_a and Z_a are generated for feature a during fuzzification, the partition $\mathbb{U}/IND(\{a\}) = \{N_a, Z_a\}$. If the fuzzy-rough feature selection process is to be useful, it must be able to deal with multiple features, finding the dependency between various subsets of the original feature set. For instance, it may be necessary to be able to determine the degree of dependency of the decision feature(s) with respect to feature set $P = \{a, b\}$. In the crisp case, $\mathbb{U}/P$ contains sets of objects grouped together that are indiscernible according to both features a and b. In the fuzzy case, objects may belong to many equivalence classes, so the cartesian product of $\mathbb{U}/IND(\{a\})$ and $\mathbb{U}/IND(\{b\})$ must be considered in determining $\mathbb{U}/P$. In general,

$$\mathbb{U}/P = \otimes\{a \in P : \mathbb{U}/IND(\{a\})\} \tag{15}$$

For example, if $P = \{a, b\}$, $\mathbb{U}/IND(\{a\}) = \{N_a, Z_a\}$ and $\mathbb{U}/IND(\{b\}) = \{N_b, Z_b\}$, then

$$\mathbb{U}/P = \{N_a \cap N_b, N_a \cap Z_b, Z_a \cap N_b, Z_a \cap Z_b\}$$

Clearly, each set in $\mathbb{U}/P$ denotes an equivalence class. The extent to which an object belongs to such an equivalence class is therefore calculated by using the conjunction of constituent fuzzy equivalence classes, say F_i, $i = 1, 2, ..., n$:

$$\mu_{F_1 \cap \ldots \cap F_n}(x) = min(\mu_{F_1}(x), \mu_{F_2}(x), ..., \mu_{F_n}(x)) \tag{16}$$

3.3 Fuzzy-Rough Reduction Method

Fuzzy-Rough Feature Selection (FRFS) [14] builds on the notion of the fuzzy lower approximation to enable reduction of datasets containing real-valued features. The process becomes identical to the crisp approach when dealing with nominal well-defined features.

The crisp positive region in the standard RST is defined as the union of the lower approximations. By the extension principle, the membership of an object $x \in \mathbb{U}$, belonging to the fuzzy positive region can be defined by

$$\mu_{POS_P(Q)}(x) = \sup_{X \in \mathbb{U}/Q} \mu_{\underline{P}X}(x) \tag{17}$$

Object x will not belong to the positive region only if the equivalence class it belongs to is not a constituent of the positive region. This is equivalent to the crisp version where objects belong to the positive region only if their underlying equivalence class does so.

Using the definition of the fuzzy positive region, a new dependency function between a set of features Q and another set P can be defined as follows:

$$\gamma'_P(Q) = \frac{|\mu_{POS_P(Q)}(x)|}{|\mathbb{U}|} = \frac{\sum_{x \in \mathbb{U}} \mu_{POS_P(Q)}(x)}{|\mathbb{U}|} \tag{18}$$

As with crisp rough sets, the dependency of Q on P is the proportion of objects that are discernible out of the entire dataset. In the present approach, this corresponds to determining the fuzzy cardinality of $\mu_{POS_P(Q)}(x)$ divided by the total number of objects in the universe.

FRQuickReduct(C,D).
C, the set of all conditional features;
D, the set of decision features.

```
$R \leftarrow \{\}$, $\gamma'_{best} \leftarrow 0$, $\gamma'_{prev} \leftarrow 0$
do
    $T \leftarrow R$
    $\gamma'_{prev} \leftarrow \gamma'_{best}$
    $\forall x \in (C - R)$
        if $\gamma'_{R\cup\{x\}}(D) > \gamma'_T(D)$
            $T \leftarrow R \cup \{x\}$
            $\gamma'_{best} \leftarrow \gamma'_T(D)$
    $R \leftarrow T$
until $\gamma'_{best} = \gamma'_{prev}$
return $R$
```

Fig. 2. The fuzzy-rough QuickReduct algorithm

A new QuickReduct algorithm, based on the crisp version [3], has been developed as given in Fig. 2. It employs the new dependency function γ' to choose which features to add to the current reduct candidate. The algorithm terminates when the addition of any remaining feature does not increase the dependency. As with the original algorithm, for a dimensionality of n, the worst case dataset will result in $(n^2 + n)/2$ evaluations of the dependency function. However, as fuzzy-rough set-based feature selection is used for dimensionality reduction prior to any involvement of the system which will employ those features belonging to the resultant reduct, this operation has no negative impact upon the run-time efficiency of the system.

3.4 A Worked Example

Table 1 contains three real-valued conditional attributes and a crisp-valued decision attribute. To begin with, the fuzzy-rough QuickReduct algorithm initializes the potential reduct (i.e. the current best set of attributes) to the empty set.

Using the fuzzy sets defined in Fig. 3 (for all conditional attributes), and setting $A = \{a\}$, $B = \{b\}$, $C = \{c\}$ and $Q = \{q\}$, the following equivalence classes are obtained:

Object	a	b	c	q
1	-0.4	-0.3	-0.5	no
2	-0.4	0.2	-0.1	yes
3	-0.3	-0.4	-0.3	no
4	0.3	-0.3	0	yes
5	0.2	-0.3	0	yes
6	0.2	0	0	no

Table 1. Example dataset: crisp decisions

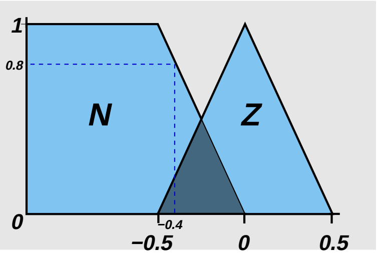

Fig. 3. Fuzzifications for conditional features

$$\mathbb{U}/A = \{N_a, Z_a\}$$
$$\mathbb{U}/B = \{N_b, Z_b\}$$
$$\mathbb{U}/C = \{N_c, Z_c\}$$
$$\mathbb{U}/Q = \{\{1,3,6\}, \{2,4,5\}\}$$

The first step is to calculate the lower approximations of the sets A, B and C, using (13). To clarify the calculations involved, table 2 contains the membership degrees of objects to fuzzy equivalence classes. For simplicity, only A will be considered here; that is, using A to approximate Q. For the first decision equivalence class X = {1,3,6}, $\mu_{\underline{A}\{1,3,6\}}(x)$ needs to be calculated:

$$\mu_{\underline{A}\{1,3,6\}}(x) = \sup_{F\in\mathbb{U}/A} min(\mu_F(x), \inf_{y\in\mathbb{U}} max\{1-\mu_F(y), \mu_{\{1,3,6\}}(y)\})$$

Considering the first fuzzy equivalence class of A, N_a:

$$min(\mu_{N_a}(x), \inf_{y\in\mathbb{U}} max\{1-\mu_{N_a}(y), \mu_{\{1,3,6\}}(y)\})$$

