Improved Rough Set Algorithms for Optimal Attribute Reduct

C. Velayutham and K. Thangavel

Abstract—Feature selection (FS) aims to determine a minimal feature (attribute) subset from a problem domain while retaining a suitably high accuracy in representing the original features. Rough set theory (RST) has been used as such a tool with much success. RST enables the discovery of data dependencies and the reduction of the number of attributes contained in a dataset using the data alone, requiring no additional information. This paper describes the fundamental ideas behind RST-based approaches, reviews related FS methods built on these ideas, and analyses more frequently used RST-based traditional FS algorithms such as Quickreduct algorithm, entropy based reduct algorithm, and relative reduct algorithm. It is found that some of the drawbacks in the existing algorithms and our proposed improved algorithms can overcome these drawbacks. The experimental analyses have been carried out in order to achieve the efficiency of the proposed algorithms.

Index Terms—Data mining, entropy based reduct, Quickreduct, relative reduct, rough set, selection of attributes.

1. Introduction

Data reduction is an important step in knowledge discovery from data. The high dimensionality of databases can be reduced using suitable techniques, depending on the requirements of the data mining processes. These techniques fall into one of the two categories: those that transform the underlying meaning of the data features and those that are semantics-preserving. Feature selection (FS) methods belong to the latter category, where a smaller set of the original features is chosen based on a subset evaluation function. The process aims to determine a minimal feature subset from a problem domain while retaining a suitably high accuracy in representing the original features. In knowledge discovery, FS methods are particularly desirable as these facilitate the interpretability of the resulting knowledge. Rough set theory (RST) has been used as such a tool with much success, enabling the discovery of data dependencies and the reduction of the number of features contained in a dataset using the data alone, requiring no additional information.

The rest of the paper is organized as follows. Section 2 presents introduction to the rough set theory. Section 3 describes the FS algorithms. Section 4 describes the proposed methods for FS. The experimental results are discussed in Section 5 and conclusion is presented in Section 6.

2. Rough Set Theory

RST has been used as a tool to discover data dependencies and to reduce the number of attributes contained in a dataset using the data alone, requiring no additional information. Over the past ten years, RST has become a hot topic and has been applied to many domains. 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.

RST possesses many features in common (to a certain extent) with the Dempster-Shafer theory of evidence, and fuzzy set theory. The rough set itself is the approximation of a vague concept (set) with a pair of precise concepts, called lower and upper approximations. These are the 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. The approximations are constructed with regard to a particular subset of features.

RST works by making use of the granularity structure of the data only. This is a major difference when compared with Dempster-Shafer theory and fuzzy set theory which...
require probability assignments and membership values, respectively. However, this does not mean that no model assumptions are made. In fact, by using the given information, the theory assumes that the data is a true and accurate reflection of the real world (which may not be the case). The numerical and other contextual aspects of the data are ignored, which may seem to be a significant omission, but keeps model assumptions to a minimum.

**Basic Rough Set Concepts**

Let \( I = (U, A \cup \{d\}) \) be an information system, where \( U \) is the universe with a non-empty set of finite objects, \( A \) is a non-empty finite set of condition attributes, and \( d \) is the decision attribute (such a table is also called decision table), \( \forall a \in A \), there is a corresponding function \( f_a : U \rightarrow V_a \), where \( V_a \) is the set of values of \( a \). If \( P \subseteq A \), there is an associated equivalence relation:

\[
\text{IND}(P) = \{(x, y) \in U \times U \mid \forall a \in P, f_a(x) = f_a(y)\}.
\]  

(1)

The partition of \( U \) generated by \( \text{IND}(P) \) is denoted as \( U/P \). If \( (x, y) \in \text{IND}(P) \), then \( x \) and \( y \) are indiscernible by attributes from \( P \). The equivalence classes of the \( P \)-indiscernibility relation are denoted as \( [x]_P \). Let \( X \subseteq U \), the \( P \)-lower approximation \( P_X \) and \( P \)-upper approximation \( \bar{P}X \) of set \( X \) can be defined as:

\[
P_X = \{x \in U \mid [x]_P \subseteq X\}  
\]  

(2)

\[
\bar{P}X = \{x \in U \mid [x]_P \cap X \neq \emptyset\}.
\]  

(3)

