Chapter 1

Introduction

1.1 Feature Selection

Feature selection is an indispensable task in pattern recognition [12]. In big data analytics, it is a common pre-processing step. Since all the features may not be necessary, we need to reduce the features in the original dataset while preserving the predictive capability resulting in reduced dimensionality. In practical datasets where a large number of features are involved, generally following two inconsistencies may be present. The dataset may contain irrelevant (noisy) features and/or redundant features [12]. In these situations, feature selection identifies and removes these irrelevant (noisy) and redundant features while maintaining acceptable classification accuracy. Feature reduction techniques are expected to select a set of feature, known as reduct, having relevant information from a dataset to perform desired tasks [13–16].

For a given dataset, the mapping between the reduced set of features (reducts) and decision features should be the same as the mapping existing between unreduced features and the decision features. Thus, feature selection reduces the dimension and yields reduced computational time and space complexity. The application of feature selection is commonplace in almost every area of computer science applications, viz. text mining [17], image processing and computer vision [18, 19], bioinformatics [20, 21] and industrial applications [22].

All the feature selection techniques can be put in two broad categories viz. feature ranking techniques and feature subset selection techniques. The techniques, which select the top ranked features out of features ranked using some measure, are known as feature

ranking techniques. These methods are normally used as a preliminary step for further processing step of feature selection. The techniques which search for optimal feature subsets by generating and evaluating different feature subsets are known as feature subset selection techniques. The feature subset selection techniques explore the correlation among features, whereas feature ranking techniques don't do so. There are three categories of feature subset selection techniques viz. (i) filter methods, (ii) wrapper methods and (iii) embedded method. These three categories of techniques are discussed as follows.

The filter based techniques assess the relevance of any feature using intrinsic properties of the data and remove low scoring features. The wrapper based methods embed the model hypothesis search within the reduct search process and feature subset is evaluated for a specified learning algorithm. The embedded based techniques are embedded into a classification process and the search for optimal reduct is guided by the inbuilt learning process of the chosen classifier.

Various computational intelligence techniques such as rough set [1], fuzzy rough set [8], Principal Component Analysis (PCA) [23], Independent Component Analysis (ICA) [24, 25], distance based methods [26], clustering techniques [27], entropy and mutual information [28], and swarm intelligence based methods [4,29–34], have been used for feature selection. Some of the swarm intelligence based methods reported in the literature for feature selection are particle swarm optimization (PSO) [4], intelligence dynamic swarm (IDS) [29] and ant colony optimization (ACO) [35].

Rough set technique has been used for feature selection [1]. This technique does not need any other information or parameter beside the data provided. While doing feature selection using rough set, one can use the dependency measure as fitness function. Dependency measure can be computed using lower approximation based positive region.

Jensen *et al.* [8] proposed to use fuzzy lower approximation based fuzzy rough dependency (L-FRFS) as fitness function. This dependency measure is based on fuzzy similarity matrices [11]. The method uses hill climbing to get reduct.

While using hill climbing method [8,9], feature selection may be implemented in the following two ways: (i) Backward elimination and (ii) Forward selection. In backward elimination, insignificant features are eliminated one-by-one to get the optimal reduct whereas in forward selection method optimal reduct is obtained by adding significant features one-by-one.

While doing feature selection we need a method which can consistently give the same performance in terms of selected feature set, number of features and classification accuracy. Fuzzy Rough Feature Selection (FRFS) was proposed in [36]. To reduce the computational complexity in the FRFS approach of [36], and to calculate the fuzzy lower approximation, a compact computational domain based on the properties of fuzzy connectives was proposed in [7]. It was observed that in the method proposed in [7], fuzzy lower approximation was not always a subset of fuzzy upper approximation and cartesian product of fuzzy equivalence classes. Resolving these problems Jensen et al [8] proposed Lower approximation based Fuzzy Rough Feature Selection(L-FRFS) which was based on fuzzy similarity matrices [11]. To get improved reducts, swarm methods of search, such as PSO, Flocks of Starlings Optimization (FSO-RFS) and alignment based FSO (FSO-FS) have been reported [30]. All these works have used dependency measure, used in [8] as fitness function. In this work an attempt has been made to further improve the method reported in [30], by exploring improved and modified versions of existing optimization methods. In this work Lukaceiwicz fuzzy implicator and t-norm has been used to calculate fuzzy lower approximation, fuzzy positive regions and fuzzy rough dependency measure.