For object 2 this can be calculated as follows. From table 2 it can be seen that the membership of object 2 to the fuzzy equivalence class N_a, $\mu_{N_a}(2)$, is 0.8. The remainder of the calculation involves finding the smallest of the following values:

Table 2. Membership values of objects to corresponding fuzzy sets

Object	a		b		c		q	
	N_a	Z_a	N_b	Z_b	N_c	Z_c	$\{1,3,6\}$	$\{2,4,5\}$
1	0.8	0.2	0.6	0.4	1.0	0.0	1.0	0.0
2	0.8	0.2	0.0	0.6	0.2	0.8	0.0	1.0
3	0.6	0.4	0.8	0.2	0.6	0.4	1.0	0.0
4	0.0	0.4	0.6	0.4	0.0	1.0	0.0	1.0
5	0.0	0.6	0.6	0.4	0.0	1.0	0.0	1.0
6	0.0	0.6	0.0	1.0	0.0	1.0	1.0	0.0

$$\begin{gathered}
\max(1\text{-}\mu_{N_a}(1),\ \mu_{\{1,3,6\}}(1)) = \max(0.2,1.0) = 1.0 \\
\max(1\text{-}\mu_{N_a}(2),\ \mu_{\{1,3,6\}}(2)) = \max(0.2,0.0) = 0.2 \\
\max(1\text{-}\mu_{N_a}(3),\ \mu_{\{1,3,6\}}(3)) = \max(0.4,1.0) = 1.0 \\
\max(1\text{-}\mu_{N_a}(4),\ \mu_{\{1,3,6\}}(4)) = \max(1.0,0.0) = 1.0 \\
\max(1\text{-}\mu_{N_a}(5),\ \mu_{\{1,3,6\}}(5)) = \max(1.0,0.0) = 1.0 \\
\max(1\text{-}\mu_{N_a}(6),\ \mu_{\{1,3,6\}}(6)) = \max(1.0,1.0) = 1.0
\end{gathered}$$

From the calculations above, the smallest value is 0.2, hence:

$$\begin{aligned}
min(\mu_{N_a}(x), \inf_{y\in\mathbb{U}} max\{1-\mu_{N_a}(y), \mu_{\{1,3,6\}}(y)\}) &= min(0.8, \inf\{1, 0.2, 1, 1, 1, 1\}) \\
&= 0.2
\end{aligned}$$

Similarly for Z_a

$$\begin{aligned}
min(\mu_{Z_a}(x), \inf_{y\in\mathbb{U}} max\{1-\mu_{Z_a}(y), \mu_{\{1,3,6\}}(y)\}) &= min(0.2, \inf\{1, 0.8, 1, 0.6, 0.4, 1\} \\
&= 0.2
\end{aligned}$$

Thus,

$$\mu_{\underline{A}\{1,3,6\}}(2) = 0.2$$

Calculating the A-lower approximation of $X = \{1, 3, 6\}$ for every object gives

$$\begin{aligned}
\mu_{\underline{A}\{1,3,6\}}(1) &= 0.2 & \mu_{\underline{A}\{1,3,6\}}(2) &= 0.2 \\
\mu_{\underline{A}\{1,3,6\}}(3) &= 0.4 & \mu_{\underline{A}\{1,3,6\}}(4) &= 0.4 \\
\mu_{\underline{A}\{1,3,6\}}(5) &= 0.4 & \mu_{\underline{A}\{1,3,6\}}(6) &= 0.4
\end{aligned}$$

The corresponding values for $X = \{2, 4, 5\}$ can also be determined:

$$\begin{aligned}
\mu_{\underline{A}\{2,4,5\}}(1) &= 0.2 & \mu_{\underline{A}\{2,4,5\}}(2) &= 0.2 \\
\mu_{\underline{A}\{2,4,5\}}(3) &= 0.4 & \mu_{\underline{A}\{2,4,5\}}(4) &= 0.4 \\
\mu_{\underline{A}\{2,4,5\}}(5) &= 0.4 & \mu_{\underline{A}\{2,4,5\}}(6) &= 0.4
\end{aligned}$$

It is a coincidence here that $\mu_{\underline{A}\{2,4,5\}}(x) = \mu_{\underline{A}\{1,3,6\}}(x)$ for this example. Using these values, the fuzzy positive region for each object can be calculated via using

$$\mu_{POS_A(Q)}(x) = \sup_{X\in\mathbb{U}/Q} \mu_{\underline{A}X}(x)$$

This results in:

$$\begin{array}{ll} \mu_{POS_A(Q)}(1) = 0.2 & \mu_{POS_A(Q)}(2) = 0.2 \\ \mu_{POS_A(Q)}(3) = 0.4 & \mu_{POS_A(Q)}(4) = 0.4 \\ \mu_{POS_A(Q)}(5) = 0.4 & \mu_{POS_A(Q)}(6) = 0.4 \end{array}$$

The next step is to determine the degree of dependency of Q on A:

$$\gamma'_A(Q) = \frac{\sum_{x \in U} \mu_{POS_A(Q)}(x)}{|U|} = 2/6$$

Calculating for B and C gives:

$$\gamma'_B(Q) = \frac{2.4}{6}, \;\; \gamma'_C(Q) = \frac{1.6}{6}$$

From this it can be seen that attribute b will cause the greatest increase in dependency degree. This attribute is chosen and added to the potential reduct. The process iterates and the two dependency degrees calculated are

$$\gamma'_{\{a,b\}}(Q) = \frac{3.4}{6}, \;\; \gamma'_{\{b,c\}}(Q) = \frac{3.2}{6}$$

Adding attribute a to the reduct candidate causes the larger increase of dependency, so the new candidate becomes $\{a, b\}$. Lastly, attribute c is added to the potential reduct:

$$\gamma'_{\{a,b,c\}}(Q) = \frac{3.4}{6}$$

As this causes no increase in dependency, the algorithm stops and outputs the reduct $\{a, b\}$. The dataset can now be reduced to only those attributes appearing in the reduct. When crisp RSAR is performed on this dataset (after using the same fuzzy sets to discretize the real-valued attributes), the reduct generated is $\{a, b, c\}$, i.e. the full conditional attribute set. Unlike crisp RSAR, the true minimal reduct was found using the information on degrees of membership. It is clear from this example alone that the information lost by using crisp RSAR can be important when trying to discover the smallest reduct from a dataset.

Conventional hill-climbing approaches to feature selection such as the algorithm presented above often fail to find maximal data reductions or minimal reducts. Some guiding heuristics are better than others for this, but as no perfect heuristic exists there can be no guarantee of optimality. When maximal data reductions are required, other search mechanisms must be employed. Although these methods also cannot ensure optimality, they provide a means by which the best feature subsets might be found. This motivates the development of feature selection based on Ant Colony Optimization.

