Enhanced Binary Cuckoo Search With Frequent Values and Rough Set Theory for Feature Selection

AHMED ALIA\textsuperscript{1,2}, AND ADEL TAWEEL\textsuperscript{3}, (Member, IEEE)

\textsuperscript{1}Institute for Advanced Simulation, Forschungszentrum Jülich, 52425 Jülich, Germany
\textsuperscript{2}Faculty of Engineering and Information Technology, An-Najah National University, Nablus, Palestine
\textsuperscript{3}Department of Computer Science, Birzeit University, Ramallah, Palestine

Corresponding author: Ahmed Alia (a.alia@fz-juelich.de)

This work was supported in part by the German Federal Ministry of Education and Research (BMBF) under Grant 01DH16027 within the framework of the Palestinian-German Science Bridge Project, and in part by the Central Library, Forschungszentrum Jülich, Germany.

\textbf{ABSTRACT} Redundant and irrelevant features in datasets decrease classification accuracy, and increase computational time of classification algorithms, overfitting problem and complexity of the underlying classification model. Feature selection is a preprocessing technique used in classification algorithms to improve the selection of relevant features. Several approaches that combine Rough Set Theory (RST) with Nature Inspired Algorithms (NIAs) have been used successfully for feature selection. However, due to the inherit limitations of RST for some data types and the inefficient convergence of NIAs for high dimensional datasets, these approaches have mainly focused on a specific type of low dimensional nominal dataset. This paper proposes a new filter feature selection approach based on Binary Cuckoo Search (BCS) and RST, which is more efficient for low and high dimensional nominal, mixed and numerical datasets. It enhances BCS by developing a new initialization and global update mechanisms to increase the efficiency of convergence for high dimensional datasets. It also develops a more efficient objective function for numerical, mixed and nominal datasets. The proposed approach was validated on 16 benchmark datasets; 4 nominal, 4 mixed and 8 numerical drawn from the UCI repository. It was also evaluated against standard BCS; five NIAs with fuzzy RST approaches; two popular traditional FS approaches; and multi objective evolutionary, Genetic, and Particle Swarm Optimization (PSO) algorithms. Decision tree and naive Bayes algorithms were used to measure the classification performance of the proposed approach. The results show that the proposed approach achieved improved classification accuracy while minimizing the number of features compared to other state-of-the-art methods. The code is available at https://github.com/abualia4/EBCS.

\textbf{INDEX TERMS} Feature selection, classification, machine learning, rough set theory, enhanced binary cuckoo search, nature inspired algorithms, optimization problem.

\section{INTRODUCTION}
The increasing availability and use of data acquisition technologies is leading to exponential collection of data \cite{1}. The number of structured datasets is growing rapidly, and these datasets are becoming larger and more complex \cite{2}, \cite{3}. This increases the importance of, as well as the demand on, data mining algorithms to extract more accurate knowledge automatically from such large and complex datasets \cite{3}–\cite{5}. Classification is an important approach to help extract useful knowledge. It often employs data mining algorithms with a goal to classify each record in the structured dataset into different classes \cite{6}. The large number of features in the datasets however causes a major problem for classification algorithms known as the curse of dimensionality \cite{7}, \cite{8}. This problem increases both computational time, overfitting problem and complexity of the classification model and reduces classification accuracy \cite{8}, \cite{9}. Specifically, redundant features are seen as the main reason for the overfitting problem \cite{8}. Feature selection (FS) is one of the approaches that aims to solve the dimensionality problem by selecting the most relative features from datasets for classification.
algorithms [8], [10], [11]. In other words, to address these issues, these approaches aim to remove both irrelevant and redundant features.

In general, efficient FS approaches aim to select the minimum number of needed features without significantly reducing classification accuracy or substantially increasing computational time [8], [12], [13].

A typical FS approach employs three main steps: firstly, the subset generation, which is responsible for generating candidate feature subsets; secondly, the subset evaluation, which is responsible for evaluating the selection [14]. The search uses three strategies to search for candidate feature subsets, which are complete, random and heuristic search [8]. Complete search is computationally expensive because it covers all combinations of features. Random search generates a subset of features randomly. Heuristic search is faster than complete search and more efficient than random search because it makes smart choices to select the near optimal subset without searching in all combinations of features. Result validation is the last step, which is used to validate and subsequently determine the performance of the FS approach [14].

Meta-heuristic needs less number of assumptions to find the near optimal feature subset compared to heuristic search [15]. FS is an optimization problem [16]. One of the most efficient popular methods used to generate candidate solutions for optimization problems is Nature Inspired Algorithms (NIAs) algorithms [17]–[19]. NIAs, developed using characteristics of biological systems, are a population meta-heuristic class of algorithms that improves concurrency of computing multiple candidate solutions [20]–[22]. Ant Colony Optimization (ACO) [23], Particle Swarm Optimization (PSO) [24], Artificial Bee Colony (ABC) [25], Cuckoo Search (CS) [26] and Binary CS (BCS) [27] are examples of popular NIAs. Moreover, NIAs are widely used for FS [28], [29].

The objective function is responsible for determining the relevancy of the generated candidate feature subsets for the classifier algorithms [30]. For the objective function, FS approaches can be classified into two groups: filter approaches and wrapper approaches [31], [32]. Filter approaches use statistical methods in their objective function to evaluate feature characteristics such as dependency degree [33] or information measure [34]. Wrapper approaches, on the other hand, use classification algorithms in their objective function [35]. Filter approaches are often considered more general, have lower probability of overfitting and much faster compared to wrapper approaches [8]. Rough Set Theory (RST) or Dependency Degree RST (RSTDD) is widely used in filter FS approaches’ objective functions for measuring the dependency between the feature subsets and class labels [36]. RSTDD have shown several advantages over other methods including greater efficiency, faster computation, and does not need any preliminary or additional information about the data [36]. These approaches usually work well for nominal datasets, however, they suffer from several inefficiency limitations when applied on mixed and numeral as well as high dimensional datasets. Therefore, the paper proposes a new more efficient filter feature selection for classification approach, named Enhanced Binary Cuckoo Search (EBCS), for three types of datasets: nominal, mixed and numerical. EBCS uses RSTDD to develop a new objective function and employs frequent values and Rough Set Theory for feature selection.

