CHAPTER - 4

HYBRID ROUGH SET IMPROVED HARMONY SEARCH BASED FEATURE SELECTION FOR PROTEIN CLASSIFICATION

4.1 OVERVIEW

Feature Selection (FS) is a preprocessing technique that plays a major role in the field of bioinformatics. During the last decade, application of feature selection in the proteomic data analysis has become a real prerequisite for model building. In particular, the high dimensional nature of protein sequence representation has given rise to a wealth of feature selection techniques being presented in the field. The classification and prediction of protein sequences are improved by eliminating the irrelevant features without deteriorating the original data. A better quality and more compact subset selection is the maxim of the best feature selection algorithm. The Protein Sequences create an ample challenge in data analysis and interpretation by its huge dimensional data.

Meta-heuristics is a problem-independent technique which intends to find good solution by ‘trial-and-error’ process, which can be applied to wide range of problems. Many real world optimization problems are
addressed by meta-heuristic algorithms such as Evolutionary Programming, Genetic Programming, Evolutionary Strategies and Harmony Search algorithm [44, 45, 99, 135, 76, 20, 97, 95, 153]. These algorithms have surpassed conventional numerical methods on providing better solutions.

Harmony search (HS) [53] is a recently developed meta-heuristic algorithm that mimics the improvisation process of music players. The musicians search for a perfect harmony by adjusting its pitches to achieve better state of composition. The effort to choose the best harmony in music is similar to selecting optimal feature subset in protein dataset. In this research, the extracted features of protein sequences are subjected to novel FS algorithms which hybridize rough set and improved harmony search to deal with high dimensionality present in the protein dataset for better classification.

The rest of this chapter is organized as follows. Section 4.2 reviews various rough set based FS algorithms; Section 4.3 describes about Improved Harmony Search algorithm. A comprehensive study on Rough Set Improved Harmony Search based FS algorithms are discussed in section 4.4 with implementation. Section 4.5 illustrates experimental analysis. The summary of the proposed work is presented in section 4.6.
4.2 ROUGH SET BASED FEATURE SELECTION

Rough set theory provides a mathematical tool that can be used for both dimension reduction and knowledge discovery. The minimal attribute reduct set can be discovered without deterioration of the quality of the classifying objects. The proposed FS algorithms are based on Rough Set Quick Reduct (RSQR) and Rough Set Relative Reduct (RSRR) to increase the effectiveness of feature selection methods by the optimal reduct.

4.2.1 Rough Set Quick Reduct (RSQR)

The RSQR algorithm ensures determination of a reduct without generating all subsets. It starts off with an empty set, and the dependency of each attribute is calculated using rough dependency measure. The attribute with high dependency is added to the set, one at a time, until the maximum value is reached. According to the Quick Reduct (QR) algorithm, the dependency degree of each feature is calculated and the best candidate is chosen [12, 70,154].

4.2.2 Rough Set Relative Reduct (RSRR)

Han et al. [62] proposed a FS method based on a relative dependency measure. Generally, the Relative Reduct (RR) algorithm performs backward elimination of features, in which the process gets initiated by
considering all the features as the reduct set, eliminates one feature and calculates the similarity of each feature vector. Then the feature with highest value is selected. If the degree of dependency is less than 1 which is the dependency of all the conditional attributes on decision attribute, the feature is removed from the set. The RSRR algorithm is proposed in [12, 13].

4.2.3 Meta-heuristics based Feature Selection

Inbarani et al. [69] proposed the feature selection algorithm based on Rough Set and PSO based Quick Reduct and Relative Reduct algorithms. This study introduced hybridized approach to combine the strengths of Rough Set Theory (RST) and Particle Swarm Optimization (PSO) to solve the medical diagnosis problems. The rough set based dependency and relative dependency measure are used as fitness evaluation measures. The experimental results show the highest predictive accuracy than other comparative algorithms for the medical dataset.

Shilaskar et al. [140] proposed a forward inclusion, forward selection and backward elimination based feature subset selection technique for the cardiovascular disease diagnosis. The experiment result of this study

62
proves that this approach finds smaller subsets and increases the accuracy of diagnosis.

Turabieh et al. [152] proposed a hybrid fish swarm optimization algorithm to solve the problem aroused in framing timetable for the examination. It is a population-based algorithm, which imitates the movement of fish searching for food under water. In this study, the fish swarm optimization is hybridized with steepest descent algorithm to enhance its quality in finding a best solution. The proposed algorithm of this study is compared with other benchmark algorithms and the experimental results show that it has produced best results for the test data.

4.3 IMPROVED HARMONY SEARCH ALGORITHM

The Improved Harmony Search (IHS) algorithm was developed by Mahdavi et al. in 2007 [98] to improve the performance of the Harmony Search (HS) algorithm in terms of parameter settings. In this section, the significance of proposed IHS algorithm along with the working principle of HS algorithm is stated in detail.