1.2 Existing Approaches of Feature Selection

Existing methods for obtaining appropriate reducts are discussed in the following sections.

1.2.1 Transformation based Approaches

Principal component analysis

Principal component analysis is a technique of linear algebra which uses orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called Principal Components. This transformation is defined in such a way that the first principal component has the largest possible variance and each succeeding component in turn has the highest variance possible under the constraint that it be orthogonal to the preceding component. With minimal additional effort PCA reduces a complex data set to a lower dimension, revealing sometimes hidden and simplified structure that often underlies it. PCA is efficient in compressing

information and eliminating correlations between variables [23].

PCA is suitable for identifying linear relationships between features, but if the data is not linearly correlated, PCA is not enough. Further PCA is not suitable when data is having high bias and consists of noise. Sometimes, PCA does not work well if data is not normally distributed because PCs will always be perpendicular to each other assuming that there should be no correlation between these 2 PCs. This assumption will not be satisfied in many kinds of distributions [37].

Independent component analysis

Independent Component Analysis is an unsupervised learning based on high order statistics. For k N-dimensional data vectors it determines up to k number of N-dimensional independent vectors. The transformation of the original feature space into independent components has proved to be useful in an important number of applications including feature selection. Independent Component Analysis is a generalization of Principal Component Analysis [24, 25].

The fundamental principle of ICA is estimating the unmixing matrix to estimate the independent components from the mixtures. Thus, the estimated independent components are the linear combination of the recorded data. If the number of features are more than the number of objects, the estimated independent components must contain some original sources. If some of the original features are predominant, the estimated independent components will be quite similar to the original features. Thus, when the number of objects is less than the total number of features, ICA is able to separate only the components which have relatively high magnitude [38].

1.2.2 Selection based Approaches

Exhaustive Method

While doing feature selection in an N dimensional dataset we should work for every subset of the given dataset, and finally find the best subset of input features. This method is called exhaustive method. Exhaustive search method, ends after exploring 2^N subsets. When N becomes high, number of total subsets increases exponentially, which might become infeasible for doing feature selection task.

To avoid the exhaustive calculations, one can have many choices, some of them are listed as follows.

Hill Climbing Method

For feature selection, hill climbing search approach is used with a suitable fitness function. Feature selection may be viewed as an optimization problem, and if we are able to optimize the fitness value on the basis of some metric based on dataset, it may provide near optimal reduced feature set. Hill climbing methods for selecting the useful features, using Fuzzy rough sets, from a given datasets are suggested in [8] and [9]. Hill climbing [8] and [9] search approach with Fuzzy Rough Feature Selection (FRFS) has also proposed in [36]

Forward Selection Method

In Hill climbing method we add significant features one-by-one is called forward selection methodology. This method uses top down strategy.

Backward Elimination Method

Hill climbing method in which we eliminate features one-by-one which are insignificant is called backward elimination methodology. This method uses bottom up approach. The basis for elimination of features are based on some certain criterion.

Exhaustive method is infeasible for practical application and hill climbing at times give sub optimal results. Hence following simultaneous search methods have been used proposed in the literature.

1.2.3 Simultaneous Methods of Feature Selection

Instead of hill climbing, for improved search, population based search technique viz. Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Intelligent Dynamic Swarm (IDS) etc. have been with a fitness function. PSO has been reported in [4]. Bae et al [29] introduced IDS in 2010 which claimed to be 30% faster than PSO.