A. THE INNOVATION

Building on the strengths of RSTDD and BCS, the proposed approach enhanced BCS and developed a new objective function with a more efficient filter FS in classification for nominal, mixed and numerical datasets. Section 2 describes the main limitations of the current hybrid NIAs and RSTDD approaches in more details, which include inefficiency for mixed and numerical datasets [37], [38], and weak convergence for high dimensional datasets [39]. EBCS, on the other hand, achieves reduced number of selected features in less computational time and without significant reduction in classification accuracy for mixed, nominal and numerical datasets with varied sizes on number of features, objects and classes. The original contributions of the paper include, firstly, the development of a new improved classification filter feature selection approach (EBCS), with a new initialization mechanism that covers most of the search space, and a new global updating mechanism that has more efficient convergence. Secondly, the development of a new EBCS objective function to produce a reduced feature subset that achieves maximum classification accuracy and minimum number of features for nominal, mixed and numerical datasets by balancing between RST, number of features and frequent values.

The remainder of this paper is organized as follows. Section II reviews related work of FS in classification, which used NIA and RST. Section III discusses briefly the basic concepts of RST, and introduces CS and BCS. In section IV the new proposed approach is described. The evaluation methodology is presented in section V. Section VI discusses experiments’ results. The conclusion is included in section VII.

II. RELATED WORK

In the literature, there are many studies on filter FS in classification, each uses different search methods and objective functions to find the most relevant features. Three traditional filter FS approaches were proposed by [40], [41]. In [40], Karthikeyan et al. proposed two filter FS approaches, the first approach combined best-first search method with Correlation-based Feature Selection (CFS), while in the second approach, a greedy search method was combined with CFS. On the other hand, CONsistency based feature selection (CON) with different traditional search methods, such as forward selection, was presented in [41]. To improve the search strategy in filter FS, many researchers used one or more NIA for exploring the search space. Recently, Zhang et al. developed two new filter FS-based NIAs, ABC was enhanced and used in the first approach [42], while Differential Evolution was improved and employed in the...
second approach [43]. The authors of [44] integrated Genetic Algorithm (GA) with CFS to address the filter FS problem. PSO and CFS were combined in [45] to develop a new filter FS. Hybrid filter FS approaches based on NIAs and RST have achieved significant results with low computational time, especially, RST, which is both efficient and fast [38], [46]–[48]. In this section, we focus on reviewing this type of hybrid approaches.

ACO is an NIA which simulates the behavior of real Ants to find the shortest path between their nest and the food source [23]. ACOAR [46], ACOFS [49], AnTRSAR [50] and ARRST [51] employed ACO and RST for filter FS approaches. ACOAR, ACOFS and AnTRSAR were developed for low dimensional nominal datasets, while ARRST was developed for low dimensional numerical datasets. ARRST used RST’s principles of in-discriminability to discretize the numerical values (convert numerical datasets to nominal datasets). ACOAR achieved maximum classification accuracy with minimum feature subset compared to other approaches, because it limits the value of pheromone and heuristic information, which prevents the algorithm from going away from the optimal feature subset. AnTRSAR, however, is more expensive than ACOFS and ACOAR. It used entropy in its heuristic information update, which is a more costly operation than that of RSTDD, which was used in ACOFS and ACOAR. However, ARRST was shown as the slowest because the discretization of numerical data is computationally expensive and takes significant time to perform. On the other hand, experimental results showed that ARRST succeeds in maximizing the classification accuracy in four out of the seven tested datasets, although, most of the features in these datasets are nominal. All these approaches used RSTDD in their objective function, but ACOAR used two user stable parameters to control the importance of the number of features and their quality (i.e. classification accuracy). AnTRSAR achieved balancing between the number of features and classification accuracy automatically.

In general, ACO uses a graph to represent its search space and employs several variables for configuration. However, these approaches are more complex, more expensive, slower, with weak convergence [52], [53] and are therefore inefficient for high dimensional datasets.

PSO is another NIA, which simulates the movements of flocks of birds around food sources [24]. Several filter FS approaches exist in the literature that combined PSO with RSTDD, such as PSORSFS [47], PSO-RR [48], PSO-QR [54], PSO-RS [55] and EPSORSNA [56]. PSORSFS was applied on low dimensional, general, datasets. PSO-RR, PSO-QR, and PSO-RS were also applied on low dimensional medical datasets. EPSORSNA used PSO, RST and quick reduce algorithm for general structured datasets. However, all of these datasets consist of features that have nominal characteristics. The big difference between these approaches is the objective function. PSORSFS developed a flexible objective function based on RSTDD with two simple parameters to control the importance of classification accuracy and the number of selected features. While PSO-RR and PSO-QR used relative-dependency and RSTDD respectively, and focused on classification accuracy only. The objective function in PSO-RS used RSTDD, but it used cardinality and two variables to select the minimum feature subset with maximum classification accuracy, which makes PSO-RS computationally inefficient. EPSORSNA, on the other hand, combined RSTDD with a quick reduction algorithm and group of parameters to minimize the number of selected features with at least the same classification accuracy as the original nominal datasets.

In general, PSO is easier to implement, requires fewer parameters, and is less expensive compared to ACO. However, because it uses the global search mechanism and poor initial solutions, it has weak convergence [57]. Similarly PSORSFS, PSO-RR, PSO-QR, and PSO-RS have weak convergence for high dimensional datasets [47], [54], [55].

ABC is another popular NIA inspired by the natural foraging behavior of honey bees [25]. Several approaches combined it with RST concepts to develop a filter FS, including NDABC [58], BeeRSAR [59], BeeQR [60], and ABC-FTSBPSD [61]. NDABC has been applied on a low dimensional nominal dataset, while both BeeRSAR and BeeQR have been applied on five low dimensional nominal medical datasets. ABC-FTSBPSD, however, has been applied on low dimensional mixed datasets with nominal and numerical features. This approach, in its initial step, “discretizes” the numerical features. This can be very complex and inefficient, and thus results in low performance compared to other approaches [61]. NDABC, on the other hand, used a meta-heuristic search and greedy selection algorithm to avoid falling into a local optimum and speed up convergence. For its objective function, NDABC uses RSTDD and one user configurable parameter to control the important balance between the number of features and classification accuracy. While other approaches used the principle of in-discriminability in their objective function with no user-balancing configuration, thus making them easier to use.