Let \( P, Q \subseteq A \) be equivalence relations over \( U \), then the positive, negative, and boundary regions can be defined as:

\[
\text{POS}_P(Q) = \bigcup_{x \in U \cap Q} P_X
\]  

(4)

\[
\text{NEG}_P(Q) = U - \bigcup_{x \in U \cap Q} \bar{P}X
\]  

(5)

\[
\text{BND}_P(Q) = \bigcup_{x \in U \cap Q} \bar{P}X - \bigcup_{x \in U \cap Q} P_X.
\]  

(6)

The positive region of the partition \( U/Q \) with respect to \( P \), \( \text{POS}_P(Q) \), is the set of all objects of \( U \) that can be certainly classified to blocks of the partition \( U/Q \) by means of \( P \). \( Q \) depends on \( P \) in a degree \( k \) (\( 0 \leq k \leq 1 \)) denoted by \( P \Rightarrow_k Q \)

\[
k = \gamma_P(Q) = \frac{\text{POS}_P(Q)}{|U|}.
\]  

(7)

When \( P \) is a set of condition attributes and \( Q \) is the decision, \( \gamma_P(Q) \) is the quality of classification\(^3\). If \( k=1 \), \( Q \) depends totally on \( P \), if \( 0<k<1 \), \( Q \) depends partially on \( P \), and if \( k=0 \) then \( Q \) does not depend on \( P \). The goal of attribute reduct is to remove redundant attributes so that the reduced set provides the same quality of classification as the original. The set of all reducts is defined as:

\[
\text{Red} = \{R \subseteq C \mid \forall k \in \{0,1\}, \gamma_R(D) = \gamma_C(D)\}. 
\]  

(8)

A dataset may have many attribute reducts. The set of all optimal reducts is:

\[
\text{Red}_\text{min} = \{R \in \text{Red} \mid \forall R' \in \text{Red}, |R| \leq |R'|\}.
\]  

(9)

**3. Feature Selection Algorithms**

The main aim of FS is to determine a minimal feature subset from a problem domain while retaining a suitably high accuracy in representing the original features. In many real world problems, FS is a must due to the abundance of noisy, irrelevant or misleading features. For instance, by removing these factors, learning from data techniques can benefit greatly. A detailed review of FS techniques devised for classification tasks can be found in [6].

The usefulness of a feature or feature subset is determined by both its relevancy and redundancy. A feature is said to be relevant if it is predictive for the decision feature(s), otherwise it is irrelevant. A feature is considered to be redundant if it is highly correlated with other features. Hence, the search for a good feature subset involves finding those features that are highly correlated with the decision feature(s), but are not correlated with each other.

**3.1 Quickreduct Algorithm**

The Quickreduct (QR) algorithm given in Algorithm 1 attempts to calculate a reduct without exhaustively generating all possible subsets\(^3\). 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. According to the algorithm, the dependency of each attribute is calculated and the best candidate is chosen.

**Algorithm 1.** The QR algorithm.

\[
\begin{align*}
\text{QR} & : (C, D) \\
& C, \text{ the set of all conditional features; } \\
& D, \text{ the set of decision features.} \\
(1) & \ R \leftarrow \emptyset \\
(2) & \text{do} \\
(3) & \ T \leftarrow R \\
(4) & \forall x \in (C - R) \\
(5) & \text{if } \gamma_{R \cup \{x\}}(D) > \gamma_T(D) \\
(6) & \ T \leftarrow R \cup \{x\} \\
(7) & \ R \leftarrow T \\
(8) & \text{until } \gamma_R(D) = \gamma_C(D) \\
(9) & \text{return } R
\end{align*}
\]
Worked Example

Considering the example dataset given in Table 1, \{a, b, c, d\} are conditional attributes and \{e\} is a decision attribute. The dependency values of each attribute are calculated in Step 1 as discussed below:

Step 1.

\[ \gamma_{\{a\}}(\{e\}) = \frac{|P_{\{a\}}(\{e\})|}{|U|} = \frac{|\{3, 7, 12\}|}{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|} = \frac{3}{14} \]

\[ \gamma_{\{b\}}(\{e\}) = \frac{|P_{\{b\}}(\{e\})|}{|U|} = \frac{|\{1, 2, 3, 5, 6, 7, 9, 11, 13\}|}{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|} = \frac{9}{14} \]

\[ \gamma_{\{c\}}(\{e\}) = \frac{|P_{\{c\}}(\{e\})|}{|U|} = \frac{|\{14\}|}{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|} = \frac{1}{14} \]

\[ \gamma_{\{d\}}(\{e\}) = \frac{|P_{\{d\}}(\{e\})|}{|U|} = \frac{|\{\}\}|}{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|} = \frac{0}{14} \]

The dependency values calculated in Step 1 are tabulated in Table 2.