4.3.1 Harmony Search Algorithm

Geem et al. [50] developed Harmony Search (HS) population based meta-heuristic algorithm that was conceptualized using musical process of
searching for a perfect state of harmony. In music improvisation process, the musician composes many tunes of various instruments and chooses the best tune. This perfect state of harmony in music is similar to finding optimality in optimization process. Similarly, in HS method the best combination of available solution (harmony) is selected and the objective function is optimized [35, 51]. It can explore the search space of a given data in parallel optimization environment, where each harmony vector is generated by intelligently exploring and exploiting a search space. The steps in the Harmony Search procedure are as follows [52].

**Step 1: Initialize the problem and algorithm parameters.**

In this step, the parameters of HS algorithms, Harmony Memory Size (HMS), Harmony Memory Considering Rate (HMCR ∈ [0,1]), Pitch Adjusting Rate (PAR ∈ [0,1]), and number of improvisations (NI) and the problem-based objective function are defined.

**Step 2: Initialize the harmony memory**

The harmony memory (HM) is a matrix of solutions with a size of HMS, where each harmony memory vector represents one solution as defined in (4.1). In this step, the random solutions are built and rearranged
in reverse order to $HM$ based on the objective/fitness function values as $f(x^1) \leq f(x^2) \leq \ldots \leq f(x^{HM})$ [50].

$$HM = \begin{pmatrix} x^1_1 & x^1_2 & \ldots & x^1_n & f(x^1) \\ x^2_1 & x^2_2 & \ldots & x^2_n & f(x^2) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x^{HM}_1 & x^{HM}_2 & \ldots & x^{HM}_n & f(x^{HM}) \end{pmatrix}$$ (4.1)

Each harmony vector in the $HM$ is represented as a binary bit string of length $n$, where $n$ is the total number of conditional attributes, as used in other population based algorithms such as PSO and GA. Therefore, each position in the harmony vector is an attribute subset.

**Step 3: Improvise a new harmony**

A new harmony vector $x' = x'_1, x'_2, \ldots, x'_n$ is generated based on three rules: (1) memory consideration, (2) pitch adjustment and (3) random selection. The process of generating new harmony is called as “improvisation” [103]. In the memory consideration, the values of the new harmony vector are randomly inherited from the historical values stored in $HM$ with a probability of $HMCR$. The $HMCR$, which varies between 0 and 1, is the rate of choosing one value from the historical values stored in the
HM, while \((1 - \text{HMCR})\) is the rate of randomly selecting one value from the possible range of values. This cumulative step ensures that good harmonies are considered as the elements of New Harmony vectors [3].

\[
x' \leftarrow \begin{cases} 
  x' \in \{x_1, x_2, \ldots, x_{\text{HMS}}\} & \text{with probability } \text{HMCR} \\
  x_i \in X_i & \text{with probability } (1 - \text{HMCR}) 
\end{cases} \tag{4.2}
\]

Every component obtained by the memory consideration is examined to determine whether it should be pitch-adjusted [108]. This operation uses the \(\text{PAR}\) parameter, which is the rate of pitch adjustment done as follows:

\[
x_i' \leftarrow \begin{cases} 
  \text{Yes (Adjusting Pitch)} & \text{with probability } \text{PAR} \\
  \text{No (doing nothing)} & \text{with Probability } (1 - \text{PAR}) 
\end{cases} \tag{4.3}
\]

The value of \((1 - \text{PAR})\) sets the rate of doing nothing. If a random number \(\text{rand}() \in [0, 1]\) is generated and its value is within the probability of \(\text{PAR}\) then, the new decision variable \((x'_i)\) is adjusted based on the following equation:

\[x'_i = (x'_i) \pm \text{rand()} \times \text{bw} \tag{4.4}\]

The bandwidth \(\text{bw}\) determines the amount of changes that may have occurred to the components of the new vector. It is an arbitrary distance
value that is used to improve the performance of HS. The \( \text{rand}() \) is a function that generates a random number \( \in [0, 1] \). In this Step, memory consideration, pitch adjustment or random selection are applied to each variable of the New Harmony vector in turn. Consequently, it explores more solutions in the search space and improves the searching abilities [52].

**Step 4: Update the harmony memory**

For each new harmony, the value of objective function \( f(x') \), is calculated. If the new harmony is better than the worst harmony in the \( HM \), the new is included in the \( HM \) and the existing worst harmony is excluded from the \( HM \).

**Step 5: Check the stopping criterion**

If the stopping criterion or the maximum number of improvisations is satisfied, computation is terminated. Otherwise, Steps 3 and 4 are repeated. Finally the best harmony memory vector is selected and is considered as the best solution to the problem.

**4.3.2 Advantages of HS Algorithm**

Harmony search algorithm had been very successful in a wide variety of optimization problems, presenting several advantages with
respect to traditional optimization techniques [3, 44, 45, 50]. HS is a novel stochastic random search based algorithm, rather than gradient in numerical calculus.

i) HS not sensitive to the initial value setting for the variables, as it imposes less mathematical complexity.

ii) In HS algorithm, a new vector is generated based on all the existing vectors in the memory, whereas the genetic algorithm (GA) considers only two parent vectors.

This flexibility in HS algorithm produces better results when compared to other conventional approaches.