In general, ABC approaches are more efficient compared to those of ACO’s and PSO’s, but they have relatively weak convergence, are relatively slow, and may fall into local optimum, especially when applied to high dimensional datasets [52].

Ibrahim et al. [62] combined the rough sets and neighborhood rough sets with a runner-root algorithm (RRA) to select the relevant features in nominal (discrete), mixed and numerical (continuous) structured datasets. This approach used rough sets and neighborhood rough sets separately as an objective function to work with different types of datasets. The improved runner-root algorithm, on the other hand, applied many operators to decrease the probability of being trapped into local optimum. In the literature, this approach tried to solve the FS on different types of structured datasets by using two objective functions; one was based on RST and the other combined RST with Euclidean distance, which resulted into a high computational cost. Das et al. [63]
proposed a group incremental feature selection for classification using rough set theory based on a genetic algorithm. In this approach, RST was used to remove imprecise, vague and inconsistent data during feature selection. The objective function applied the positive region and the previously generated reduction to select the relevant features in small nominal datasets. In [64], the authors proposed an ant lion optimizer, which is mainly based on the hybridization between rough set and conditional entropy. This work tried to improve the quality of the initial population, and eventually the final optimal solution, but the conditional entropy and different approaches that were used increased the computational time. In addition, the approach failed to address the numerical datasets. One of the goals of the work [65] was proposing a new FS approach that employed a RST and a binary version of the water wave optimization approach. The objective function used two parameters with RST to decrease the number of selected features and to increase the values obtained by RST. The approach based on RST objective function was evaluated on sixteen datasets with less than 69 features and most of them have nominal features, especially, RST’s dependency degree is inefficient for numerical features. Moreover, the binary version of the water wave optimization requires a lot of computational operations. This approach is efficient for small nominal datasets, and the number of computational operations increases the complexity of computational time. The authors of [66] proposed a filter-based FS approach by combining the Master River Multiple Creeks Intelligent Water Drops (MRMC-IWD) model with different objective functions, such as fuzzy RST [67], to solve FS. MRMC-IWD with fuzzy RST produced minimum feature subset, but without maximum classification performance. Similarly, Diao and Shen [68] developed a novel filter FS approach based on the Harmony Search (HS) algorithm and several objective functions; fuzzy RST is one of these functions. To improve the performance of HS, the authors introduced some methods to select the initial parameters dynamically, and the search process was modified to select small feature subsets while keeping classification performance. In the experiments, the authors combined three NIAs including genetic algorithms (GA) [69], PSO [47] and hill climbing (HC) [70] with fuzzy RST to evaluate HS based on fuzzy RST, results showed that it achieved the best feature reduction, but not the best classification accuracy.

Cuckoo Search (CS) is, an efficient NIA algorithm, developed by Yang and Deb [26]. It mimics the reproduction strategy of cuckoo birds. It has been used in several different domains due to its improved efficiency [71]–[73]. It employs a more efficient search mechanism and requires fewer number of user-configurable variables compared to ACO, PSO, genetic algorithm, RRA, HS, HC and ABC. Generally, CS is considered relatively easy to implement, and has fast and efficient convergence. Binary CS (BCS) is a binary version of CS that uses binary representation [27].

Recently, a number of hybrid filter FS approaches based on BCS and RST principles have been proposed, including the Modified cuckoo search algorithm with rough sets for feature selection (MCRS) [37], feature selection based on Hybrid Binary Cuckoo Search and Rough Set Theory in Classification for Nominal Datasets (FS-BCS) [38] and a hybrid Rough Set Feature Selection using Cuckoo Search Optimization (RSFSCSO) [74]. Both MCRS and FS-BCS have been applied on low dimensional nominal datasets, while RSFSCSO has been applied on a Malware classifier dataset built with header fields values of Portable Executable files (69 features). MCRS and FS-BCS used RSTDD in their objective function with parameters to balance the number of selected features and classification accuracy. However, MCRS is enhanced over FS-BCS by two features: it changed $\alpha$ from constant to variable, which decreases as the number of iterations increases, to avoid drifting away from the best feature subset, and it used parallel techniques to update the population of candidate feature subsets. In other words, FS-BCS combined standard BCS with RSTDD without any modification to solve low nominal dimensional datasets, while MCRS used RSTDD and BCS with $\alpha$ modification to address the low nominal dimensional datasets. On the other hand, RSFSCSO did not use any parameter to balance between the number of selected features and classification accuracy, which means the approach tries to find the solution which has the highest RSTDD regardless of the number of selected features. However, All these approaches suffer from two main drawbacks: Firstly, weak convergence when they are applied to high dimensional datasets [39], because BCS’s initial and global mechanisms do not cover the large search space. This is described in more detail in subsection IV-B. Secondly, it is inefficient for numerical and mixed datasets, because their objective function, which uses RSTDD, is only efficient for nominal datasets [37], [38].

To overcome the above limitations, this paper proposes a more efficient filter FS by enhancing BCS, and developing a new objective function based on frequent values and RST. To increase the convergence efficiency of BCS, the paper developed a new initialization mechanism that covers most of the search space and a new global updating mechanism that has more efficient convergence for both low and high dimensional datasets. This also makes the proposed approach more efficient for nominal, mixed and numerical datasets. The paper also developed a new EBCS objective function to produce a reduced feature subset that achieves maximum classification accuracy and minimum number of features by balancing RST, number of features and frequent values. This approach aims to reduce the number of selected features with improved computational time and improved or similar classification accuracy for mixed, nominal, and numerical high dimensional datasets. The following section describes some of the preliminary concepts used in the proposed approach.