4 Ant-based Feature Selection

Swarm Intelligence (SI) is the property of a system whereby the collective behaviours of simple agents interacting locally with their environment cause coherent functional global patterns to emerge [2]. SI provides a basis with which it is possible to explore collective (or distributed) problem solving without centralized control or the provision of a global model. One area of interest in SI is Particle Swarm Optimization [17], a population-based stochastic optimization technique. Here, the system is initialised with a population of random solutions, called particles. Optima are searched for by updating generations, with particles moving through the parameter space towards the current local and global optimum particles. At each time step, the velocities of all particles are changed depending on the current optima.

Ant Colony Optimization (ACO) [2] is another area of interest within SI. In nature, it can be observed that real ants are capable of finding the shortest route between a food source and their nest without the use of visual information and hence possess no global world model, adapting to changes in the environment. The deposition of pheromone is the main factor in enabling real ants to find the shortest routes over a period of time. Each ant probabilistically prefers to follow a direction rich in this chemical. The pheromone decays over time, resulting in much less pheromone on less popular paths. Given that over time the shortest route will have the higher rate of ant traversal, this path will be reinforced and the others diminished until all ants follow the same, shortest path (the "system" has converged to a single solution). It is also possible that there are many equally short paths. In this situation, the rates of ant traversal over the short paths will be roughly the same, resulting in these paths being maintained while others are ignored. Additionally, if a sudden change to the environment occurs (e.g. a large obstacle appears on the shortest path), the ACO system can respond to this and will eventually converge to a new solution. Based on this idea, artificial ants can be deployed to solve complex optimization problems via the use of artificial pheromone deposition.

ACO is particularly attractive for feature selection as there seems to be no heuristic that can guide search to the optimal minimal subset every time. Additionally, it can be the case that ants discover the best feature combinations as they proceed throughout the search space. This section discusses how ACO may be applied to the difficult problem of finding optimal feature subsets and, in particular, fuzzy-rough set-based reducts.

4.1 ACO Framework

An ACO algorithm can be applied to any combinatorial problem as far as it is possible to define:

- *Appropriate problem representation.* The problem can be described as a graph with a set of nodes and edges between nodes.
- *Heuristic desirability (η) of edges.* A suitable heuristic measure of the "goodness" of paths from one node to every other connected node in the graph.

- *Construction of feasible solutions.* A mechanism must be in place whereby possible solutions are efficiently created. This requires the definition of a suitable traversal stopping criterion to stop path construction when a solution has been reached.
- *Pheromone updating rule.* A suitable method of updating the pheromone levels on edges is required with a corresponding evaporation rule, typically involving the selection of the n best ants and updating the paths they chose.
- *Probabilistic transition rule.* The rule that determines the probability of an ant traversing from one node in the graph to the next.

Each ant in the artificial colony maintains a memory of its history - remembering the path it has chosen so far in constructing a solution. This history can be used in the evaluation of the resulting created solution and may also contribute to the decision process at each stage of solution construction.

Two types of information are available to ants during their graph traversal, local and global, controlled by the parameters β and α respectively. Local information is obtained through a problem-specific heuristic measure. The extent to which the measure influences an ant's decision to traverse an edge is controlled by the parameter β. This will guide ants towards paths that are likely to result in good solutions. Global knowledge is also available to ants through the deposition of artificial pheromone on the graph edges by their predecessors over time. The impact of this knowledge on an ant's traversal decision is determined by the parameter α. Good paths discovered by past ants will have a higher amount of associated pheromone. How much pheromone is deposited, and when, is dependent on the characteristics of the problem. No other local or global knowledge is available to the ants in the standard ACO model, though the inclusion of such information by extending the ACO framework has been investigated [2].

4.2 Feature Selection

The feature selection task may be reformulated into an ACO-suitable problem [13, 16]. ACO requires a problem to be represented as a graph - here nodes represent features, with the edges between them denoting the choice of the next feature. The search for the optimal feature subset is then an ant traversal through the graph where a minimum number of nodes are visited that satisfies the traversal stopping criterion. Figure 4 illustrates this setup - the ant is currently at node a and has a choice of which feature to add next to its path (dotted lines). It chooses feature b next based on the transition rule, then c and then d. Upon arrival at d, the current subset $\{a, b, c, d\}$ is determined to satisfy the traversal stopping criteria (e.g. a suitably high classification accuracy has been achieved with this subset, assuming that the selected features are used to classify certain objects). The ant terminates its traversal and outputs this feature subset as a candidate for data reduction.

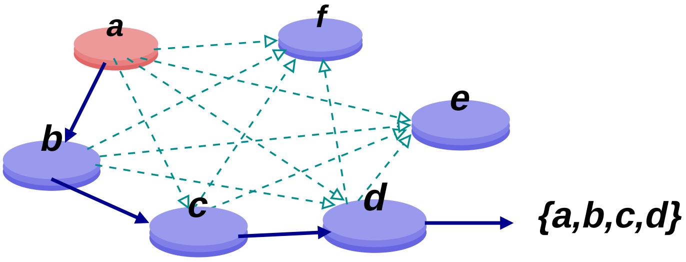

Fig. 4. ACO problem representation for feature selection

A suitable heuristic desirability of traversing between features could be any subset evaluation function - for example, an entropy-based measure [24] or the fuzzy-rough set dependency measure. Depending on how optimality is defined for the particular application, the pheromone may be updated accordingly. For instance, subset minimality and "goodness" are two key factors so the pheromone update should be proportional to "goodness" and inversely proportional to size. How "goodness" is determined will also depend on the application. In some cases, this may be a heuristic evaluation of the subset, in others it may be based on the resulting classification accuracy of a classifier produced using the subset.

The heuristic desirability and pheromone factors are combined to form the so-called probabilistic transition rule, denoting the probability of an ant k at feature i choosing to move to feature j at time t:

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha}.[\eta_{ij}]^{\beta}}{\sum_{l \in J_i^k}[\tau_{il}(t)]^{\alpha}.[\eta_{il}]^{\beta}} \tag{19}$$

where J_i^k is the set of ant k's unvisited features, η_{ij} is the heuristic desirability of choosing feature j when at feature i and $\tau_{ij}(t)$ is the amount of virtual pheromone on edge (i, j). The choice of α and β is determined experimentally. Typically, several experiments are performed, varying each parameter and choosing the values that produce the best results.