4.3.3 Problems faced in HS Algorithm

Harmony Search is good at identifying the high performance regions of the solution space at a reasonable time, but gets into trouble in performing local search for numerical applications. In order to improve the fine-tuning characteristic of HS algorithm, IHS employs a new method that enhances fine-tuning characteristic and convergence rate of harmony search. The IHS algorithm has the power of the HS algorithm with the fine tuning feature of mathematical techniques and can outperform either one
To show the great power of this method, IHS algorithm is applied to various standard engineering optimization problems. Numerical results reveal that the proposed algorithm is a powerful search algorithm for various optimization problems [44, 50, 52].

4.3.4 Improvisations to HS Algorithm

The harmony search algorithm is one of the optimization algorithms and at the same time, it is one the most efficient algorithm in the field of combinatorial optimization [52]. It has attracted many researchers in various fields and guided them to improve its performance according to the field of their research and requirements. The two concepts of the HS algorithm are the main area where the improvements can be done, i.e. Parameters setting and hybridization with other algorithms [3].

The proper selection of HS parameter values are considered as one of the challenging tasks not only for HS algorithm but also for other metaheuristic algorithms. This difficulty is due to different reasons, and the most important one is the absence of general rules governing this aspect. Actually, setting these values is problem dependant and therefore the experimental trials are the only guide to the best values. However, this matter guides the research into new variants of HS. These variants are...
based on adding some extra components or concepts to make part of these parameters dynamically adapted.

In HS algorithm, the three parameters HMCR, PAR, and bw (bandwidth) were set initially in the first step. Among the three parameters \( PAR \) and \( bw \) are very important parameters in fine-tuning of optimized solution vectors. However in the traditional HS algorithm, the \( PAR \) and \( bw \) were initialized with a static value, which cannot be changed during new generations [2]. The main drawback of this method is that the number of iterations increases to find an optimal solution. To improve the performance of the HS algorithm and to eliminate the drawbacks with fixed values of \( PAR \) and \( bw \), IHS algorithm uses updated \( PAR \) and \( bw \) values in improvisation step (Step 3). In this research, a very well known improvement is adopted to HS algorithm as stated in Mahdavi et al. [98]. The parameters \( PAR \) and \( bw \) are changed dynamically with generation number and expressed as follows:

\[
PAR(gn) = PAR_{\text{min}} + \frac{PAR_{\text{max}} - PAR_{\text{min}}}{NI} \ast gn \tag{4.5}
\]

Where, \( PAR(gn) \) = Pitch Adjusting Rate for each generation; \( PAR_{\text{min}} \) = Minimum Pitch Adjusting Rate; \( PAR_{\text{max}} \) = Maximum Pitch
Adjusting Rate; \( NI = \) Number of Improvisations and \( gn = \) Generation Number

\[
 bw(gn) = bw_{\text{max}} \cdot \exp(c \cdot gn); \quad (4.6)
\]

\[
 c = \ln \left( \frac{bw_{\text{min}}}{bw_{\text{max}}} \right) / NI \quad (4.7)
\]

Where, \( bw(gn) = \) Bandwidth for each generation; \( bw_{\text{min}} = \) Minimum bandwidth and \( bw_{\text{max}} = \) Maximum bandwidth.

The IHS algorithm explores the search space more widely and efficiently and parameter refinements generate good solutions.

**4.4 PROPOSED RSIHS BASED FEATURE SELECTION**

RST is the best tool for the protein sequence data analysis as it is capable of dealing with vague, uncertain and imperfect data. It has been used to discover data dependencies and to reduce the number of attributes contained in a dataset using the data alone, requiring no additional information. As the protein dataset is huge in attribute size, a population based IHS algorithm is hybridized with RST for the optimal balance between exploration and exploitation in the search space. Besides, in IHS algorithm, multiple harmonies can be used in parallel and the good
parallelism leads to better performance. Hence the IHS algorithm is very flexible to combine with other algorithms [169].

4.4.1 Parameter Setting

The most responsible parameters of RSIHS based algorithms to yield the optimal result are $HMCR$, $PAR$ and bandwidth ($bw$). The Harmony memory takes a vital role, since it ensures the optimal harmonies to be considered as new solution vectors. The accepting rate of the $HM$ to be fixed precisely by setting the parameter $HMCR \in [0, 1]$ as it is neither too low nor too high, because there is a possibility to converge too slowly with few memories and too many memories will not explore well for the optimal solution respectively. For example, the $HMCR$ is 0.75 which indicates that the IHS algorithm accepts $HM$ with 75% probability. The adjustment of $PAR$ parameter is a refinement process of local optimum which gives better solution. In this study, the initial parameters for the proposed FS algorithms are set as shown in Table 4.1. The $HMCR$ parameter is set to 0.9 based on the analysis depicted in Table 4.2 and Table 4.3 which is discussed in the later section.
Table 4.1 Initial Parameter Setting for Proposed FS algorithms