### III. PRELIMINARY CONCEPTS

This section provides a general discussion of the background concepts used in the rest of the paper.
A. ROUGH SET THEORY (RST)
Zdzisław Pawlak developed Rough Set Theory (RST) in 1982 [75]. RST is a mathematical tool for data analysis and data mining. RST has been applied in many domains for several reasons: it provides an efficient approach for discovering useful patterns in data, is easy to implement and understand, and depends on input data alone [76]. Basic concepts of RST are discussed below.

1) INFORMATION TABLE OR INFORMATION SYSTEM
The dataset is called an information table or an information system. Let \( I = (U,A) \) where \( I \) is the information table, \( U \) is a nonempty set of finite objects and \( A \) is a nonempty set of features. In the table 1, \( \{a,b,c,d\} \) are features, but the features consist of four conditional features \( \{a,b,c,d\} \) and one decision feature(Class) \( D = \{e\} \). \( U = \{0,1,2,3,4,5,6,7\} \) are objects [76].

| \( X \in U \) | a | b | c | d | e |
|---|---|---|---|---|---|
| 0 | 1 | 0 | 2 | 2 | 0 |
| 1 | 0 | 1 | 1 | 1 | 2 |
| 2 | 2 | 0 | 0 | 1 | 1 |
| 3 | 1 | 1 | 0 | 2 | 2 |
| 4 | 1 | 0 | 2 | 0 | 1 |
| 5 | 2 | 2 | 0 | 1 | 1 |
| 6 | 2 | 1 | 1 | 1 | 2 |
| 7 | 0 | 1 | 1 | 0 | 1 |

2) INDISCERNIBILITY
Any subset \( P \) of \( A \) determines a binary relation \( \text{IND}(P) \) on \( U \), which is called an indiscernibility relation. Indiscernibility \( \text{IND}(X) \) is equivalence relation on the set \( U \), where all the values are identical in relation to a subset of attributes [50]. For example, if \( P = \{b,c\} \), then objects 1, 6 and 7 are indiscernible; as are objects 0 and 4. \( \text{IND}(P) = \{\{2\},\{0,4\},\{3\},\{1,6,7\},\{5\}\} \).

3) POSITIVE REGION (POS)
Let \( P \) and \( Q \) be equivalence relations over \( U \), then the positive region contains all objects of \( U \) that can be classified as classes of \( U/Q \) using information in attributes/features \( P [50] \). In other words, objects that have the same values in their features are classified into the same class. For example (Table 1), let \( P = \{a,b,c\} \) and \( Q = \{e\} \), now we need to find the objects that have the same values in \( a \), \( b \), and \( c \) features, and at same time belong to the same classes. Table 2 explains the objects that have the same values in \( a \), \( b \), and \( c \) features by highlighting the rows with the same colours. The \( \{0,4\} \) objects have the same values in \( a \), \( b \), and \( c \) features, but they do not belong to the positive region because they are in different classes. The same applies for the \( \{1,7\} \) objects. The remaining objects have different values in their features \( (a,b,c) \) compared to other objects and belong to one class. This means that 2, 3, 5, and 6 objects are included in the positive region, \( \text{pos}_p(Q) = \bigcup\{\{2\},\{3\},\{5\},\{6\}\} = \{2,3,5,6\} \).

4) DEPENDENCY DEGREE
is a very important issue in data analysis to discover the dependencies between attributes. For \( P, Q \subseteq A \), if all attribute values from \( Q \) are uniquely determined by the values of the attributes from \( P \) then \( Q \) depends totally on \( P \). If some of the values of \( Q \) partially depend on \( P \) then \( Q \) depends on \( P \) with a degree of \( K(0 \leq K \leq 1) \). If \( K = 0 \), then \( Q \) does not depend on \( P \). The dependency degree can be defined by the equation 1 [76]:

\[
\gamma_P(Q) = K = \frac{|\text{pos}_p(Q)|}{|U|}
\]  
(1)

where \( |U| \) is the total number of objects, \( |\text{pos}_p(Q)| \) is the number of objects in a positive region, and \( \gamma_P(Q) \) is the dependency between the feature subset \( P \) and classes \( Q \).

For example (Table 2), \( \gamma_{a,b,c}(e) = \frac{|\{2,3,5,6\}|}{|\{0,1,2,3,4,5,6,7\}|} = \frac{4}{8} \).

Many filter FS in classification approaches use dependency degree to build their objective function to guide the search algorithms to optimal/nearest solution by measuring the dependencies between the feature subsets and class labels [58], [77]. The frequently used objective function that uses dependency degree balanced with the number of selected features is(Equation 2) [50]:

\[
\text{Objective Function}(P) = \gamma_P(Q) \times \frac{|U| - |P|}{|U|}
\]  
(2)

where \( |U| \) is the total number of features.

B. CUCKOO SEARCH
This section briefly reviews the cuckoo birds, Lévy flight of birds, CS algorithm and BCS algorithm.

1) CUCKOO BREEDING BEHAVIOR AND LÉVY FLIGHT
a: CUCKOO BREEDING BEHAVIOUR
The reproduction strategy of cuckoo birds is aggressive. Cuckoos use the nests of other host birds to lay their eggs in and rely on these birds for hosting the eggs. Sometimes the other host birds discover the strange eggs and they either throw these strange eggs out, or leave their nest and build a new one. Cuckoos lay eggs that look like the pattern and color of the native eggs to reduce the probability of discovering...
them. If the egg of the cuckoo hatches first, then the cuckoo chick destroys all eggs in the nest to get all the food provided by its host bird [26], [78].

b: LÉVY FLIGHT
In nature, many animals and insects search for food by moving to the next location based on the current location. This behavior of search called Lévy flight, which is a special case of random walks where the step size has a Lévy probability distribution to improve convergence [79]. Step sizes is the main factor of the efficiency of Lévy flight search. Lévy flight is modelled in equations 3 and 4.

\[ X_{t+1}^i = X_t^i + \alpha \odot L'\text{evy}(\lambda) \]  

where \( X_{t+1}^i \) represents the next solution, \( X_t^i \) is current solution, \( t \) is the iteration, and \( \alpha \) is the step size \((\alpha > 0)\) scaling factor of the problem. In most cases we can use \( \alpha = 1 \). \( \odot \) means entry-wise multiplications, and \( \lambda \) is Lévy distribution coefficient \((0 < \lambda \leq 3)\). Random step length(Lévy(\lambda)) is calculated from the power law by equation 4.

\[ L’\text{evy } u = t^{-\lambda} \]  

2) CUCKOO SEARCH AND BINARY CUCKOO SEARCH ALGORITHMS
Cuckoo Search (CS) is an efficient NIA algorithm that has been developed by Yang and Deb [26]. CS depends on three rules [26]:

1) Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest.
2) The best nests with high quality eggs (solutions) will carry over to the next generations.
3) The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability \( P_a \in [0, 1] \).