Genetic Algorithm (GA)

GA is a heuristic search technique based on the Darwinian theory survival of the fittest. This heuristic is used to generate useful solutions for optimization and search problems. This approach makes no assumptions of relationships among features involved while searching the space for feature selection. GA easily encodes decisions (about selecting any feature) as sequences of Boolean values, allowing for exploration of the feature space by retaining those decisions that benefit the classification task, and simultaneously avoiding local optima due to their intrinsic randomness [39, 40]. Although GA is computationally intractable, it avoids the pitfall of local optima [41].

In the context of feature selection [42, 43], individuals in the population are binary strings, with 1 indicating that a feature was included and 0 indicating that it was not. GA generates the solutions to optimization problems using operators inspired by natural evolution such as selection, crossover, and mutation [43, 44].

Steps of GA

The steps of GA [43, 45] are as follows. The computational procedure involved in maximizing the fitness function $F(x_1, x_2, x_3, ..., x_n)$ in the genetic algorithm can be described by the following steps.

1. Choose a suitable string length N. Assume suitable values for the following parameters: population size p, crossover probability p_c, mutation probability p_m and maximum number of generations to be used as a convergence criterion.

2. Generate a random population of size p, Evaluate the fitness values $F_i, i = 1, 2, ..., p$, of the p strings.

3. Carry out the reproduction or selection process.

4. Carry out the crossover operation using the crossover probability p_c.

5. Carry out the mutation operation using the mutation probability p_m to find the new generation of p strings.

6. Evaluate the fitness values $F_i, i = 1, 2, ..., p$, of the p strings of the new population.

7. If fitness function does not improve or, does not change during last G (specified) number of generations, then stop and return the solution achieved. Otherwise, go to step 2.

Particle Swarm Optimization (PSO)

Fundamentals of PSO

PSO is an evolutionary algorithm based on swarm intelligence, inspired from the social behavior of bird flocking and fish schooling [46–48]. In PSO we initialize probable solutions, called population, randomly with several particles. Based on the values of fitness function, each particle tries to move towards the particles having global and local best values with some velocity.

$$V_i = w * V_i + r_1 * C_1(P_i - X_i) + r_2 * C_2(P_G - X_i) \tag{1.1}$$

$$X_i = X_i + V_i \tag{1.2}$$

where V_i is velocity, and w is inertia weight, r_1 and r_2 are random numbers, C_1 and C_2 are constants. Normally we use $C_1 = C_2 = 2$. The term *pbest* denotes the historical best for a particular particle itself, and *gbest* is the neighborhood best position i.e. best of *pbests*, P_i denotes the i^{th} particle, and X_i is the position of that particle.

In this thesis, wherever PSO is used, it refers to the modified PSO algorithm given in [4].

Modified PSO for feature selection

Wang *et al.* [4] has suggested a modified PSO for feature selection. According to [4], velocity is obtained using following equation.

$$\begin{aligned} V_i = & round(w * V_i + r_1 * C_1 * velocityupdate(Pbest_i - X_i) \\ & + r_2 * C_2 * velocityupdate(Gbest - X_i)) \end{aligned} \tag{1.3}$$

where V_i is an integer value. Function $velocityupdate(Gbest - X_i)$ returns an integer. The working of this function can be explained with an example as follows.

Suppose,

$Pbest_i$ = [0 1 1 0 0 1 1 0 0], and

X_i = [1 1 0 0 1 1 1 0 1]

Then

$(Pbest_i - X_i)$ =[-1 0 1 0 -1 0 0 0 -1]

Number of '1's, $N_1 = 1$

Number of '-1's, $N_{-1} = 3$

The value returned by function $velocityupdate = N_1 - N_{-1}$ = 3 - 1 = 2

The value of V_i obtained in this function is evaluated as follows.

$$V_i = \begin{cases} 1; \text{ if } V_i \leq 1 \\ round(N/3); \text{ if } V_i > N/3, \\ unchanged; \text{ otherwise,} \end{cases} \tag{1.4}$$

where N is the total number of features. Using this value of V_i, particle X_i is updated as shown below.

$$X_i = positionupdate(Gbest, X_i, V_i) \tag{1.5}$$

Working of function *positionupdate* is explained with an example as follows.