<table>
<thead>
<tr>
<th>FS Algorithm</th>
<th>HM</th>
<th>HMC</th>
<th>Minimum PAR ($PAR_{min}$)</th>
<th>Maximum PAR ($PAR_{max}$)</th>
<th>Minimum bandwidth ($bw_{min}$)</th>
<th>Minimum bandwidth ($bw_{max}$)</th>
<th>Number of Iterations (NI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSIHSQR</td>
<td>30</td>
<td>0.9</td>
<td>0.5</td>
<td>0.8</td>
<td>0.0001</td>
<td>1.0</td>
<td>50</td>
</tr>
<tr>
<td>RSIHSRR</td>
<td>30</td>
<td>0.9</td>
<td>0.5</td>
<td>0.8</td>
<td>0.0001</td>
<td>1.0</td>
<td>50</td>
</tr>
</tbody>
</table>

4.4.2 RSIHSQR Feature Selection Algorithm

In the proposed RSIHSQR algorithm, a reduct set is computed without generating all possible subsets. It starts with an empty set and it adds one at a time, in turn. The Harmony Memory is constructed and the parameters $HMCR$, $PAR_{min}$, $PAR_{max}$, $bw_{min}$, $bw_{max}$ and $NI$ are initialized as depicted in Table 4.1. The fitness function of each harmony vector is evaluated using rough dependency measure which is defined in (4.8).

$$\gamma_{RU(x)}(D) = \frac{|POSR(u)(D)|}{|U|}$$  \hspace{1cm} (4.8)

A harmony with highest fitness value among all the possible solutions is considered and rearranged in $HM$ as local best solution. Then the new harmony vector is constructed. Based on the probability of $HMCR$, the new harmony is constructed which is randomly inherited from the historical
values stored in $HM$. Each vector is constructed based on $HMCR$ probability which must be examined for pitch adjustment. Then the $PAR$ and $bw$ parameters are updated as stated in equations (4.5 – 4.7). The updated parameters are applied to each variable of the new harmony in turn. The harmony memory is updated based on the fitness value calculated for each new vector in $HM$. If the new harmony is better than the worst harmony in the $HM$, then the new harmony is included in the resulting $HM$ and the existing worst harmony is excluded. This process is repeated until the stopping criterion is met. The returned harmony vector is selected and is considered as the best solution for the RSIHQSQR algorithm, as depicted in figure 4.1.

**Algorithm: RSIHQSQR(C,D)**

**Input**: $C$, the set of conditional attributes; $D$, the set of decision attributes  
**Output**: Best Reduct (feature)

**Step 1**: Define the fitness function, $f(X)$  
Initialize the variables $HMS=10$  
$HMCR = 0.95$  
$NI = 100$  
$PVB$  
$PAR_{\text{min}}, PAR_{\text{max}}, bw_{\text{min}}, bw_{\text{max}}$,  
Pitch Adjusting Rate & bandwidth $\epsilon$(0 to 1)

$fit = 0; X_{old} = X_1; bestfit = X_1; bestreduct = \{}$

**Step 2**: Initialize Harmony Memory, $HM = (X_1, X_2, ..., X_{HMS})$  
For $i = 1$ to $HMS$  
∀: $X_i$ // $X_i$ is the $i^{th}$ harmony vector of $HM$

//Compute fitness of feature subset of $X_i$

$R \leftarrow$ Feature subset of $X_i$ (1’s of $X_i$)
\( \forall x \in (C - R) \)
\[ \gamma_{R \cup \{x\}}(D) = \frac{|\text{POS}_{R \cup \{x\}}(D)|}{|\emptyset|} \]
\( f(X_i) = \gamma_{R \cup \{x\}}(D) \quad \forall X \subseteq R, \gamma_X(D) \neq \gamma_C(D) \)

If \( f(X_i) > \text{fit} \)
\[ \text{fit} \leftarrow f(X_i) \]
\[ X_{\text{old}} \leftarrow X_i \]
End if

**Step 3: Improvise new Harmony Memory**

While \( \text{iter} \leq Nl \) or \( \text{fit} = 1 \) // Stopping Criterion

for \( j = 1, 2, \ldots, \text{NVAR} \)

\( \forall : \text{X}_{\text{old}}(j) \) // \( x \) is the variable of \( X \)

Update Pitch Adjusting Rate();

Update bandwidth();

if \( \text{rand}() \leq \text{HMCR} \) // \( \text{rand} \in [0, 1] \)

// construct the new harmony \( X_{\text{new}} \) from the best harmony vector \( X_{\text{old}} \)

\( X_{\text{new}} \leftarrow X_{\text{old}}; \) // assigning the best harmony to the new harmony

if \( \text{rand}() \leq \text{PAR} \) // \( \text{rand} \in [0, 1] \)

\( X_{\text{new}}(j) = X_{\text{new}}(j) \pm \text{rand}() \times \text{bw} \)

end if

else

// choose a random value of variable \( X_{\text{new}} \)

\( X_{\text{new}}(j) = \text{PVBlower} + \text{rand}() \times (\text{PVBupper} - \text{PVBlower}) \)

end if

end for

**Step 4: Update the new Harmony Memory**

Compute fitness function for New Harmony \( X_{\text{new}} \) as defined in Step 2.

if \( f(X_{\text{new}}) \geq f(X_{\text{old}}) \)

// Accept and replace the old harmony vector with new harmony.