Selection Process

The overall process of ACO feature selection can be seen in Fig. 5. It begins by generating a number of ants, k, which are then placed randomly on the graph (i.e. each ant starts with one random feature). Alternatively, the number of ants to place on the graph may be set equal to the number of features within the data; each ant starts path construction at a different feature. From these initial positions, they traverse edges probabilistically until a traversal stopping criterion is satisfied. The resulting subsets are gathered and then evaluated. If an optimal subset has been found or the algorithm has executed a certain

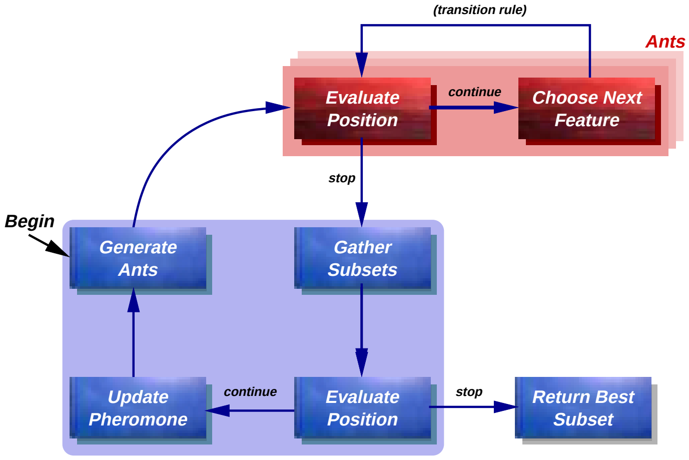

Fig. 5. ACO-based feature selection overview

number of times, then the process halts and outputs the best feature subset encountered. If neither condition holds, then the pheromone is updated, a new set of ants are created and the process iterates once more.

Complexity Analysis

The time complexity of the ant-based approach to feature selection is $O(IAk)$, where I is the number of iterations, A the number of original features, and k the number of ants. In the worst case, each ant selects all the features. As the heuristic is evaluated after each feature is added to the reduct candidate, this will result in A evaluations per ant. After one iteration in this scenario, Ak evaluations will have been performed. After I iterations, the heuristic will be evaluated IAk times.

Pheromone Update

Depending on how optimality is defined for the particular application, the pheromone may be updated accordingly. To tailor this mechanism to find fuzzy-rough set reducts, it is necessary to use the dependency measure given in (18) as the stopping criterion. This means that an ant will stop building its feature subset when the dependency of the subset reaches the maximum for the dataset (the value 1 for consistent datasets). The dependency function may also be chosen as the heuristic desirability measure, but this is not necessary. In fact, it may be of more use to employ a non-rough set related heuristic for this purpose. By using an alternative measure such as an entropy-based heuristic, the method may avoid feature combinations that may mislead the fuzzy-rough set-based heuristic. Again, the time complexity of this fuzzy-rough ant-based method will be the same as that mentioned earlier, $O(IAk)$.

The pheromone on each edge is updated according to the following formula:

$$\tau_{ij}(t+1) = (1-\rho).\tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{20}$$

where

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{n}(\gamma'(S^k)/|S^k|) \tag{21}$$

This is the case if the edge (i, j) has been traversed; $\Delta\tau_{ij}(t)$ is 0 otherwise. The value ρ is a decay constant used to simulate the evaporation of the pheromone, S^k is the feature subset found by ant k. The pheromone is updated according to both the rough (or fuzzy-rough) measure of the "goodness" of the ant's feature subset (γ') and the size of the subset itself. By this definition, all ants update the pheromone. Alternative strategies may be used for this, such as allowing only the ants with the currently best feature subsets to proportionally increase the pheromone.

5 Crisp Ant-based Feature Selection Evaluation

In order to compare several mainstream approaches to crisp rough set-based feature selection with ACO-based selection, an investigation into how these methods perform in terms of resulting subset optimality has been carried out here. Several real and artificial datasets are used for this purpose. In particular, it is interesting to compare those methods that employ an incremental-based search strategy with those that adopt a more complex stochastic/probabilistic mechanism. Five techniques for finding crisp rough set reducts are tested here on 13 datasets. These techniques are: RSAR (using QuickReduct), EBR (an entropy-based approach [15]), GenRSAR (genetic algorithm-based), AntRSAR (ant-based) and SimRSAR (simulated annealing-based)[1].

5.1 Experimental Setup

Before the experiments are described, a few points must be made about the later three approaches, GenRSAR, AntRSAR and SimRSAR.

GenRSAR employs a genetic search strategy in order to determine rough set reducts. The initial population consists of 100 randomly generated feature subsets, the probabilities of mutation and crossover are set to 0.4 and 0.6 respectively, and the number of generations is set to 100. The fitness function

[1] These algorithms and datasets (as well as FRFS and antFRFS) can be downloaded from the webpage: http://users.aber.ac.uk/rkj/index.html

considers both the size of subset and its evaluated suitability, and is defined as follows:

$$fitness(R) = \gamma_R(\mathbb{D}) * \frac{|\mathbb{C}| - |R|}{|\mathbb{C}|} \tag{22}$$

AntRSAR follows the mechanism described in section 4.2. Here, the precomputed heuristic desirability of edge traversal is the entropy measure, with the subset evaluation performed using the rough set dependency heuristic (to guarantee that true rough set reducts are found). The number of ants used is set to the number of features, with each ant starting on a different feature. For the datasets used here, the performance is not affected significantly using this number of ants. However, for datasets containing thousands of features or more, fewer ants may have to be chosen due to computational limitations. Ants construct possible solutions until they reach a rough set reduct. To avoid fruitless searches, the size of the current best reduct is used to reject those subsets whose cardinality exceed this value. Pheromone levels are set at 0.5 with a small random variation added. Levels are increased by only those ants who have found true reducts. The global search is terminated after 250 iterations, α is set to 1 and β is set to 0.1.

SimRSAR employs a simulated annealing-based feature selection mechanism [15]. The states are feature subsets, with random state mutations set to changing three features (either adding or removing them). The cost function attempts to maximize the rough set dependency (γ) whilst minimizing the subset cardinality. For these experiments, the cost of subset R is defined as:

$$cost(R) = \left[\frac{\gamma_{\mathbb{C}}(\mathbb{D}) - \gamma_R(\mathbb{D})}{\gamma_{\mathbb{C}}(\mathbb{D})}\right]^a + \left[\frac{|R|}{|\mathbb{C}|}\right]^b \tag{23}$$

where a and b are defined in order to weight the contributions of dependency and subset size to the overall cost measure. In the experiments here, $a = 1$ and $b = 3$. The initial temperature of the system is estimated as $2 * |\mathbb{C}|$ and the cooling schedule is $T(t+1) = 0.93 * T(t)$.

The experiments were carried out on 3 datasets from [25], namely *m-of-n, exactly* and *exactly2*. The remaining datasets are from the machine learning repository [1]. Those datasets containing real-valued attributes have been discretized to allow all methods to be compared fairly.