In more detail, each nest represents a solution and CS aims to replace the “not so good” solutions (nests) with a new one that is better. CS starts to generate the population of nests (solutions) randomly, then in each iteration it uses the threshold value \((P_a \in [0, 1])\) to find the nests that have the lowest quality, which are updated using Lévy flight. Population update is repeated until the maximum number of user-defined iterations are reached [26].

Binary Cuckoo Search (BCS) [27] is a binary version of CS, in which search space is modelled as a n-binary vector. For FS, BCS represents each nest as a candidate features subset and each egg represents a feature where 1 corresponds to a selected feature and 0 corresponds to not selected [27]. Recall that each solution in BCS must be a binary vector, but Lévy flights does not return a binary bit. Therefore, BCS uses equations 5 and 6 to create a binary vector for each new candidate solution [27].

\[ S(x_{ij}^t) = \frac{1}{1 + e^{-x_{ij}^t}} \]  

\[ x_{ij}^{t+1} = \begin{cases} 1, & S(x_{ij}^t) > \sigma \; \text{Otherwise} \\ 0, & \end{cases} \]  

where \( \sigma \in U(0, 1) \), in iteration \( t \).

BCS Algorithm 1 presents the main steps of the standard BCS.

Algorithm 1 Binary CS (BCS)

1: Input:
2: \( N \leftarrow 20; \; \{\text{Number of nests}\} \)
3: \( P_a \leftarrow 0.5; \; \{\text{Probability of } P_a \in [0, 1]\} \)
4: \( T \leftarrow 20; \; \{\text{Maximum number of iterations}\} \)
5: Output: bestNest \( \leftarrow \{} \; \{\text{Best global feature subset}\} \)
6: Initialization:
7: Set \( n \leftarrow 1. \)
8: while \( n \leq N \) do
9: \( \text{nest}[n] \leftarrow \text{random} \)
10: Evaluate \( \text{nest}[n] \) using eq. 2
11: end while
12: Rank the nests based on their quality and find the best nest.
13: bestNest \( \leftarrow \) best nest.
14: Updating:
15: \( t \leftarrow 1. \)
16: while \( t \leq T \) do
17: Global Search:
18: Set \( n \leftarrow 1. \)
19: while \( n \leq N \) do
20: Update \( \text{nest}[n] \) randomly
21: \( \text{Quality(\text{nest}[n])} \leftarrow \text{Evaluate \text{nest}[n] using eq. 2} \)
22: end while
23: Local Search:
24: Set \( n \leftarrow 1. \)
25: while \( n \leq N \) do
26: if \( \text{Quality(\text{nest}[n])} \leq P_a \) then
27: \( \text{nest}[n] \leftarrow \text{Lévy flights} \)
28: \( \text{Quality(\text{nest}[n])} \leftarrow \text{Evaluate \text{nest}[n] using eq. 2} \)
29: end if
30: end while
31: Rank the nests and find the best nest.
32: if \( \text{Quality(best nest)} \geq \text{bestNest} \) then
33: bestNest \( \leftarrow \) best nest
34: end if
35: end while
36: return bestNest

IV. ENHANCED BINARY CUCKOO SEARCH (EBCS) WITH FREQUENT VALUES AND ROUGH SET THEORY FOR FEATURE SELECTION
This section describes the proposed approach by enhancing the binary cuckoo search algorithm with a new objective function. EBCS improves the binary cuckoo search algorithm by developing a new initialization mechanism, a new updating mechanism, and a new objective function. The new objective function aims to find the minimum features subset without significant reduction in classification performance for nominal, mixed, and numerical datasets with varying number of features. Subsection IV-A describes the newly developed objective function and subsections IV-B and IV-C describe the enhanced BCS.

A. NEW OBJECTIVE FUNCTION
In general, using a features subset that has high relevancy to class labels and high frequent values in its features increases classification performance in many classification algorithms [80], [81]. High frequent values, which are used to build the future classification model, correspond to a high reuse probability. The high frequent values alone, however,
are not a sufficient indicator for classification performance, because the dependency between the features subset and class labels may be low. RSTDD provides an efficient measure to determine the dependency between the features subset and class labels. However, it is not sufficient alone to provide a scalable indicator for classification performance in all types of datasets, especially in datasets that have low and different frequent values (e.g., mixed and numerical datasets). Therefore, it is necessary to develop a new more efficient objective function for nominal, mixed and numerical datasets by balancing between frequent values, dependency degree, and number of selected features.

1) FREQUENT VALUES
When a feature has high frequent values, it is an indicator that the number of distinct values in this feature is low. We use the number of distinct values and the total number of objects to calculate the percentage of distinct values in each feature to provide an indicator for frequent values. See equation 7. This equation provides low percentage when frequent values are high.

\[
\text{Distinct}(R)\% = \frac{|\text{DistinctValues}(R)|}{|\text{Objects}|} \quad (7)
\]

where \( R \) is a subset of features, \( \text{DistinctValues}(R) \) is the average of number of distinct values for \( R \) features, and \( \text{Objects} \) is the number of total objects in dataset.

2) DEPENDENCY DEGREE
We use RSTDD (equation 1) to measure the dependency between the feature subset and class labels.