Suppose,

$Gbest$ = [1 0 1 0 0 1 0 0 1], and

current X_i = [0 1 1 0 1 1 1 0 0]

Indices of bits dissimilar in $Gbest$ and X_i are [1,2,5,7,9] i.e. number of dissimilar bits $s_i = 5$.

Bits of current X_i which are not matching with corresponding bits of $Gbest$ are highlighted below.

X_i = [**0** **1** 1 0 **1** 1 **1** 0 **0**]

In this algorithm a V_i is chosen such that $V_i \leq s_i$, and randomly V_i bits out of highlighted bits are flipped. Suppose randomly 1^{st}, 5^{th} and 7^{th} bits are randomly flipped, then resulting X_i is as follows.

X_i = [$\boxed{1}$ 1 1 0 $\boxed{0}$ 1 $\boxed{0}$ 0 0],

where bits in $\square$ are obtained after flipping.

Again let $V_i = 6$ i.e. $V_i > s_i$, then we flip every dissimilar elements in current particle X_i to be same as $Gbest$ and out of similar elements, $V_i - s_i$ = 6 - 5 = 1 (bits) are randomly

flipped to facilitate exploration. Suppose, out of similar bits, bit of index number 3 is randomly chosen and flipped, then final X_i will be as follows.

$X_i = [\textcircled{1}\ \textcircled{0}\ \textcircled{0}\ 0\ \textcircled{0}\ 1\ \textcircled{0}\ 0\ \textcircled{1}]$

Bits in ◯ are obtained after flipping.

The pseudocode for the above PSO method is given in *Algorithm 1.*

Steps of modified PSO

1. Initialize the swarm (population) of particles (solutions) randomly.
2. Evaluate the fitness function for each particle.
3. For every particle, compare it's *pbest* value with it's current fitness value. If the current value is better than the *pbest* value, then set this value as the pbest and the current particle's position x_i as P_i.
4. Search the particle that has best fitness value till now, i.e. best of *pbest* values and call it as *gbest* and it's position as P_g.
5. Update the velocities and positions of all the particles using Equations (1.1 to 1.5) and fitness value.
6. Repeat steps 2 to 5 until a stopping criterion is met [4].

Intelligent Dynamic Swarm (IDS) based optimization

Fundamentals of IDS

This method is similar to PSO however it does not have velocity term as defined in PSO. In this method again we initialize the population randomly.

Steps of IDS

1. Initialize the swarm (population) of particles (solutions) randomly.
2. Compute the fitness function for each particle.
3. Compare the fitness value of each particle with it's pbest value if it is greater than pbest value then set or replace current fitness value as pbest.

Algorithm 1 Algorithm for PSO

Initialize population using *Algorithm 2*;
pop_size, The population size;
V_i, Velocity of the i^{th} solution;
X_i, i^{th} solution;
$Pbest_i$, Personal best of i^{th} solution;
Gbest, Global best value of all solutions;
K, Generation number;
MAXITER, Maximum number of generation;
w, inertial weight;
$Pgbest$, Global best solution;
$Fitness_i$, Fitness function of i^{th} solution;
N, The total number of features or dimensions;
r_1, r_2, Random number in range (0.1);
c_1, c_2, Positive acceleration constants;
Initialize $Pbest_i$ with some small value, For every solution i;
Initialize Gbest = max($Pbest_i$);
while $K < MAXITER$ **do**
    $K = K + 1$;
    For every solution i, Compute Fitness Function, $Fitness_i$;
    for solutions i **do**
        if $Fitness_i > Pbest_i$ **then**
            $Pbest_i = Fitness_i$;
        end if
        if $Fitness_i > Gbest$ **then**
            $Gbest = Fitness_i$;
            $gbest = i$;
        end if
    end for
    for solutions i **do**
        $V_i = round(w * V_i + r_1 * C_1 * velocityupdate(Pbest_i - X_i) + r_2 * C_2 * velocityupdate(Gbest - X_i))$
        if $V_i > N/3$ **then**
            $V_i = N/3$;
        else if $V_i \leq 1$ **then**
            $V_i = 1$;
        else
            V_i stays unchanged;
        end if
      $X_i = positionupdate(Gbest, X_i, V_i)$
    end for
    return $Pgbest$;
end while