5.2 Experimental Results

Table 3 presents the results of the five methods on the 13 datasets. It shows the size of reduct found for each method, as well as the size of the optimal (minimal) reduct. RSAR and EBR produced the same subset every time, unlike AntRSAR and SimRSAR that often found different subsets and sometimes different subset cardinalities. On the whole, it appears to be the case that AntRSAR and SimRSAR outperform the other three methods. This is at the

Table 3. Subset sizes found for five techniques

Index	Dataset	Features	Optimal	RSAR	EBR	AntRSAR	SimRSAR	GenRSAR
0	M-of-N	13	6	8	6	6	6	6(6) 7(12)
1	Exactly	13	6	9	8	6	6	6(10) 7(10)
2	Exactly2	13	10	13	11	10	10	10(9) 11(11)
3	Heart	13	6	7	7	6(18) 7(2)	6(29) 7(1)	6(18) 7(2)
4	Vote	16	8	9	9	8	8(15) 9(15)	8(2) 9(18)
5	Credit	20	8	9	10	8(12) 9(4) 10(4)	8(18) 9(1) 11(1)	10(6) 11(14)
6	Mushroom	22	4	5	4	4	4	5(1) 6(5) 7(14)
7	LED	24	5	12	5	5(12) 6(4) 7(3)	5	6(1) 7(3) 8(16)
8	Letters	25	8	9	9	8	8	8(8) 9(12)
9	Derm	34	6	7	6	6(17) 7(3)	6(12) 7(8)	10(6) 11(14)
10	Derm2	34	8	10	10	8(3) 9(17)	8(3) 9(7)	10(2) 11(8)
11	WQ	38	12	14	14	12(2) 13(7) 14(11)	13(16) 14(4)	16
12	Lung	56	4	4	4	4	4(7) 5(12) 6(1)	6(8) 7(12)

expense of the time taken to discover these reducts as can be seen in Fig. 6 (results for RSAR and EBR do not appear as they are consistently faster than the other methods). In all experiments the rough ordering of techniques with respect to time is: RSAR $<$ EBR $\leq$ SimRSAR $\leq$ AntRSAR $\leq$ GenRSAR. AntRSAR and SimRSAR perform similarly throughout - for some datasets, AntRSAR is better (e.g. Vote) and for others SimRSAR is best (e.g. LED). The performance of these two methods may well be improved by fine-tuning the parameters to each individual dataset.

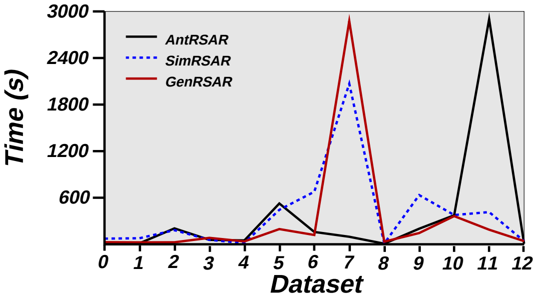

Fig. 6. Average runtimes for AntRSAR, SimRSAR and GenRSAR

From these results it can be seen that even for small and medium-sized datasets, incremental hill-climbing techniques often fail to find minimal subsets. For example, RSAR is misled early in the search for the LED dataset, resulting in it choosing 7 extraneous features. Although this fault is due to the non-optimality of the guiding heuristic, a perfect heuristic does not exist rendering these approaches unsuited to problems where a minimal subset is essential. However, for most real world applications, the extent of reduction achieved via such methods is acceptable. For systems where the minimal subset is required (perhaps due to the cost of feature measurement), stochastic feature selection should be used.

6 Fuzzy Ant-based Feature Selection Evaluation

To show the utility of fuzzy-rough feature selection and to compare the hill-climbing and ant-based fuzzy-rough approaches, the two methods are applied as pre-processors to web classification and within a complex systems monitoring application. Both methods preserve the semantics of the surviving features after removing 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.

6.1 Web Classification

There are an estimated 1 billion web pages available on the world wide web, with around 1.5 million web pages being added every day. The task to find a particular web page, which satisfies a user's requirements by traversing hyperlinks, is very difficult. To aid this process, many web directories have been developed - some rely on manual categorization whilst others make decisions automatically. However, as web page content is vast and dynamic, manual categorization is becoming increasingly impractical. Automatic web site categorization is therefore required to deal with these problems.

System Overview

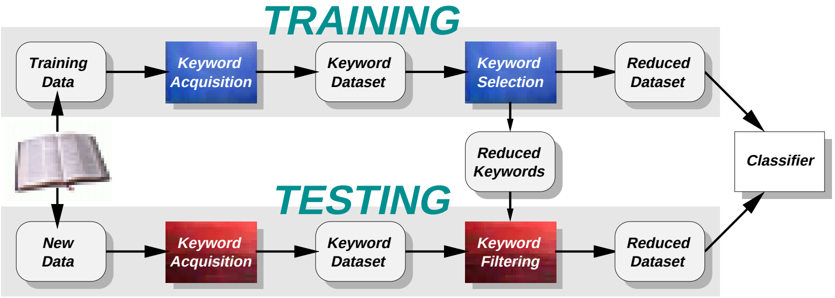

Fig. 7. Modular decomposition of the classification system

The general overview of the classification system developed here can be seen in Fig. 7. A key issue in the design of the system was that of modularity; it should be able to integrate with existing (or new) techniques. The current implementations allow this flexibility by dividing the overall process into several independent sub-modules:

- *Keyword Acquisition.* From the collection of web documents, only the natural language terms are extracted and considered to be keywords. These are then weighted according to their perceived importance in the document, resulting in a new dataset of weight-term pairs. These weights are almost always real-valued, hence the problem serves well to test the present work. For this, the TF-IDF metric [27] is used which assigns higher weights to those keywords that occur frequently in the current document but not in most others. Note that in this work, no sophisticated keyword acquisition techniques methods are used as the current focus of attention is on the evaluation of attribute reduction. However, the use of more effective keyword acquisition techniques recently built in the area of information retrieval would help improve the system's overall classification performance further.

- *Keyword Selection.* As the newly generated datasets are too large, mainly due to keyword redundancy, a dimensionality reduction step is carried out using the techniques described previously.

- *Keyword Filtering.* Employed only in testing, this simple module filters the keywords obtained during acquisition, using the reduct generated in the keyword selection module.

- *Classification.* This final module uses the reduced dataset to perform the actual categorization of the test data. Four classifiers were used for comparison, namely C4.5 [24], JRip [5], PART [33] and a fuzzy rule inducer, QSBA [26]. Both JRip and PART are available from [34].
  C4.5 creates decision trees by choosing the most informative features and recursively partitioning the data into subtables based on their values. Each node in the tree represents a feature with branches from a node representing the alternative values this feature can take according to the current subtable. Partitioning stops when all data items in the subtable have the same classification. A leaf node is then created, and this classification assigned.
  JRip learns propositional rules by repeatedly growing rules and pruning them. During the growth phase, antecedents are added greedily until a termination condition is satisfied. Antecedents are then pruned in the next phase subject to a pruning metric. Once the ruleset is generated, a further optimization is performed where rules are evaluated and deleted based on their performance on randomized data.

- PART generates rules by means of repeatedly creating partial decision trees from data. The algorithm adopts a separate-and-conquer strategy in that it removes instances covered by the current ruleset during processing. Essentially, a rule is created by building a pruned tree for the current set of instances; the leaf with the highest coverage is made into a rule.
- QSBA induces fuzzy rules by calculating the fuzzy subsethood of linguistic terms and the corresponding decision variables. These values are also weighted by the use of fuzzy quantifiers. This method utilises the same fuzzy sets as those involved in the fuzzy-rough reduction methods.