3) BALANCING BETWEEN THE PERCENTAGE OF DISTINCT VALUES AND DEPENDENCY DEGREE
In general, high dependency degree is a good indicator to measure relevancy between the feature subset and class labels. Low distinct percentage for the feature subset is a good indicator for high frequent values. In other words, a feature subset, which has maximum dependency degree and minimum distinct values percentage, is desirable for classification algorithms; dependency degree is close to one (\(~100\%) and percentage of distinct values is close to zero. Equation 8 balances between the two; it provides high quality when the feature subset is more desirable for classification algorithms.

\[
\text{Quality}(R)\% = \gamma_R(Q)\% - \text{DistinctValues}(R)\% \quad (8)
\]

where \( R \) is a subset of features, and \( \gamma_R(Q) \) is the dependency between feature subset \( R \) and classes \( Q \).

4) BALANCING BETWEEN THE QUALITY AND NUMBER OF SELECTED FEATURES
FS is a multi-objective problem where maximum classification performance and minimum number of selected features are goals of FS [81]. Equation 8 achieves the first objective only. To achieve the two objectives, equation 8 is modified to equation 9.

\[
\text{FinalQuality}(R)\% = \text{Quality}(R) \times \frac{|C| - |R|}{|C|} \quad (9)
\]

where Quality (\( R \)) is calculated from equation 8, \( C \) is the number of available features (total features), \( R \) is the number of selected features. Equation 9 uses the factor \((|C|-|R|)/|C|\) in addition to the quality of feature subset to achieve the two objectives. In other words, equation 9 multiplies equation 8 by the above factor; to guide the EBCS toward the feature subset that achieves the minimum number of selected features and maximum classification performance.

B. NEW INITIALIZATION MECHANISM
In BCS [27], the initial solution (or nest) is randomly initialized as shown in algorithm 2.

Algorithm 2 Initialization Mechanism for Basic BCS.

\[
\begin{align*}
1: & \text{for } \text{<nest}=0 \text{ to population Size }> \text{do} \\
2: & \text{for } \text{<egg}=0 \text{ to total feature -1 }> \text{do} \\
3: & \text{<Solution}[\text{<nest}][\text{<egg}]=\text{random}(0,1)> \\
4: & \text{end for} \\
5: & \text{end for}
\end{align*}
\]

Given its binary selection, from a search space, the probability of being selected is 50% [82]. A good initialization mechanism is one capable of generating initial solutions with many different number of features covering as much of the search space as possible.

For example, assume the total number of features in a search space is 50. The possible number of features to select for the search ideally could be from 0 to 50. A good initialization strategy is one that generates solutions uniformly between 0 and 50. However, algorithm 2 does not guarantee uniform generation across the search space.

To understand the reason behind this, consider the probability theory of algorithm 2. The number of successes, in a sequence of \( n \) (total number of features) independently selected or not selected experiments, is the binomial distribution [82] where each success (probability \( p = 50\%) \) and the number of features \( n \) play a major role in determining the probability of numbers of successes. The probability of getting exactly \( k \) successes in \( n \) trials is given by the probability mass function. Subfigures 1 (A-D) shows examples of the probability mass function for different values of \( n \). The result of the binomial distribution is that the probability for selecting the features subsets near the end points decreases with increasing \( n \). For large \( n \), the probability is near 0 in the feature selection regions <25% and >75% of the total number of features.

However, this strategy does not help BCS cover most of the search space (i.e., most of the possible numbers of selected features), and thus causes slow and weak convergence. Some approaches used different initialization methods to improve the initialization strategy [83], [84]. However, to improve the initialization strategy in BCS for FS, a new mechanism that divides the initialization space equally to three parts is
FIGURE 1. Probability of getting exactly number of selected features in different n: Total number of features (probability mass function).

FIGURE 2. Probability mass function of new initialization mechanism.

developed. See figure 2 and the algorithm 3 (Lines 7-34). This is described below in more details.

Small Part: aims to generate feature subsets that have number of the selection of around the 25% of the available features. This helps to find optimal solutions that have small size numbers. Small initialization consists of three steps: first, it starts from an empty set. Second, select random number “s” between one and half number of the available features. Third, select randomly “s” features from all available features, then add them to the empty set. Thus according to binomial distribution, the possible numbers of selected features include around quarter of available features, which would then have greater chance of selection than others. See algorithm 3 (Lines 7-14).

Medium Part: aims to reach and search the area of the feature subsets with medium size numbers. This helps to find the optimal solutions within the medium size range numbers. This mechanism also starts from an empty set. Then selects randomly features from all available features. It then adds selected features to the empty set. This initialization focuses on the feature subsets that have around half the number of available features. This part is the same as the traditional BCS initialization strategy. Figure 2 shows the possible numbers of selected features that have around half the number of available features with greater chance being selected than others. See algorithm 3 (Line 15-22).

Large Part: This mechanism is capable of searching for the feature subsets that are selected close to the full number of available features. This helps to find the optimal solutions
Algorithm 3 EBCS

1: Inputs:
2: \( N \leftarrow 20; \) \{Number of nests/solutions\}
3: \( F \leftarrow \text{Number of eggs/Features} \)
4: \( P_a \leftarrow 0.5; \) \{Probability of \( P_a \in [0,1]\)\}
5: \( T \leftarrow 20; \) \{Maximum number of iterations\}
6: Output: \( \text{bestNest} \leftarrow \{\} \) \{Best global feature subset\}

7: \( n \leftarrow 1. \) \( \triangleright \) Initial Population
8: \( \text{while } n \leq 0.33N \text{ do} \)
9: \( \text{egg} \leftarrow 0. \)
10: \( \text{while } \text{egg} \leq F/2 \text{ do} \)
11: \( \text{nest}[n][\text{random}(1-F)] \leftarrow \text{random}(1.0) \)
12: \( \text{end while} \)
13: \( \text{Evaluate nest}[n] \) using eq. 9
14: \( \text{end while} \)