Algorithm 2 Algorithm for random initialization of population

For every solution i, initialize X_i randomly

if $rand < 0.5$ **then**

$x_{i,j} = 0$;

else

$x_{i,j} = 1$;

end if

4. Search the particle that has best fitness value till now, i.e. best of pbest values and call it as gbest and it's position.

5. For each particle generate a random number r between 0 and 1.
   if $0 \leq r < C_w$ then the current particle will be kept.
   otherwise if $C_w \leq r < C_p$ then the current particle will be replaced by the particle having pbest value.
   otherwise if $C_p \leq r < C_g$ then the current particle will be replaced by the particle having gbest value.
   otherwise if $C_g \leq r < 1$ then the current particle will be replaced by the particle generated randomly.

6. Repeat steps 2 to 5 until a stopping criterion is met.

Parameters, C_w, C_p and C_g need to be specified for the problem. The pseudocode for the above steps of IDS is given in *Algorithm 3* [29].

1.3 Motivation of the Thesis

There are two basic approaches to the feature selection. The first approach is to determine the redundancy of features among themselves whereas the second approach is to consider the performance of the feature subset for a given objective function. Further, the methods applied for feature selection may follow the following three approaches.

1. Hill climbing

2. Simultaneous selection

Algorithm 3 Algorithm for IDS

Initialize population using *Algorithm 2*;
pop_size, The population size;
X_i, i^{th} solution;
$Pbest_i$, Personal best of i^{th} solution;
Gbest, Global best value of all solutions;
K, Generation number;
MAXITER, Maximum number of generation;
$Pgbest$, Global best solution;
$Fitness_i$, Fitness function of i^{th} solution;
N, The total number of features or dimensions;
r, Random number in range (0.1);
c_w, c_p, c_g, Positive constants between (0,1);
Initialize $Pbest_i$ with some small value, For every solution i;
Initialize Gbest = max($Pbest_i$);
while $K < MAXITER$ **do**
    $K = K + 1$;
    For every solution i, Compute Fitness Function, $Fitness_i$;

    for solutions i **do**
        if $Fitness_i > Pbest_i$ **then**
            $Pbest_i = Fitness_i$;
        end if
        if $Fitness_i > Gbest$ **then**
            $Gbest = Fitness_i$;
            $gbest = i$;
        end if
    end for
    for solutions i **do**
        Generate a Random number r between (0,1);
        if $0 < r < c_w$ **then**
            X_i stays unchanged;
        else if $c_w \leq r < c_p$ **then**
            $X_i = Pbest_i$;
        else if $c_p \leq r < c_g$ **then**
            $X_i = Pgbest$;
        else if $c_g \leq r < 1$ **then**
            $X_i = random(X_i)$;
        end if
    end for
    return $Pgbest$;
end while

3. Heuristics based selection

These three approaches are three categories, under which all the feature selection methods can be classified.

As discussed in Section 1.2.2, Hill Climbing [8, 9, 36] method incrementally adds or removes a feature in the selected subset of features, using forward selection or backward elimination methods.

Forward Selection Method: In Hill climbing method we add significant features one-by-one and it is called forward selection methodology. This method uses top down strategy.

Backward Elimination Method: Hill climbing method in which we eliminate insignificant features one-by-one is called backward elimination methodology. This method uses a bottom up approach. The basis for elimination of features is based on certain criteria such as improvement in accuracy, improvement in diversity measure.