Experimentation and Results

Initially, datasets were generated from large textual corpora collected from Yahoo [36] and separated randomly into training and testing sets. Each dataset is a collection of web documents. Five classification categories were used, namely Art & Humanity, Entertainment, Computers & Internet, Health, Business & Economy. A total of 280 web sites were collected from Yahoo categories and classified into these categories. From this collection of data, the keywords, weights and corresponding classifications were collated into a single dataset.

Table 4 shows the resulting degree of dimensionality reduction, performed via selecting informative keywords, by the standard fuzzy-rough method (FRFS) and the ACO-based approach (AntFRFS). AntFRFS is run several times, and the results averaged both for classification accuracy and number of features selected. It can be seen that both methods drastically reduce the number of original features. AntFRFS performs the highest degree of reduction, with an average of 14.1 features occurring in the reducts it locates.

Table 4. Extent of feature reduction

Original	FRFS	AntFRFS
2557	17	14.10

To see the effect of dimensionality reduction on classification accuracy, the system was tested on the original training data and a test dataset. The results are summarised in table 5. Clearly, the fuzzy-rough methods exhibit better resultant accuracies for the test data than the unreduced method for all classifiers. This demonstrates that feature selection using either FRFS or AntFRFS can greatly aid classification tasks. It is of additional benefit to rule inducers as the induction time is decreased and the generated rules involve significantly fewer features. AntFRFS improves on FRFS in terms of the size of subsets found and resulting testing accuracy for QSBA and PART, but not for C4.5 and JRip.

The challenging nature of this particular task can be seen in the overall low accuracies produced by the classifiers, though improved somewhat after fea-

Table 5. Classification performance

	Original (%)		FRFS (%)		AntFRFS (%)	
Classifier	Train	Test	Train	Test	Train	Test
C4.5	95.89	44.74	86.30	57.89	81.27	48.39
QSBA	100.0	39.47	82.19	46.05	69.86	50.44
JRip	72.60	56.58	78.08	60.53	64.84	51.75
PART	95.89	42.11	86.30	48.68	82.65	48.83

ture selection. Both fuzzy-rough approaches require a reasonable fuzzification of the input data, whilst the fuzzy sets are herein generated by simple statistical analysis of the dataset with no attempt made at optimizing these sets. A fine-tuned fuzzification will certainly improve the performance of FRFS-based systems. Finally, it is worth noting that the classifications were checked automatically. Many websites can be classified to more than one category, however only the designated category is considered to be correct here.

6.2 Systems Monitoring

In order to further evaluate the fuzzy-rough approaches and to illustrate its domain-independence, another challenging test dataset was chosen, namely the Water Treatment Plant Database [1]. The dataset itself is a set of historical data charted over 521 days, with 38 different input features measured daily. Each day is classified into one of thirteen categories depending on the operational status of the plant. However, these can be collapsed into just two or three categories (i.e. *Normal* and *Faulty*, or *OK*, *Good* and *Faulty*) for plant monitoring purposes as many classifications reflect similar performance. 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. Collapsing 13 categories into 2 or 3 classes helps reduce this difficulty for the present application. Note that this dataset has been utilised in many previous studies, including that reported in [29] (to illustrate the effectiveness of applying crisp RSAR as a pre-processing step to rule induction).

It is likely that not all of the 38 input features are required to determine the status of the plant, hence the dimensionality reduction step. However, choosing the most informative features is a difficult task as there will be many dependencies between subsets of features. There is also a monetary cost involved in monitoring these inputs, so it is desirable to reduce this number.

Note that the original monitoring system (Fig. 8) developed in [29] consisted of several modules; it is this modular structure that allows the FRFS techniques to replace the existing crisp method. Originally, a precategorization step preceded feature selection where feature values were quantized. To

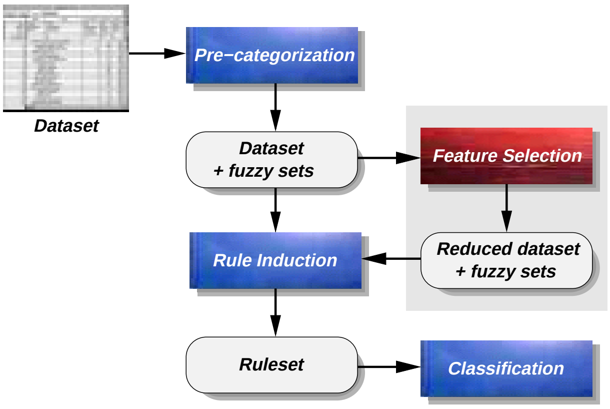

Fig. 8. Modular decomposition of the implemented system

reduce potential loss of information, the original use of just the dominant symbolic labels of the discretized fuzzy terms is now replaced by a fuzzification procedure. This leaves the underlying feature values unchanged but generates a series of fuzzy sets for each feature. These sets are generated entirely from the data while exploiting the statistical properties attached to the dataset (in keeping with the rough set ideology in that the dependence of learning upon information provided outside of the training dataset is minimized). This module may be replaced by alternative fuzzifiers, or expert-defined fuzzification if available. Based on these fuzzy sets and the original real-valued dataset, the feature selection module calculates a reduct and reduces the dataset accordingly. Finally, rule induction is performed on the reduced dataset. For this set of experiments, the decision tree method C4.5 [24] is used for induction and the learned rules for classification.

The first set of experiments compares the hill-climbing and ACO-based fuzzy-rough methods. An investigation into another feature selector based on the entropy measure is then presented. This is followed by comparisons with a transformation-based dimensionality reduction approach, PCA [7] and a support vector classifier [22].

Comparison of Fuzzy-Rough Methods

Three sets of experiments were carried out on both the (collapsed) 2-class and 3-class datasets. The first bypasses the feature selection part of the system, using the original water treatment dataset as input to C4.5, with all 38 conditional attributes. The second method employs FRFS to perform the feature selection before induction is carried out. The third uses the ACO-based method, AntFRFS, to perform feature selection over a number of runs, and the results averaged.

Table 6. Results for the 2-class dataset

Method	Attributes	γ' value	Training accuracy (%)	Testing accuracy (%)
Unreduced	38	-	98.5	80.9
FRFS	10	0.58783	89.2	74.8
AntFRFS	9.55	0.58899	93.5	77.9

The results for the 2-class dataset can be seen in table 6. Both FRFS and AntFRFS significantly reduce the number of original attributes with AntFRFS producing the greatest data reduction on average. As well as generating smaller reducts, AntFRFS finds reducts of a higher quality according to the fuzzy-rough dependency measure. This higher quality is reflected in the resulting classification accuracies for both the training and testing datasets, with AntFRFS outperforming FRFS.