\( X_{\text{old}} \leftarrow X_{\text{new}}; \)

if \( f(X_{\text{new}}) > \text{fit} \)

\[ \text{fit} \leftarrow f(X_{\text{new}}); \]

\[ \text{bestfit} \leftarrow X_{\text{new}}; \]

End if

Exit

end if

// continue with the next iteration

end while

\( \text{bestreduct} \leftarrow \text{feature subset of bestfit} \) // Reduced feature subset: 1's of bestfit

Figure 4.1 Proposed RSIHSQR Feature Selection algorithm
4.4.3 RSIHSRR Feature Selection Algorithm

The RSIHSRR feature selection algorithm performs the same steps of rough set improved harmony search, except that it employs relative reduct dependency metric for the fitness function [14, 15]. The functioning of relative reduct algorithm is described earlier in this chapter.

\[
f(x) = K_{R-\{x\}}(D) = \frac{|U/\text{IND}(R)|}{|U/\text{IND}(RUD)|} \quad (4.9)
\]

In the RSIHSRR algorithm, each harmony vector is evaluated using the fitness function as defined in (4.9). The detailed working principle of the proposed RSIHSRR algorithm is represented in figure 4.2.

<table>
<thead>
<tr>
<th>Algorithm: RSIHSRR(C,D)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong>: C, the set of conditional attributes; D, the set of decision attributes</td>
</tr>
<tr>
<td><strong>Output</strong>: Best Reduct (feature)</td>
</tr>
</tbody>
</table>

**Step 1**: Define the fitness function, \( f(X) \)
- Initialize the variables HMS = 10 // Harmony Memory Size (population)
- HMCR = 0.95 // Harmony Memory Consideration Rate (for improvisation)
- NI = 100 // Maximum number of Iterations,
- PVB // Possible value bound of X
- \( PAR_{\text{min}}, PAR_{\text{max}}, bw_{\text{min}}, bw_{\text{max}} \) // Pitch Adjusting Rate & bandwidth \( \epsilon \)(0 to 1)
- fit = 0; \( X_{\text{old}} = X_1 \); bestfit = \( X_1 \);
- bestreduct = \{\};

**Step 2**: Initialize Harmony Memory, \( HM = (X_1, X_2, \ldots, X_{\text{HMS}}) \)
- For \( i = 1 \) to HMS // for each harmony
- \( \forall X_i \) // \( X_i \) is the \( i^{th} \) harmony vector of HM
- // Compute fitness of feature subset of \( X_i \)
- \( R \leftarrow \) Feature subset of \( X_i \) (1's of \( X_i \))
- \( \forall x \in R \)
- \( K_{R-\{x\}}(D) = \frac{|U/\text{IND}(R)|}{|U/\text{IND}(RUD)|} \)
- \( f(X_i) = K_{R-\{x\}}(D) \quad \forall X \subseteq R, K_X(D) \neq K_C(D) \)
- if \( f(X_i) > \text{fit} \)
fit ← f(X_i)
X_{old} ← X_i
End if

if fit == 1
bestfit=fit; Return R;
Endif

Step 3: Improvise new Harmony Memory
While iter ≤ NI or fit != 1 // Stopping Criterion
for j=1,2,...,NVAR
∀ :X_{old} (j)  // x is the variable of X
Update Pitch Adjusting Rate();
Update bandwidth();
if rand () ≤ HMCR       // rand ∈ [0,1]
// construct the new harmony X_{new} from the best harmony vector X_{old}.
X_{new} ← X_{old};  // assigning the best harmony to the new harmony
if rand () ≤ PAR       // rand ∈ [0,1]
X_{new}(j) = X_{new}(j) ± rand() * bw
end if
else  //choose a random value of variable X_{new}
X_{new}(j)=PV_{Blower} + rand( ) * (PV_{Bupper} − PV_{Blower})
end if
end for

Step 4: Update the new Harmony Memory
Compute fitness function for New Harmony  X_{new} as defined in Step 2.
if f(X_{new}) == 1
bestfit=X_{new}; Return R;
End if
if f(X_{new}) ≥ f(X_{old})
// Accept and replace the old harmony vector with new harmony.
X_{old} ← X_{new};
if f(X_{new}) > fit
fit ← f(X_{new});
bestfit ← X_{new}; End if
Exit
end if
//continue with the next iteration
end while
bestreduct ← feature subset of bestfit
// Reduced feature subset: 1’s of bestfit

Figure 4.2 Proposed RSIHSRR Feature Selection algorithm
4.5 EXPERIMENTAL ANALYSIS

In this section, the consistent performance of proposed feature selection algorithms RSIHSQR and RSIHSRR are evaluated with three sets of experiments, which are based on parameter analysis, feature reduction and optimality in terms of precision, sensitivity and error-rate measures.