Thus, the Hill climbing methods are fast and supposed to be computationally efficient as these methods start from one feature and explore other features. Thus the steps involved in feature selection are fixed. Also, it easily allows to incorporate certain heuristics into the approach. However, Hill climbing method picks one feature at a time and has a tendency to be caught in a local minima set of features as it depends on starting point for search.

Hence, an approach is needed which can avoid local minima. The simultaneous search [4, 29, 43, 44] is a good candidate for this approach. Simultaneous search produces the selected subset of features, in which all the selected or dropped features are suggested simultaneously using random search like GA, PSO etc. Simultaneous method is comparatively inefficient and it takes longer time. However, this approach has a good chance of getting an optimal set of features.

The third category of approach is the Heuristic methods [49, 50], which are strictly problem dependent and use ad-hoc mechanisms such as the level of statistical significance of features based on autocorrelation and correlation coefficients. The methods may also employ the concept of Hill climbing up to certain extent. The method also employs the statistical measures among features themselves to ascertain their acceptance or rejection.

As the Heuristic methods are strictly problem dependent and use ad-hoc mechanisms, these cannot be generalized for the problem of feature selection, whereas simulta-

neous approach has a good chance of getting an optimal set of features even though it is comparatively inefficient and takes longer time. In view of the above, an attempt has been made to make the simultaneous approach computationally efficient to yield practically useful speed while maintaining the optimality of the solution feature set. The thesis also attempts to show that the approaches or methods developed are superior in terms of statistical significance tests.

The present thesis focuses on the simultaneous feature selection approaches for discernibility performance of the data. In this thesis, three metrics are used to formulate the objective function based on which feature selection is performed. These three metrics (measures) are (i) classification accuracy based on classification methods such as J48, (ii) dependency measure based Rough Set (RS), and (iii) dependency measure based on Lower Approximation based Fuzzy Rough Set (L-FRS).

J48 classifier is a Java version of decision tree based C4.5 classifier [51].

The tuple $\langle \underline{P}X, \overline{P}X \rangle$ is called a Rough Set [1] where,

$$\underline{P}X = \{x \in U | [x]_P \subseteq X\} \tag{1.6}$$

$$\overline{P}X = \{x \in U | [x]_P \cap X \neq \phi\} \tag{1.7}$$

P is a set of conditional features. $[x]_P$ denotes the equivalent classes of the P-indiscernibility relations. $X \subseteq U$ and $I = (U, A)$ be an information systems, where U is a non-empty set of finite objects and A is the nonempty finite set of features including the class labels.

Rough sets can do classification task in rough sense by capturing the structural relationship within a data. However, this method is applicable to discrete-valued features and hence continuous valued classification tasks requires discretization of continuous valued features.

Rough set establishes a set of pair of so called upper and lower approximations on a set. Lower approximation is a rough set which classifies the features uniquely i.e. without ambiguity whereas the upper approximation gives a non-unique classification of features. In this work, a POSitive region (POS) defined by lower approximation is used to evaluate a dependency measure which describes the dependency of a given feature set to a class

label in a rough sense. This so called rough dependency measure is used as a component in a multi-objective formulation of the problem.

In the crisp case, elements that belong to lower approximation (i.e. have a membership value 1) are said to belong to the approximated set with absolute certainty.

In the fuzzy rough case elements may have membership in range [0,1], which allows greater flexibility in handling uncertainty and vagueness. Fuzzy-rough sets incorporate the distinct concepts of vagueness (for fuzzy sets) and indiscernibility (for rough sets), both occuring as a result of uncertainty in knowledge [8, 9].