Attribute \(b\) generates the highest dependency degree, hence attribute \(b\) is chosen and indiscernibility of sets \{a, b\}, \{b, c\} and \{b, d\} is evaluated and the degree of dependency is calculated as given in Step 2.

Step 2:

\[ \gamma_{\{a, b\}}(\{e\}) = \frac{|P_{\{a,b\}}(\{e\})|}{|U|} = \frac{|\{1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|}{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|} = \frac{11}{14} \]

\[ \gamma_{\{b, c\}}(\{e\}) = \frac{|P_{\{b,c\}}(\{e\})|}{|U|} = \frac{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|}{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|} = \frac{14}{14} \]

\[ \gamma_{\{b, d\}}(\{e\}) = \frac{|P_{\{b,d\}}(\{e\})|}{|U|} = \frac{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|}{|\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}|} = \frac{9}{14} \]

The dependency values calculated in Step 2 are tabulated in Table 3.

This process continues until the dependency value is 1. If the dependency value is 1 then the given dataset is consistent, otherwise the dataset is inconsistent. Since dependency value of \{b, c\} is 1, the algorithm terminates after evaluating the subset \{b, c\}. The generated reduct shows the way of reducing the dimensionality of the original dataset by eliminating those conditional attributes that do not appear in the subset.

3.2 Entropy Based Reduce Algorithm

Another technique for rough set FS is entropy-based reduction (EBR), developed from the work carried out in [9]. This approach is based on the entropy heuristic employed by machine learning techniques such as C4.5[10]. A similar approach was adopted in [6] where an entropy measure was used for ranking features \{b\}[11].

Table 3: Dependency values

<table>
<thead>
<tr>
<th>Subset</th>
<th>{a, b}</th>
<th>{b, c}</th>
<th>{b, d}</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\gamma_{{P}}({\Omega}))</td>
<td>0.78571</td>
<td>1</td>
<td>0.64286</td>
</tr>
</tbody>
</table>

Algorithm 2. Entropy-based reduction.

EBR \((C, D)\)

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

(1) \(R \leftarrow \{\}\)

(2) do

(3) \(T \leftarrow R\)

(4) \(\forall x \in (C - R)\)

(5) if \(H(R \cup \{x\}) < H(T)\)

(6) \(T \leftarrow R \cup \{x\}\)

(7) \(R \leftarrow T\)

(8) until \(H(D \mid R) = H(D \mid C)\)

(9) return \(R\)
EBR is concerned with examining a dataset and determining those attributes that provide the most gain in information. The entropy of attribute $A$ (which can take values $a_1, a_2, \ldots, a_n$) with respect to the conclusion $C$ (of possible values $c_1, c_2, \ldots, c_m$) is defined as:

$$H(C | A) = -\sum_{j=1}^{n} p(a_j) \sum_{i=1}^{m} p(c_i | a_j) \log_2 p(c_i | a_j) \tag{10}$$

This can be extended to deal with subsets of attributes instead of individual attributes only. Using this entropy measure, the algorithm used in rough set-based attribute reduction can be modified to that shown in Algorithm 2. This algorithm requires no thresholds in order to realize the function that the search for the best feature subset is stopped when the resulting subset entropy is equal to that of the entire feature set. For consistent data, the final entropy of the subset will be zero. It is interesting to note that any subset with an entropy of 0 will also have a corresponding rough set dependency of 1. Hence, this technique can be used for finding rough set reductions if the data is consistent.