4.5.1 Datasets

Experiments are performed on the two datasets with 2000 protein objects in each decision table. The 1-mer dataset consists of $20 + 1$ features (20 conditional attributes and 1 class attribute). The 2-mer dataset consists of 420 conditional attributes and one class attributes (421 features). The decision table construction from the protein sequences are already described in the earlier chapter. In this work, a Max-Min discretization [125, 136] method is applied for protein dataset.

4.5.2 Results and Discussion

The results of the proposed algorithms RSIHSQR and RSIHSRR are analyzed with three sets of experiments. The obtained results of the proposed FS algorithms are compared with the existing FS and classification algorithms.
4.5.2.1 Experiment I

The performance of the proposed algorithms RSIHSQR and RSIHSRR algorithms are executed, by varying the values of $HMCR$ parameter. The results of two datasets, with three different values (0.7, 0.8, 0.9), are evaluated in terms of feature reduct and predictive accuracy that are shown in Table 4.2 and 4.3. The comparative charts of the $HMCR$ parameter with proposed algorithms on two datasets are exposed in Figure 4.3, Figure 4.4, Figure 4.5, Figure 4.6, Figure 4.7 and Figure 4.8.

<table>
<thead>
<tr>
<th>Protein Dataset</th>
<th>Proposed FS Algorithm</th>
<th>HMCR Value</th>
<th>Number of Features Selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-mer</td>
<td>RSIHSQR</td>
<td>0.7</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>0.9</strong></td>
<td><strong>12</strong></td>
</tr>
<tr>
<td></td>
<td>RSIHSRR</td>
<td>0.7</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>0.9</strong></td>
<td><strong>8</strong></td>
</tr>
<tr>
<td>2-mer</td>
<td>RSIHSQR</td>
<td>0.7</td>
<td>104</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>83</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>0.9</strong></td>
<td><strong>32</strong></td>
</tr>
<tr>
<td></td>
<td>RSIHSRR</td>
<td>0.7</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>0.9</strong></td>
<td><strong>26</strong></td>
</tr>
</tbody>
</table>
### Table 4.3 Analysis of HMCR in terms of Classification Accuracy

<table>
<thead>
<tr>
<th>Protein Dataset</th>
<th>Proposed IHS based FS Algorithm</th>
<th>HMC R Value</th>
<th>Average Precision (%) of Classification Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>0.7</td>
<td>K-NN</td>
</tr>
<tr>
<td>1-mer</td>
<td>RSIHSQR</td>
<td>0.7</td>
<td>64.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>82.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>0.9</strong></td>
<td><strong>84.3</strong></td>
</tr>
<tr>
<td></td>
<td>RSIHSRR</td>
<td>0.7</td>
<td>78.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>72.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>0.9</strong></td>
<td>82.0</td>
</tr>
<tr>
<td>2-mer</td>
<td>RSIHSQR</td>
<td>0.7</td>
<td>76.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>79.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>0.9</strong></td>
<td><strong>90.3</strong></td>
</tr>
<tr>
<td></td>
<td>RSIHSRR</td>
<td>0.7</td>
<td>72.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.8</td>
<td>88.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>0.9</strong></td>
<td><strong>92.5</strong></td>
</tr>
</tbody>
</table>

**Figure 4.3** Analysis of HMCR on 1-mer dataset
Figure 4.4 Analysis of HMCR on 2-mer dataset

Figure 4.5 Analysis of HMCR with RSIHSQR based on classification algorithms on 1-mer dataset
Figure 4.6 Analysis of HMCR with RSIHSQR based on classification algorithms on 2-mer dataset

Figure 4.7 Analysis of HMCR with RSIHSRR based on classification algorithms on 1-mer dataset
4.5.2.2 Experiment II

The Experiment II is conducted to evaluate the performance of the proposed RSIHSQR and RSIHSRR algorithms in terms of the size of the feature subset. The attribute size is the major issue with many predictive algorithms. The results of the proposed FS algorithms are compared with the existing RSQR, RSRR, RSPSOQR and RPSORR algorithms. The results obtained by the feature selection algorithms are shown in Table 4.4. The performance analysis of the proposed algorithms over the existing algorithms on 1-mer and 2-mer datasets are exposed in the figures 4.9 and 4.10 respectively.
### Table 4.4 Results of Feature Selection algorithms

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Number of features selected using feature selection algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RSQR</td>
</tr>
<tr>
<td>1-mer</td>
<td>14</td>
</tr>
<tr>
<td>2-mer</td>
<td>280</td>
</tr>
</tbody>
</table>

#### Figure 4.9 Comparative analysis of RSIHSQR and RSIHSRR algorithms on 1-mer dataset with other FS algorithms
Table 4.4 shows the number of features selected by the proposed and compared feature selection algorithms. It is observed that RSPSORR and RSIHSRR selected least number of features on the 1-mer dataset, whereas on the 2-mer dataset RSIHSQR and RSIHSRR recorded small subset. Figure 4.9 and 4.10 shows the comparative analysis of feature selection algorithms, in which the RSIHSRR feature selection algorithm recorded small feature subset on both 1-mer and 2-mer datasets.
Table 4.5 Precision of Classification algorithms for 1-mer dataset