The tuple $< \mu_{\underline{P}X}, \mu_{\overline{P}X} >$ is called a fuzzy rough set. Fuzzy P-lower approximation, $\mu_{\underline{P}X}$ and fuzzy P-upper approximation, $\mu_{\overline{P}X}$ can be computed using the definition of fuzzy rough sets with the following formula suggested in [6].

$$\mu_{\underline{P}X}(F_i) = \inf_x max\left\{1 - \mu_{F_i}(x), \mu_X(x)\right\} \forall i, and \tag{1.8}$$

$$\mu_{\overline{P}X}(F_i) = \sup_x min\left\{\mu_{F_i}(x), \mu_X(x)\right\} \forall i, \tag{1.9}$$

where F_i is a fuzzy equivalence class, and X is the (fuzzy) concept to be approximated.

Fuzzy rough set based dependency measure is also used in the objective function due to following motivations.

1. For the case of real valued features, for applying rough set method, we need to discretize the feature values. The discernibility of features are affected by the quantization and therefore becomes dependent on the quantization of feature values.

2. In case of fuzzy rough sets, the real feature values are taken as it is and therefore no such quantization is needed and discernibility of features are therefore more accurate as well as meaningful.

3. For real valued features the number of quantization levels can be large or infinite. Thus to capture the feature values in discrete sense is not simply possible.

4. If a real valued feature based problem is solved using rough set through discretization, it is highly possible that while application, a newer intermediate value for which the quantization level was not fixed may arise, making the rough set based method of no use since the nominations to values are predefined. In other words,

the new value is a new nomination and the rough set based feature reduction has to be done once again including the new nominations.

1.4 Outline of the Thesis and Main Contributions

Chapter 1 of the thesis introduces the problem of feature selection in the context of methods used for the purpose in this thesis. Some of the swarm intelligence based methods have been reported in literature for feature selection which are particle swarm optimization (PSO), intelligence dynamic swarm (IDS) and ant colony optimization (ACO).This chapter also introduces methods of Particle Swarm Optimization (PSO), Genetic Algorithm (GA) and Intelligent Dynamic Swarm (IDS). The measures of fitness values used in the thesis are also discussed in this chapter.

This thesis work presents feature selection methods using swarm based algorithms, with dependency measures as fitness function, using rough and fuzzy rough sets. Inadequacy of hill climbing methods (as they may stuck in local optima) prompted to work with swarm based algorithms and genetic algorithm (GA) for simultaneous selection of features.

Chapter 2 introduces Elitist GA (eGA) based algorithm for feature selection considering classification accuracy as objective function. The performance of eGA on various data sets has been tested and the results have been compared with existing feature selection methods which use Fuzzy Rough Set (FRS) based measures. The method of eGA has been applied for feature selection of variables for short-term price forecasting problem also. The main contribution of this chapter is that it established the common ground for comparison of method/approaches proposed in subsequent chapters. The chapter also establishes reasons for preferring rough set and fuzzy rough set based fitness functions over classification accuracy based fitness function.

The fundamentals of Rough Set (FS) and Fuzzy Rough Set (FRS) in the context of feature selection problem has been introduced in Chapter 3. The calculation of performance measures used in the further chapters have been demonstrated in detail. The chapter prepares a background for understanding Rough sets and Fuzzy Rough Sets used in further chapters.

Chapter 4 discusses the development and implementation of Distributed Sample

(DS) initialization proposed in this thesis. This proposed initialization method was implemented in PSO and IDS methods and was tested for its generality. The main contribution of this chapter is development and validation of a new initialization method which can be embedded in swarm and evolutionary algorithms.

In Chapter 5 proposes a hybridized version of PSO and IDS. This hybridization of methods has been carried out in the sense of series implementations of PSO and IDS. The implementation is further tested on various data set and its performance has been demonstrated. The advantages of hybridization of PSO and IDS has been demonstrated through comparison with existing PSO and IDS as well as DS initialized PSO and IDS. The main contribution of this chapter is the development of hybrid structure for PSO and IDS and demonstration of the effectiveness of the approach on various datasets and initializations.