**Worked Example**

Considering the example dataset given in Table 1, EBR first evaluates the entropy of each individual attribute. Table 4 shows the indiscernibility of single conditional attributes and Table 5 shows the indiscernibility of decision attributes.

$$p(a_1) = \frac{[1, 2, 8, 9, 11]}{|U|} = \frac{5}{14}$$

$$p(c_1 | a_1) = \frac{[1, 2, 6, 8, 14] \cap [1, 2, 8, 9, 11]}{|1, 2, 8, 9, 11|} = \frac{3}{5}$$

$$p(c_2 | a_1) = \frac{[3, 4, 5, 7, 9, 10, 11, 12, 13] \cap [1, 2, 8, 9, 11]}{|1, 2, 8, 9, 11|} = \frac{2}{5}$$

$$p(a_2) = \frac{[3, 7, 12]}{|U|} = \frac{3}{14}$$

$$p(c_1 | a_2) = \frac{[1, 2, 6, 8, 14] \cap [3, 7, 12]}{|3, 7, 12|} = \frac{0}{3}$$

$$p(c_2 | a_2) = \frac{[3, 4, 5, 7, 9, 10, 11, 12, 13] \cap [3, 7, 12]}{|3, 7, 12|} = \frac{3}{3}$$

Table 4: Indiscernibility of single conditional attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>{1, 2, 8, 9, 11}</td>
<td>{3, 7, 12}</td>
<td>{4, 5, 6, 10, 13, 14}</td>
</tr>
<tr>
<td>$a_2$</td>
<td>{1, 2, 6}</td>
<td>{4, 8, 10, 12, 14}</td>
<td>{3, 5, 7, 9, 11, 13}</td>
</tr>
<tr>
<td>$a_3$</td>
<td>{1, 2, 3, 4, 10, 12}</td>
<td>{5, 6, 7, 8, 9, 11, 13}</td>
<td>{14}</td>
</tr>
<tr>
<td>$a_4$</td>
<td>{1, 3, 4, 5, 8, 9, 10, 13}</td>
<td>{2, 6, 7, 11, 12, 14}</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Indiscernibility of decision attributes

<table>
<thead>
<tr>
<th>Decision Attribute</th>
<th>$c_1$</th>
<th>$c_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_1$</td>
<td>{1, 2, 6, 8, 14}</td>
<td>{3, 4, 5, 7, 9, 10, 11, 12, 13}</td>
</tr>
</tbody>
</table>

Similarly, we can calculate the entropy values for other attributes, and the values are tabulated in Table 6.

Here, the subset $\{b, c\}$ is chosen as the result in the lowest entropy. Additionally, the stopping criterion has been met as this value equals the entropy for the entire feature set ($H(D) = 0 = H(D(C))$). The algorithm terminates and returns this feature subset, and the dataset can now be reduced to these features only. As the resulting entropy is zero, the returned subset is a rough set reducet.

### 3.3 Relative Reduct Algorithm

In [13], a FS method based on an alternative dependency measure is presented. The technique was originally proposed to avoid the calculation of discernibility functions or positive regions, which can be computationally expensive without optimizations. The authors replace the traditional rough set degree of dependency with an alternative measure\(^{[14]}\). The degree of dependency is defined as follows:

$$K(D) = \frac{|U/IND(R)|}{|U/IND(R \cup D)|}.$$  \(\tag{11}\)

Table 6: Entropy values

<table>
<thead>
<tr>
<th>Subset</th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entropy</td>
<td>0.74032</td>
<td>0.34677</td>
<td>0.82512</td>
<td>0.89216</td>
</tr>
</tbody>
</table>

Table 7: Indiscernibility for the combination of two attributes

<table>
<thead>
<tr>
<th>Subset</th>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
<th>$d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entropy</td>
<td>0.19678</td>
<td>0</td>
<td>0.33963</td>
<td></td>
</tr>
</tbody>
</table>
Algorithm 3. The relative reduct (RR) algorithm.

\[ K_{[b,c]}(D) = \frac{U \setminus \text{IND}(b,c)}{U \setminus \text{IND}(b,c, e)} \]

\[ \left[ \left\{ [1,2],[6],[4,10],[12],[5,9,13],[7,11] \right\} \right] = \frac{6}{8}. \]

As the relative dependency is equal to 1, attribute \( d \) can be removed from the reduct candidate \( R = \{ b, c \} \). Hence, the current reduct candidate is \( R = \{ b, c \} \). As there is no further attribute to consider, the algorithm terminates and outputs the reduct \( \{ b, c \} \).

While analyzing the above FS algorithm we identified some drawbacks in QR algorithm and EBR algorithm, whereas the RR algorithm selects more optimal number of features perfectly, since it performs backward elimination process and sometimes the algorithm selects different attributes which is also more optimal.