15: \( n \leftarrow 0.33N. \) \( \triangleright \) Small Part
16: \( \text{while } n \leq 0.66N \text{ do} \)
17: \( \text{egg} \leftarrow 0. \)
18: \( \text{while } \text{egg} \leq F \text{ do} \)
19: \( \text{nest}[n][\text{random}(1-F)] \leftarrow \text{random}(1.0) \)
20: \( \text{end while} \)
21: \( \text{Evaluate nest}[n] \) using eq. 9
22: \( \text{end while} \)

23: \( n \leftarrow 0.66N. \) \( \triangleright \) Medium Part
24: \( \text{while } n \leq N \text{ do} \)
25: \( \text{nest}[n] \leftarrow \text{Full Set.} \)
26: \( r \leftarrow \text{random}(1 - F/2). \) \( \triangleright \) \( r \) is the number of removed features
27: \( \text{egg} \leftarrow 0. \)
28: \( \text{while } \text{egg} \leq r \text{ do} \)
29: \( \text{nest}[n][\text{random}(1-F)] \leftarrow 0 \)
30: \( \text{end while} \)
31: \( \text{Evaluate nest}[n] \) using eq. 9
32: \( \text{end while} \)
33: \( \text{Rank the nests based on their quality and find the best nest.} \)
34: \( \text{bestNest} \leftarrow \text{best nest.} \)

35: \( \text{while } r \leq T \text{ do} \)
36: \( n \leftarrow 1. \) \( \triangleright \) Updating
37: \( \text{while } n \leq 0.33N \text{ do} \)
38: \( \text{egg} \leftarrow 0. \)
39: \( \text{while } \text{egg} \leq F \text{ do} \)
40: \( \text{nest}[n][\text{random}(1-F)] \leftarrow \text{random}(1.0) \)
41: \( \text{end while} \)
42: \( \text{Evaluate nest}[n] \) using eq. 9
43: \( \text{end while} \)

44: \( n \leftarrow 0.33N. \) \( \triangleright \) Large Part
45: \( \text{while } n \leq 0.66N \text{ do} \)
46: \( \text{egg} \leftarrow 0. \)
47: \( \text{while } \text{egg} \leq F \text{ do} \)
48: \( \text{nest}[n][\text{random}(1-F)] \leftarrow \text{random}(1.0) \)
49: \( \text{end while} \)
50: \( \text{Evaluate nest}[n] \) using eq. 9
51: \( \text{end while} \)
52: \( n \leftarrow 0.66N. \)
53: \( \text{while } n \leq N \text{ do} \)
54: \( \text{nest}[n] \leftarrow \text{Full Set.} \)
55: \( r \leftarrow \text{random}(1 - F/2). \) \( \triangleright \) \( r \) is the number of removed features
56: \( \text{egg} \leftarrow 0. \)
57: \( \text{while } \text{egg} \leq r \text{ do} \)
58: \( \text{nest}[n][\text{random}(1-F)] \leftarrow 0 \)
59: \( \text{end while} \)
60: \( \text{Evaluate nest}[n] \) using eq. 9
61: \( \text{end while} \)

62: \( \text{Set } n \leftarrow 1. \) \( \triangleright \) Local Search
63: \( \text{while } n \leq N \text{ do} \)
64: \( \text{if } \left( \text{Quality(nest}[n]) \leq P_a \right) \text{ then} \)
65: \( \text{nest}[n] \leftarrow \text{Lévy flights} \)
66: \( \text{Evaluate nest}[n] \) using eq. 9
67: \( \text{end if} \)
68: \( \text{end while} \)
69: \( \text{Rank the nests and find the best nest.} \)
70: \( \text{if } \left( \text{Quality(best nest)} \geq \text{bestNest} \right) \text{ then} \)
71: \( \text{bestNest} \leftarrow \text{best nest} \)
72: \( \text{end if} \)
73: \( \text{end while} \)
74: \( \text{return bestNest} \)

that have around 75% of the number of available features. Large initialization consists of three steps: First, it starts from a full set. Second, it selects a random number “s” between the one and half the number of available features. Third, it removes these features from the full set. See algorithm 3 (Lines 23-34).

C. NEW GLOBAL SEARCH MECHANISM

The goal of the global search is to cover as much of the search space as possible to guarantee global convergence and low computational time. Because the global search in BCS uses the traditional initialization mechanism, it is not guaranteed to converge when the search space contains more than 20 features. BCS’s global search is also developed using the same strategy as was introduced for the new initialization mechanism to cover as much of the search space as possible. This is achieved by using algorithm 3, lines 35-61.

V. EVALUATION METHODOLOGY

This section describes the followed evaluation methodology and the datasets used in the experiments.

A. DATASETS

In order to evaluate the performance of EBCS, a group of experiments were run on sixteen datasets, which were selected from the University of California at Irvine, known as the UCI data repository of machine learning database [85]. UCI divides classification datasets according to their types of features to three groups: firstly, the “nominal” group, which contains 28 datasets; secondly, the “Numerical” group, which contains 137 datasets; and thirdly, or the “mixed” group, which includes 37 datasets.

Our approach aims to achieve FS for nominal, mixed, and numerical datasets with different characteristics, including different number of features, different number of objects, and different number of class values. To evaluate this approach, sixteen datasets that possess these characteristics are selected randomly as follows: Four datasets from the “nominal” group, four datasets from the “mixed” group, and eight datasets from the “numerical” group. Table 3 shows these datasets and their characteristics.

A common way to conduct the experiments on FS for classification is to divide each dataset into two sub-datasets randomly: a training set, and a learning and test set. A training set has around 70% of the dataset’s objects, and 30% of the dataset’s objects are for the learning and test set [84], [86]. The training set is used by FS approaches to achieve features reduction, and the learning and test set is used to build the classification model and estimates the performance of classification.

In this work, K-fold cross-validation [87] was used to split the learning and test set into two disjoint sets to build the classification model and estimate the classification performance. When an object belongs to the test set, its class is hidden from
In particular, K-fold cross-validation splits the learning and test set into K subsets, then the learning set is created on K-1 subsets, and the test set is created on the remaining subset. The process is repeated with several partitions to calculate the classification performance.