Chapter 6 introduces a Butterfly Optimizer (BO) for the purpose of feature selection. As the BO supports real coded variables, two methods are devised to handle requirements of binary coding in the feature selection problem. The results of feature selection on various datasets are compared with the hybrids methods. The main contribution of this chapter is application of BO method for feature selection and testing the effectiveness on various datasets.

Chapter 7 proposes an algorithm named Adaptive Genetic Algorithm with Modified Operator (AGA-MO) for the problem of feature selection. The algorithm is modification of GA in in such a way that the mutation operator is adaptively dependent on the distribution of particles in a population. When particles are diverse, the probability of mutation is high and when the population is converging, the mutation is reduced. The test results of application of AGA-MO establishes its superiority over existing eGA and BO algorithms. The main contribution of this chapter is development of a new evolutionary optimization method and its validation on feature selection problems.

Chapter 8 summarizes the overall conclusions of various chapters of the thesis. Highlights and the comparative outcomes of various chapters of the thesis are summarized in this chapter.

For the datasets reported in this work, the evaluation metric (the reduct size and accuracy) corresponding to the existing top performing (state-of-the-art) feature selection methods have been reported in Table 1.1

Table 1.1: Average reduct size using state-of-the-art methods

Dataset	Best result reported in literature	
	Average Reduct size	Accuracy
Cleveland	7.81 [30]	52.6 [30]
Ecoli	3 [52]	77.38 [52]
Glass	8.44 [30]	65.14 [30]
Ionosphere	7.3 [30]	86.17 [30]
Lung	NA	NA
Soybean small	2 [29]	100 [29]
Wine	2 [52]	90.44 [52]
LSVT	NA	NA

Evaluation metric reported in the Table 1.1 corresponding to dataset Cleveland and Ionosphere are for the method FSO-FS [30], corresponding to the dataset Glass are for the method FSO-RFS [30], corresponding to the dataset Ecoli and Wine are for the method USQR [52], and corresponding to Soybean small are for the method IDS with RST [29].

The method adopted in chapter wise presentation of work in the thesis is as follows. The candidate has first explained the method developed/proposed and compared the results obtained to that of earlier chapters.

A logical ground of improved objective function is taken as we go along the thesis. As far as eGA and adaptive GA (AGA-MO) methods are concerned, the result of AGA-MO are superior to the method reported in earlier chapter and therefore, to maintain the continuity of approach of presentation the AGA-MO was described along with the result obtained in Chapter 7.

The main contribution in Chapter 2 is that it established the common ground for comparison of method / approaches proposed in subsequent chapters. The Chapter also establishes reasons for preferring rough set and fuzzy rough set based fitness functions over classification accuracy based fitness function.

In Chapter 2, the fitness function is different from other chapters and the search method is eGA. After this, in Chapter 3, the theoretical concepts of Rough set theory and Fuzzy rough set theory with example is discussed, and is used in subsequent chapters

to form fitness functions for various used and proposed search methods.

From Chapter 4 to 7 work has been carried out on search methods whereas the fitness function is based on Rough set theory and Fuzzy rough set theory.

Further, starting from Chapter 4, first of all improvement in initialization in PSO and IDS is proposed and is compared with randomly initialized PSO and IDS for both RST and L-FRFS approaches.

Incrementally, in Chapter 5, further improvement in algorithm is done by hybridization and is compared with the method proposed in Chapter 4.

In the Chapter 6, a newly proposed Butterfly Optimizer (BO) is used to further improve the performance of the search for the same feature selection problems. The result from BO is also compared with the proposed hybrid algorithm of PSO and IDS, and shown little improvement and is established with the statistical tests.

In the Chapter 7, a new Algorithm Adaptive Genetic Algorithm with modified Operator (AGAMO) is proposed and is compared with the top performer and the eGA method (with RST and L-FRFS based fitness function) and the supremacy of AGAMO is established with statistical tests. Here eGA is compared since the proposed algorithm is the modified version of eGA, so we wanted to see the comparative performance.

In this way we have incrementally improved the performances of the search method.