3.4 Optimal Number of Features

Considering the example dataset given in Table 9, \( a, b, c, d \) are conditional attributes and \( \{ e \} \) is a decision attribute. The optimal number of features can also be found by evaluating positive region. If the positive region of a subset of attributes contains all the objects, then the cardinality of the subset is an optimal number of features. A detailed review of selecting optimal number of feature can be found in [15]. First we evaluate the positive region for each subset which is also more optimal.

As the relative dependency is not equal to 1, attribute \( c \) is not removed from \( R \). The algorithm then evaluates the elimination of attribute \( b \) from \( R \):

\[ K_{[a,b]}(D) = \frac{U \setminus \text{IND}(a,b)}{U \setminus \text{IND}(a,b,c,d)} \]

\[ \left[ \left\{ [1,2],[6],[4,10],[12],[5,9,13],[7,11] \right\} \right] = \frac{5}{9}. \]

As the relative dependency is not equal to 1, attribute \( b \) is not removed from \( R \). The algorithm then evaluates the elimination of attribute \( c \) from \( R \):

\[ K_{[a,c]}(D) = \frac{U \setminus \text{IND}(a,c)}{U \setminus \text{IND}(a,c,d)} \]

\[ \left[ \left\{ [1,2],[6],[4,10],[12],[5,9,13],[7,11] \right\} \right] = \frac{7}{7}. \]

As the relative dependency is equal to 1, attribute \( d \) can be removed from the reduct candidate \( R = \{ a, b, c \} \). Hence, the current reduct candidate is \( R = \{ a, b, c \} \). As there is no further attribute to consider, the algorithm terminates and outputs the reduct \( \{ a, b, c \} \).

Table 9: Example dataset

<table>
<thead>
<tr>
<th>x ( \in ) U</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
The following illustration shows the drawbacks of the algorithms. Now consider the performance of three FS algorithms for the example dataset given in Table 9.

### A. Drawback of QR Algorithm

According to the QR algorithm, the dependency of each attribute is to be calculated and we have to select the best candidate. Table 12 shows the dependency values of single attribute.

The attribute \( d \) generates the highest dependency degree, so the attribute \( d \) is chosen and the dependency values of the constructed sets \( \{a, d\} \), \( \{b, d\} \), and \( \{c, d\} \) are evaluated. Table 13 shows the dependency values of two attributes.

The attribute set \( \{c, d\} \) generates the highest dependency degree, so the attribute \( c \) is chosen and the set \( \{a, b, c\} \) is constructed and the dependency value is evaluated as shown in Table 15.

The attribute set \( \{a, b, c\} \) generates the dependency degree 1 and it is also the signal for terminating the process. Hence the algorithm selects \( \{a, b, c\} \) but the optimal feature set is \( \{a, b, c\} \). Therefore, this Algorithm is not guaranteed to find optimal subset.

### B. Drawback in Entropy Based Reduct Algorithm

The EBR algorithm, first evaluates the entropy of each individual attribute as shown in Table 16.

The subset with lowest entropy is \( \{d\} \). The algorithm selects attribute \( d \) and adds it to the current feature subset. The next step is to calculate the entropy of all subsets containing \( d \) and another attribute. Table 17 shows entropy values of two attributes.
Here the subset \{c, d\} has the lowest entropy. The algorithm selects attribute \{c, d\} and adds it to the current feature subset. The next step is to calculate the entropy of all subsets containing \{c, d\} and one other attribute. Table 18 shows entropy values of three attributes.

Here the subsets \{a, c, d\} and \{b, c, d\} have the same entropy. Since \{a, c, d\} is evaluated first, the algorithm selects attribute set \{a, c, d\} and adds it to the current feature subset. The next step is to calculate entropy of \{a, b, c, d\}. Table 19 shows entropy values of four attributes.

Since the entropy value of set \{a, b, c, d\} is 0, we reach the terminating point and the algorithm selects all the attributes \{a, b, c, d\}, but the optimal feature set is \{a, b, c\}. Therefore, this algorithm is also not guaranteed to find an optimal subset.

C. RR Algorithm

Based on relative dependency, this algorithm is performed by backward elimination of attribute. First, initialize \(R\) to the set of conditional attributes, \{a, b, c, d\}, next the elimination of attribute \{a\} is considered.

As the relative dependency is not equal to 1, attribute \{a\} is not removed from \(R\). The algorithm then considers the elimination of \{b\} from \(R\) and evaluates the relative dependency. As the relative dependency is also not equal to 1, \{b\} is not removed from \(R\). Table 20 shows relative dependency values.