Table 7. Results for the 3-class dataset

Method	Attributes	γ' value	Training accuracy (%)	Testing accuracy (%)
Unreduced	38	-	97.9	83.2
FRFS	11	0.59479	97.2	80.9
AntFRFS	9.09	0.58931	94.8	80.2

Table 7 shows the results for the 3-class dataset experimentation. The hill-climbing fuzzy-rough method chooses 11 out of the original 38 features. The ACO-based method chooses fewer attributes on average, however this is at the cost of a lower dependency measure for the generated reducts. Again the effect of this can be seen in the classification accuracies, with FRFS performing slightly better than AntFRFS. For both fuzzy methods, the small drop in accuracy as a result of feature selection is acceptable.

By selecting a good feature subset from data it is usually expected that the applied learning method should benefit, producing an improvement in results. For some applications, less features may result in a better classification performance due to the removal of heavy noise attached to those features removed. The ant-based approach should improve upon C4.5 in these situations. However, when the original training (and test) data is very noisy, selected features may not necessarily be able to reflect all the information contained within the original entire feature set. As a result of removing less informative features, partial useful information may be lost. The goal of selection methods in this situation is to minimise this loss, while reducing the number of features to the greatest extent. Therefore, it is not surprising that the classification performance for this challenging dataset can decrease upon data reduction, as shown in table 7. However, the impact of feature selection can have different effects on different classifiers. With the use of an alternative classifier in section 6.2, performance can be seen to improve for the test data.

The results here also show a marked drop in classification accuracy for the test data. This could be due to the problems encountered when dealing with datasets of small sample size. Overfitting can occur, where a learning algorithm adapts so well to a training set, that the random disturbances present are included in the model as being meaningful. Consequently, as these disturbances do not reflect the underlying distribution, the performance on the test data will suffer. Although such techniques as cross-validation and bootstrapping have been proposed as a way of countering this, these still often exhibit high variance in error estimation.

Comparison with Entropy-based Feature Selection

To support the study of the performance of the fuzzy-rough methods for use as pre-processors to rule induction, a conventional entropy-based technique is used for comparison. This approach utilizes the entropy heuristic typically employed by machine learning techniques such as C4.5 [24]. Those features that provide the most gain in information are selected. A summary of the results of this comparison can be seen in table 8.

Table 8. Results for the three selection methods

Approach	No. of Classes	No. of Features	Training Accuracy (%)	Testing Accuracy (%)
FRFS	2	10	89.2	74.8
AntFRFS	2	9.55	93.5	77.9
Entropy	2	13	97.7	80.2
FRFS	3	11	97.2	80.9
AntFRFS	3	9.09	94.8	80.2
Entropy	3	14	98.2	80.9

For both the 2-class and 3-class datasets, FRFS and AntFRFS select at least three fewer features than the entropy-based method. However, the entropy-based method outperforms the other two feature selectors with the resulting C4.5 classification accuracies. This is probably due to the fact that C4.5 uses exactly the same entropy measure in generating decision trees. In this case, the entropy-based measure will favour those attributes that will be the most influential in the decision tree generation process. The use of more features here may also contribute to the slightly better classification performance.

Comparison with the use of PCA

The effect of using a different dimensionality reduction technique, namely Principal Components Analysis (PCA) [7], is also investigated. PCA transforms the original features of a dataset with a (typically) reduced number of uncorrelated ones, termed principal components. It works on the assumption that a large feature variance corresponds to useful information, with small variance equating to information that is less useful. The first principle component indicates the direction of maximum data variance. Data is transformed in such a way as to allow the removal of those transformed features with small variance. This is achieved by finding the eigenvectors of the covariance matrix of data points (objects), constructing a transformation matrix from the ordered eigenvectors, and transforming the original data by matrix multiplication.

Here, PCA is applied to the dataset and the first n principal components are used. A range of values is chosen for n to investigate how the performance varies with dimensionality. As PCA irreversibly destroys the underlying dataset semantics, the resulting decision trees are not human-comprehensible nor directly measurable but may still provide useful automatic classifications of new data. Table 9 shows the results from applying PCA to the datasets.

Table 9. Results for the 2-class and 3-class datasets using PCA

		No. of Features								
Accuracy	Class	5	6	7	8	9	**10**	11	12	13
Training (%)	2	80.0	80.0	80.0	80.0	80.3	**80.3**	80.3	80.8	82.1
Testing (%)	2	72.5	72.5	72.5	72.5	73.3	**73.3**	73.3	35.1	34.4
Training (%)	3	73.6	73.6	73.6	73.6	73.6	**75.9**	75.9	75.9	76.4
Testing (%)	3	80.9	80.9	80.9	80.9	80.9	**80.9**	80.9	80.9	80.2

Both AntFRFS and FRFS significantly outperform PCA on the 2-class dataset. Of particular interest is when 10 principal components are used as this is roughly the same number chosen by AntFRFS and FRFS. The resulting accuracy for PCA is 80.3% for the training data and 73.3% for the test data. For AntFRFS the accuracies were 93.5% (training) and 77.9% (testing), and for FRFS 89.2% (training) and 74.8% (testing). In the 3-class dataset experimentation, both fuzzy-rough methods produce much higher classification accuracies than PCA for the training data. For the test data, the performance is about the same, with PCA producing a slightly higher accuracy than AntFRFS on the whole. It is worth reiterating, however, that PCA does not carry out feature selection but transformation. Hence, the classifier built with such transformed features is hard for human users to understand.

Comparison with the use of a Support Vector Classifier

A possible limitation of employing C4.5 in this context is that it performs a degree of feature selection itself during the induction process. The resulting decision trees do not necessarily contain all the features present in the original training data. As a result of this, it is beneficial to evaluate the use of an alternative classifier that uses all the given features. For this purpose, a support vector classifier [28] is employed, trained by the sequential minimal optimization (SMO) algorithm [22]. The results of the application of this classifier can be found in table 10.

Table 10. Results for the 2-class and 3-class datasets using SMO

Approach	No. of Classes	No. of Features	Training Accuracy (%)	Testing Accuracy (%)
Unreduced	2	38	80.0	71.8
FRFS	2	10	80.0	72.5
AntFRFS	2	9.55	80.0	72.5
Unreduced	3	38	74.6	80.9
FRFS	3	11	73.6	80.2
AntFRFS	3	9.09	73.6	80.9

For the 2-class dataset, the training accuracy for both FRFS and AntFRFS is the same as that of the unreduced approach. However, this is with significantly fewer attributes. Additionally, the resulting testing accuracy *is* increased with these feature selection methods. With the more challenging 3-class problem, the training accuracies are slightly worse (as seen with the C4.5 analysis). The AntFRFS method performs better than FRFS for the test data and is equal to the unreduced method, again using fewer features.

7 Conclusion

This chapter has presented an ACO-based method for feature selection, with particular emphasis on fuzzy-rough feature selection. This novel approach has been applied to aid classification of web content and to complex systems monitoring, with very promising results. In all experimental studies there has been no attempt to optimize the fuzzifications or the classifiers employed. It can be expected that the results obtained with such optimization would be even better than those already observed.