<table>
<thead>
<tr>
<th>Classification Algorithms</th>
<th>Feature Selection Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unreduced Set</td>
</tr>
<tr>
<td>K-NN</td>
<td>0.643</td>
</tr>
<tr>
<td>MLP</td>
<td>0.679</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.609</td>
</tr>
<tr>
<td>SVM</td>
<td>0.623</td>
</tr>
<tr>
<td>C4.5</td>
<td>0.664</td>
</tr>
</tbody>
</table>

Table 4.6 Recall of Classification algorithms for 1-mer dataset

<table>
<thead>
<tr>
<th>Classification Algorithms</th>
<th>Feature Selection Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unreduced Set</td>
</tr>
<tr>
<td>K-NN</td>
<td>0.652</td>
</tr>
<tr>
<td>MLP</td>
<td>0.698</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.612</td>
</tr>
<tr>
<td>SVM</td>
<td>0.632</td>
</tr>
<tr>
<td>C4.5</td>
<td>0.649</td>
</tr>
</tbody>
</table>
4.5.2.3 Experiment III

The Experiment III is conducted to evaluate the consistency of the proposed and existing feature selection algorithms by five benchmark classifiers which are frequently implemented in various studies of protein analysis, and are already described in section 2.4. The reliability of the feature selection algorithms for two datasets are assessed by precision, recall and error rate measures [82].

Table 4.7 Error Rate of Classification algorithms for 1-mer dataset

<table>
<thead>
<tr>
<th>Classification Algorithms</th>
<th>Feature Selection Algorithms</th>
<th>Unreduced Set</th>
<th>RSQR</th>
<th>RSRR</th>
<th>RSPSOQR</th>
<th>RSPSORR</th>
<th>RSIHSQR</th>
<th>RSIHSRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-NN</td>
<td></td>
<td>0.265</td>
<td>0.261</td>
<td>0.211</td>
<td>0.232</td>
<td>0.224</td>
<td>0.103</td>
<td>0.119</td>
</tr>
<tr>
<td>MLP</td>
<td></td>
<td>0.271</td>
<td>0.236</td>
<td>0.193</td>
<td>0.356</td>
<td>0.148</td>
<td>0.126</td>
<td>0.118</td>
</tr>
<tr>
<td>Random Forest</td>
<td></td>
<td>0.353</td>
<td>0.259</td>
<td>0.186</td>
<td>0.163</td>
<td>0.157</td>
<td>0.086</td>
<td>0.098</td>
</tr>
<tr>
<td>SVM</td>
<td></td>
<td>0.311</td>
<td>0.273</td>
<td>0.171</td>
<td>0.206</td>
<td>0.141</td>
<td>0.121</td>
<td>0.12</td>
</tr>
<tr>
<td>C4.5</td>
<td></td>
<td>0.295</td>
<td>0.269</td>
<td>0.188</td>
<td>0.169</td>
<td>0.131</td>
<td>0.118</td>
<td>0.092</td>
</tr>
</tbody>
</table>
Table 4.8 Precision of Classification algorithms for 2-mer dataset

<table>
<thead>
<tr>
<th>Classification Algorithms</th>
<th>Feature Selection Algorithms</th>
<th>Unreduced Set</th>
<th>RSQR</th>
<th>RSRR</th>
<th>RSPSOQR</th>
<th>RSPSORR</th>
<th>RSIHSQR</th>
<th>RSIHSRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-NN</td>
<td></td>
<td>0.535</td>
<td>0.513</td>
<td>0.802</td>
<td>0.867</td>
<td>0.888</td>
<td>0.891</td>
<td>0.922</td>
</tr>
<tr>
<td>MLP</td>
<td></td>
<td>0.565</td>
<td>0.505</td>
<td>0.789</td>
<td>0.867</td>
<td>0.896</td>
<td>0.910</td>
<td>0.914</td>
</tr>
<tr>
<td>Random Forest</td>
<td></td>
<td>0.767</td>
<td>0.704</td>
<td>0.858</td>
<td>0.901</td>
<td>0.909</td>
<td>0.921</td>
<td>0.934</td>
</tr>
<tr>
<td>SVM</td>
<td></td>
<td>0.789</td>
<td>0.696</td>
<td>0.862</td>
<td>0.886</td>
<td>0.905</td>
<td>0.872</td>
<td>0.934</td>
</tr>
<tr>
<td>C4.5</td>
<td></td>
<td>0.762</td>
<td>0.787</td>
<td>0.805</td>
<td>0.893</td>
<td>0.906</td>
<td>0.887</td>
<td>0.935</td>
</tr>
</tbody>
</table>