Next the elimination of \{c\} from \(R\) is considered and the relative dependency is evaluated. As the relative dependency is also not equal to 1, \{c\} is not removed from \(R\), then the attribute \{d\} is finally eliminated and the relative dependency is evaluated. The relative dependency is equal to 1, therefore attribute \{d\} is removed from \(R\). As there is no further attribute to consider, the algorithm terminates and outputs the reduct \{a, b, c\}. This algorithm performs perfectly to find optimal subset for the example data given in Table 9.

It is found that the QR and EBR are not evaluated optimal (minimal) subsets. So the existing QR and EBR algorithms have some drawbacks to evaluate optimal subset. Hence we propose modified QR and modified EBR algorithms, which overcome these drawbacks. The experimental analyses have been carried out in order to achieve the efficiency of the proposed algorithms.

4. Proposed Methods

In above section, the existing rough set based FS algorithms QR, EBR and RR are analysed. From the results given in Table 21, it can be seen that even for small and medium-sized datasets, QR and EBR methods often fail to find minimal subsets that are illustrated in the example dataset given in Table 9. Hence, two modified methods MQR and MEBR are proposed to give better results.

4.1 Modified Quickreduct (MQR) Algorithm

When the cardinality of attribute becomes one and the dependency value is greater than or equal to the threshold value \(t\) then select the attribute while testing the single attribute. This modification is carried out from the existing QR algorithm.

Algorithm 4. The MQR algorithm.

\[
\text{MQR}(C, D)\\ \quad C, \text{ the set of all conditional features};\\ \quad D, \text{ the set of decision features}.\\ (1) R \leftarrow \emptyset\\ (2) \text{do}\\ (3) \quad T \leftarrow R\\ (4) \quad \forall x \in (C - R)\\ (5) \quad \text{if Card}(T) = 1 \land \gamma_T(D) \geq t\\ (6) \quad R \leftarrow T\\ (7) \quad \text{else if } \gamma_{R \cup \{x\}}(D) > \gamma_T(D)\\ (8) \quad T \leftarrow R \cup \{x\}\\ (9) \quad R \leftarrow T\\ (10) \text{until } \gamma_R(D) = \gamma_C(D)\\ (11) \text{return } R
\]

The modified QR algorithm is experimented using the example dataset given in Table 9, and the dependency of each attribute is calculated and tabulated in Table 22 to choose if the degree of dependency is greater than or equal to the threshold value \(t=0.8\). Here, no attribute dependency is greater than or equal to 0.8, so no candidate is selected.
The next step evaluates dependency values for all possible combination two attribute sets \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, and \{c, d\} as tabulated in Table 23.

The attribute set \{a, b\} generates the highest dependency degree, so the attribute \{a, b\} is chosen and the sets \{a, b, c\} and \{a, b, d\} are evaluated and tabulated in Table 24.

The attribute \{a, b, c\} generates the dependency degree 1 and it is also the signal for terminating the process. Hence, the algorithm selects \{a, b, c\}, which is the optimal features subset.

### 4.2 Modified Entropy Based Reduce Algorithm

The modified entropy based reduce (MEBR) algorithm is tested using example dataset given in Table 9, first evaluates the entropy of each individual attribute. When the cardinality of attribute becomes 1 and the entropy is less than or equal to the threshold value \(t\) then select the attribute while testing the single attribute. This modification is carried out from the existing entropy based reduce algorithm.

**Algorithm 5. The MEBR algorithm.**

MEBR(\(C, D\))

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

1. \(R \leftarrow \emptyset\)
2. do
3. \(T \leftarrow R\)
4. \(\forall x \in (C - R)\)
5. if \(\text{Card}(T) = 1 \& H(T) \leq t\) then 
6. \(R \leftarrow T\)
7. else if \(H(R \cup \{x\}) < H(T)\) then 
8. \(T \leftarrow R \cup \{x\}\)
9. \(R \leftarrow T\)
10. until \(H(D | R) = H(D | C)\)
11. return \(R\)

Table 25 shows the entropy values. If any one of the entropy is less than or equal to the threshold value \(t = 0.2\) then choose that candidate, otherwise do not choose any candidate at this stage. Here no attribute dependency is less than or equal to 0.2. So no one is chosen. Next the algorithm calculates the entropy of all possible subsets combination with two attributes as tabulated in Table 26.