The techniques presented here focus mainly on the use of ACO for rough and fuzzy-rough feature selection. However, many alternative selection measures exist that are used within incremental hill-climbing search strategies to locate minimal subsets. Such measures could be easily incorporated into the existing ACO-framework. For AntFRFS, it can be expected that it is best suited for the optimization of fuzzy classifiers, as the feature significance measure utilizes the fuzzy sets required by these techniques.

There are many issues to be explored in the area of ACO-based feature selection. The impact of parameter settings should be investigated - how the values of α, β and others influence the search process. Other important factors to be considered include how the pheromone is updated and how it decays. There is also the possibility of using different static heuristic measures to determine the desirability of edges. A further extension would be the use of dynamic heuristic measures which would change over the course of feature selection to provide more search information.

Acknowledgement. The author would like to thank Qiang Shen for his support during the development of the ideas presented in this chapter.

References

1. Blake CL, Merz CJ (1998) UCI Repository of machine learning databases. Irvine, University of California
http://www.ics.uci.edu/~mlearn/
2. Bonabeau E, Dorigo M, Theraulez G (1999) Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press Inc., New York, NY, USA
3. Chouchoulas A, Shen Q (2001) Rough set-aided keyword reduction for text categorisation. Applied Artificial Intelligence, Vol. 15, No. 9, pp. 843–873
4. Chouchoulas A, Halliwell J, Shen Q (2002) On the Implementation of Rough Set Attribute Reduction. Proceedings of the 2002 UK Workshop on Computational Intelligence, pp. 18–23
5. Cohen WW (1995) Fast effective rule induction. In Machine Learning: Proceedings of the 12th International Conference, pp. 115–123
6. Dash M, Liu H (1997) Feature Selection for Classification. Intelligent Data Analysis, Vol. 1, No. 3, pp. 131–156
7. Devijver P, Kittler J (1982) Pattern Recognition: A Statistical Approach. Prentice Hall
8. Drwal G (2000) Rough and fuzzy-rough classification methods implemented in RClass system. In Proceedings of the 2nd International Conference on Rough Sets and Current Trends in Computing (RSCTC 2000), pp 152–159
9. Dubois D, Prade H (1992) Putting rough sets and fuzzy sets together. In [31], pp. 203–232
10. Düntsch I, Gediga G (2000) Rough Set Data Analysis. In: A. Kent & J. G. Williams (Eds.) Encyclopedia of Computer Science and Technology, Vol. 43, No. 28, pp. 281–301
11. Düntsch I, Gediga G (2000) Rough Set Data Analysis: A road to non-invasive knowledge discovery. Bangor: Methodos
12. Ho TB, Kawasaki S, Nguyen NB (2003) Documents clustering using tolerance rough set model and its application to information retrieval. Studies In Fuzziness And Soft Computing, Intelligent Exploration of the Web, pp. 181–196
13. Jensen R, Shen Q (2003) Finding Rough Set Reducts with Ant Colony Optimization. In Proceedings of the 2003 UK Workshop on Computational Intelligence, pp 15–22
14. Jensen R, Shen Q (2004) Fuzzy-rough attribute reduction with application to web categorization. Fuzzy Sets and Systems, Vol. 141, No. 3, pp. 469–485
15. Jensen R, Shen Q (2004) Semantics-Preserving Dimensionality Reduction: Rough and Fuzzy-Rough Based Approaches. IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 12, pp. 1457–1471

16. Jensen R, Shen Q (2005) Fuzzy-Rough Data Reduction with Ant Colony Optimization. Fuzzy Sets and Systems, Vol. 149, No. 1, pp. 5–20
17. Kennedy J, Eberhart RC (1995) Particle swarm optimization. Proceedings of IEEE International Conference on Neura l Networks, pp. 1942–1948
18. Langley P (1994) Selection of relevant features in machine learning. In Proceedings of the AAAI Fall Symposium on Relevance, pp. 1–5
19. Pal SK, Skowron A (eds.) (1999) Rough-Fuzzy Hybridization: A New Trend in Decision Making. Springer Verlag, Singapore
20. Pawlak Z (1982) Rough Sets. International Journal of Computer and Information Sciences, Vol. 11, No. 5, pp. 341–356
21. Pawlak Z (1991) Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishing, Dordrecht.
22. Platt J (1998) Fast Training of Support Vector Machines using Sequential Minimal Optimization. Advances in Kernel Methods - Support Vector Learning, B. Schölkopf, C. Burges, and A. Smola, eds., MIT Press
23. Polkowski L, Lin TY, Tsumoto S (eds.) (2000) Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems, Vol. 56 Studies in Fuzziness and Soft Computing, Physica-Verlag, Heidelberg, Germany
24. Quinlan JR (1993) C4.5: Programs for Machine Learning. The Morgan Kaufmann Series in Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA
25. Raman B, Ioerger TR (2002) Instance-based filter for feature selection. Journal of Machine Learning Research, Vol. 1, pp. 1–23
26. Rasmani K, Shen Q (2004) Modifying weighted fuzzy subsethood-based rule models with fuzzy quantifiers. In Proceedings of the 13th International Conference on Fuzzy Systems, pp. 1687–1694
27. Salton G, Buckley C (1988) Term Weighting Approaches in Automatic Text Retrieval. Information Processing and Management, Vol. 24, No. 5, pp. 513–523
28. Schölkopf B (1997) Support Vector Learning. R. Oldenbourg Verlag, Munich
29. Shen Q, Chouchoulas A (2000) A modular approach to generating fuzzy rules with reduced attributes for the monitoring of complex systems. Engineering Applications of Artificial Intelligence, Vol. 13, No. 3, pp. 263–278
30. Skowron A, Grzymala-Busse JW (1994) From rough set theory to evidence theory. In Advances in the Dempster-Shafer Theory of Evidence, (R. Yager, M. Fedrizzi, and J. Kasprzyk eds.), John Wiley & Sons, Inc.
31. Slowinski R (ed.) (1992) Intelligent Decision Support. Kluwer Academic Publishers, Dordrecht
32. Swiniarski RW (1996) Rough set expert system for online prediction of volleyball game progress for US olympic team. In Proceedings of the 3rd Biennial European Joint Conference on Engineering Systems Design Analysis, pp. 15–20
33. Witten IH, Frank E (1998) Generating Accurate Rule Sets Without Global Optimization. In Machine Learning: Proceedings of the 15th International Conference, Morgan Kaufmann Publishers, San Francisco
34. Witten IH, Frank E (2000) Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco
35. Wygralak M (1989) Rough sets and fuzzy sets - some remarks on interrelations. Fuzzy Sets and Systems, Vol. 29, No. 2, pp. 241–243
36. Yahoo. www.yahoo.com
37. Zadeh LA (1965) Fuzzy sets. Information and Control, 8, pp. 338–353