Table 4.9 Recall of Classification algorithms for 2-mer dataset

<table>
<thead>
<tr>
<th>Classification Algorithms</th>
<th>Feature Selection Algorithms</th>
<th>Unreduced Set</th>
<th>RSQR</th>
<th>RSRR</th>
<th>RSPSOQR</th>
<th>RSPSORR</th>
<th>RSIHSQR</th>
<th>RSIHSRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-NN</td>
<td></td>
<td>0.539</td>
<td>0.516</td>
<td>0.806</td>
<td>0.866</td>
<td>0.89</td>
<td>0.893</td>
<td>0.925</td>
</tr>
<tr>
<td>MLP</td>
<td></td>
<td>0.568</td>
<td>0.509</td>
<td>0.791</td>
<td>0.867</td>
<td>0.895</td>
<td>0.911</td>
<td>0.915</td>
</tr>
<tr>
<td>Random Forest</td>
<td></td>
<td>0.769</td>
<td>0.712</td>
<td>0.862</td>
<td>0.906</td>
<td>0.911</td>
<td>0.919</td>
<td>0.935</td>
</tr>
<tr>
<td>SVM</td>
<td></td>
<td>0.791</td>
<td>0.701</td>
<td>0.866</td>
<td>0.889</td>
<td>0.908</td>
<td>0.868</td>
<td>0.934</td>
</tr>
<tr>
<td>C4.5</td>
<td></td>
<td>0.766</td>
<td>0.791</td>
<td>0.811</td>
<td>0.895</td>
<td>0.909</td>
<td>0.889</td>
<td>0.936</td>
</tr>
</tbody>
</table>
The results of feature selection algorithms for 1-mer dataset are shown in Table 4.5, Table 4.6 and Table 4.7. The classification results of the feature subset of 2-mer dataset are displayed in Table 4.8, Table 4.9 and Table 4.10. The comparative analyses of feature selection and classification algorithms on 1-mer and 2-mer datasets are depicted in Figures 4.11 – 4.16.

**Table 4.10 Error Rate of Classification algorithms for 2-mer dataset**

<table>
<thead>
<tr>
<th>Classification Algorithms</th>
<th>Feature Selection Algorithms</th>
<th>Unreduced Set</th>
<th>RSQR</th>
<th>RSRR</th>
<th>RSPSOQR</th>
<th>RSPSORR</th>
<th>RSIHSQR</th>
<th>RSIHSRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-NN</td>
<td></td>
<td>0.441</td>
<td>0.423</td>
<td>0.121</td>
<td>0.112</td>
<td>0.051</td>
<td>0.054</td>
<td>0.038</td>
</tr>
<tr>
<td>MLP</td>
<td></td>
<td>0.402</td>
<td>0.445</td>
<td>0.141</td>
<td>0.109</td>
<td>0.053</td>
<td>0.044</td>
<td>0.042</td>
</tr>
<tr>
<td>Random Forest</td>
<td></td>
<td>0.164</td>
<td>0.207</td>
<td>0.092</td>
<td>0.051</td>
<td>0.045</td>
<td>0.041</td>
<td>0.032</td>
</tr>
<tr>
<td>SVM</td>
<td></td>
<td>0.143</td>
<td>0.218</td>
<td>0.102</td>
<td>0.056</td>
<td>0.048</td>
<td>0.064</td>
<td>0.033</td>
</tr>
<tr>
<td>C4.5</td>
<td></td>
<td>0.165</td>
<td>0.146</td>
<td>0.119</td>
<td>0.052</td>
<td>0.051</td>
<td>0.054</td>
<td>0.032</td>
</tr>
</tbody>
</table>

The feature subsets are investigated in terms of precision, recall and error rate validity measures. The feature selection algorithms are evaluated by five benchmark classifiers K-Nearest Neighbor (K-NN), Multi Layer Perceptron (MLP), Random Forest (RF), Support Vector Machine (SVM) and C4.5 using 10-fold cross validation approach. For the analysis purpose,
initially the classification is made on unreduced dataset of proteins, followed by the feature subsets selected by feature selection algorithms.

From the tables and figures, it is clearly observed that the proposed RSIHSQR and RSIHSRR feature selection algorithms roughly recorded better precision and recall with all the classifiers compared to other FS algorithms. The error rate is gradually decreased with the proposed feature selection algorithms on both the datasets. It is also noted that the 2-mer dataset has revealed better classification result with least error rate than 1-mer dataset. Thus it has been proved that the proposed RSIHSRR algorithm outperforms the existing FS algorithms and revealed high precision on 2-mer dataset.
Figure 4.12 Performance Analysis of FS algorithms based on Recall/Specificity of Classifiers on 1-mer dataset

Figure 4.13 Performance Analysis of FS algorithms based on Error Rate of Classifiers on 1-mer dataset
Figure 4.14 Performance Analysis of FS algorithms based on Precision of Classifiers on 2-mer dataset

Figure 4.15 Performance Analysis of FS algorithms based on Recall of Classifiers on 2-mer dataset
4.6 SUMMARY

The key challenge faced in the proteomic research area is the large dimension with imperfect data. In this chapter, two hybridized feature selection algorithms RSIHSQR and RSIHSRR are proposed and compared with existing algorithms. The experiments are performed on two datasets extracted from various domains of proteins. The features selected by the feature selection algorithms are evaluated with benchmark classification algorithms. Based on the experimental results, it is observed that the
proposed algorithm RSIHSRR revealed an optimal reduct, ever since it improves the predictive accuracy of the feature subsets with decreased error rate. Hence the analysis section clearly demonstrates that the proposed algorithms can be applied for the protein dataset for the better classification.