### 5. Experimental Results

Table 29 presents the dataset information, as well as the size of the optimal (minimal) reduce. The datasets are taken from the UCI Repository Machine Learning Database[10]. Table 25 shows the entropy values. Further, Table 30 shows the size of the reduce set found by five methods. MQR, EBR, and MEBR produced the same subset every time, unlike QR and RR which often found different subsets and sometimes different subset cardinalities. On the whole, it appears to be the case that our proposed algorithm MQR and MEBR perform well to find optimal reduce.

From the Table 30, it is found that the RR algorithm selects optimal number of attributes but different attributes (WBCD, Lungs Cancer, Protein). The attribute selection of the five methods is shown in Fig. 1.

The expense of the time taken to discover these reduces as can be seen in Table 31. In all the experiments the rough ordering of techniques with respect to time is: EBR<QR<MQR<MEBR<RR, which is shown in Fig. 2.
### Table 30: Subset size evaluate by five methods

<table>
<thead>
<tr>
<th>Index</th>
<th>Dataset</th>
<th>Number of attributes selected by algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>QR</td>
</tr>
<tr>
<td>1</td>
<td>Iris</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>WBCD</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Car</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>ECOLI</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>Lense</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>BUPALiver</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>LungsCancer</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>Protein</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>Diabetes</td>
<td>3</td>
</tr>
</tbody>
</table>

### Table 31: Runtimes for five methods

<table>
<thead>
<tr>
<th>Index</th>
<th>Time in seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>QR</td>
</tr>
<tr>
<td>1</td>
<td>0.8244</td>
</tr>
<tr>
<td>2</td>
<td>2.1330</td>
</tr>
<tr>
<td>3</td>
<td>1.6391</td>
</tr>
<tr>
<td>4</td>
<td>7.5931</td>
</tr>
<tr>
<td>5</td>
<td>0.0280</td>
</tr>
<tr>
<td>6</td>
<td>4.8964</td>
</tr>
<tr>
<td>7</td>
<td>0.7817</td>
</tr>
<tr>
<td>8</td>
<td>47.5599</td>
</tr>
<tr>
<td>9</td>
<td>50.1364</td>
</tr>
</tbody>
</table>

Fig. 1. Attribute selection of the five methods.

Fig. 2. Dataset vs. runtime for five methods.

Fig. 3. Runtime for QR vs. MQR.

Fig. 4. Runtime for EBR vs. MEBR.

Fig. 3 shows the runtime variations for QR vs. MQR algorithms and Fig. 4 shows the runtime variations for EBR vs MEBR algorithms. From these results, it can be seen that even for small and medium-sized datasets, QR and EBR methods often fail to find minimal subsets.

### 6. Conclusions

The existing rough set based FS algorithms QR, EBR, and RR are analysed. From these results, it can be seen that even for small and medium-sized datasets, QR and EBR methods often fail to find minimal subsets. Hence these methods need to be modified to get better results. Though we proposed two modified methods for QR and EBR, the performance of these two proposed methods may well be improved to each individual dataset. In future, the modified approach can be extended to mammogram image datasets for breast cancer diagnosis.

### References


C. Velayutham was born in 1965 at Thanjavur, Tamilnadu, India. He received the Master of Science in applied mathematics in 1989, and Post Graduate Diploma in computer application in 1990, from Bharathidasan University, Trichy, India. He obtained his M.Phil (computer science) degree from Manonmaniam Sundaranar University, Trunelveli, India in 2002. Currently, he is working as an associate professor with the Department of Computer Science, Aditanar College of Arts and Science, Tiruchendur, Tamil Nadu, India. His research interests include medical image processing, data mining, neural network, fuzzy logic, and rough set.

K. Thangavel was born in 1964 at Namakkal, Tamilnadu, India. He received his M.S. degree from the Department of Mathematics, Bharathidasan University in 1986, and Master of Computer Applications Degree from Madurai Kamaraj University, India in 2001. He obtained his Ph.D. degree from the Department of Mathematics, Gandhigram Rural Institute-Deemed University, Gandhigram, India in 1999. Currently, he is working as a professor and the Head with Department of Computer Science, Periyar University, Salem. He is a recipient of Tamilnadu Scientist award for the year 2009. His research interests include medical image processing, artificial intelligence, neural network, fuzzy logic, data mining, and rough set.