**B. EVALUATION METHOD**

In order to evaluate the performance of EBCS, the paper takes an indirect approach [12]. Two main comparisons are commonly used in the indirect approach to evaluate the FS. Firstly, before and after comparison, which measures the classification performance of all available features and for selected features. Secondly, comparison checks the efficiency of a specific FS approach by comparing it to other FS approaches after applying them on the selected 16 datasets. Our developed approach or algorithm (EBCS) is compared with the baseline approach, i.e. Basic Binary Cukcoo Search (FS-BCS) algorithm III-B2, which uses basic binary cuckoo search. The binary cuckoo search algorithm for the search strategy and objective function (equation 2) are taken from the literature [38]. In addition, it will be compared to ten FS approaches as follows: In section VI-E, EBCS is compared to the experiments that were published in [68] over five datasets, which were taken from UCI [88], and these experiments have a comparison between five NIA (HS,GA, PSO, HC and MMRM-IWD) with fuzzy RST approaches. Comparison between EBCS, genetic algorithm [89] with CFS [90], multi objective evolutionary [91] with CFS [90] and PSO [92] with CFS [90] are discussed in section VI-F. The experimental results for comparison between EBCS, best first with CFS [90], [93] and linear forward selection with Consistency Subset Evaluator (CON) [41], [94] are included in the section VI-G. Two different classification types (DT [95] and NB [96]) are used to measure the classification performance.

**C. BENCHMARKING AND EXPERIMENT DESIGN**

All experiments are run on a personal computer running Windows 10 with (i7) 3 GHz processor and 16 GB RAM. EBCS and FS-BCS are implemented using PHP. Other approaches in the experiments have standard implementations in the Weka tool [97]. The standard Weka tool implementation of K-fold cross-validation is also used [97].

The parameters’ settings are identified based on default settings in the Weka tool, the original BCS and test experiments. For the Weka tool, default values were used for the parameters except for population size (=20), maximum number of iterations (=20), and the size of k (=10 objects) in the K-fold cross-validation.

Test experiments were run initially to find optimal values. The authors found that running the experiments more than five times did not improve the results, which were sufficient to obtain consistent results between the EBCS and FS-BCS. Running the experiments more than five times did not improve this comparison, and thus results for more than five times are not reported. The number of selected features (features subset) and computational time are recorded for each run. In the next step, the run that achieved best classification accuracy is selected. Similarly, PSO and genetic approaches are run for each training set until good FS is achieved, where the number of selected features are recorded for each run.

DT “J48” [95] and NB [96] classification algorithms, which are implemented within Weka tool, are used to measure the classification performance for all approaches that are used in our experiments. They are applied on each reduced learning set to build the classification model, which are then applied on the test set to measure the classification performance. Also classification performance was measured for all datasets before and after FS. NB and DT are selected because they are commonly used algorithms to evaluate FS, which are considered two of the top ten data mining
TABLE 4. Results of FS-BCS and EBCS. (All: Original datasets. Size: Number of features. SR: % of size reduction against all features. Acc: Accuracy. Prec: Precision average. Rec: Recall average. Difference accuracy (DA). The different between all features and the features subset accuracies. Highlight corresponds to the best approach based on balancing between classification accuracy and number of selected features.

| Dataset                     | Approach | Size | SR% | Decision Tree | Naive Bayes |
|-----------------------------|----------|------|-----|---------------|-------------|
|                             |          |      |     | % Acc | Prec | Rec | DA | % Acc | Prec | Rec | DA |
| Congressional Voting Records| All      | 16   | 62.50 | 96.7 | 96.8 | 96.8 | -1.32 | 90.1 | 90.3 | 90.1 |
|                            | EBCS     | 6    | 62.50 | 93.58 | 95.6 | 95.4 | -1.32 | 94.6 | 94.6 | 94.6 |
|                            | FS-BCS   | 6    | 62.50 | 93.58 | 95.6 | 95.4 | -1.32 | 94.6 | 94.6 | 94.6 |
| Soybean (small)             | All      | 35   | 88.57 | 95.7 | 96.5 | 95.7 | -0.10 | 100.0 | 100.0 | 100.0 |
|                            | EBCS     | 4    | 71.43 | 85.7 | 74.6 | 85.7 | -10   | 92.8 | 94   | 92.9 |
|                            | FS-BCS   | 6    | 72.73 | 99.7 | 99.8 | 99.8 | -0.3  | 98.1 | 98.3 | 98.2 |
| Mushroom                    | All      | 22   | 81.82 | 98.4 | 98.5 | 98.4 | -1.6  | 98.3 | 98.4 | 98.3 |
|                            | EBCS     | 6    | 72.73 | 99.7 | 99.8 | 99.8 | -0.3  | 98.1 | 98.3 | 98.2 |
| Lung Cancer                 | All      | 56   | 83.93 | 66.6 | 75.6 | 66.7 | 16.6  | 66.6 | 66.7 | 66.7 |
|                            | EBCS     | 7    | 67.86 | 37.5 | 31.3 | 37.5 | -12.5 | 62.5 | 39.1 | 62.5 |

VI. RESULTS AND DISCUSSIONS

This section presents and discusses the results of EBCS compared to the baseline FS-BCS approach, ten known filter FS approaches, and all available features (before FS). Differences in accuracy is considered significant when it is more than 5% [99], and accuracy of different methods is considered equal when the difference is less than 1% [84]. In all results, no significant difference was noted between average of precision and average of recall, thus accuracy was sufficient to evaluate the classification performance for the datasets used in the experiments. In addition, statistical algorithms that do not require complex initial parameters settings [12], [98].
significance of classification accuracy was tested using the Sign test method [100] on the results of the evaluated 10 approaches. The results were found statistically significant for the evaluated approaches, for which the probability, p-value, was found less than or equal to 0.05.

A. COMPARISONS BETWEEN EBCS AND FS-BCS
Experimental results of EBCS and FS-BCS for the sixteen datasets are shown in Table 4. The following discusses the results for each type of the datasets.

1) NOMINAL DATASETS
Table 4 and subfigures 3 (A,B) show that EBCS and FS-BCS achieve the same size reduction, DT accuracy, and NB accuracy for the congressional voting records dataset. For the Mushroom, Soybean, and Lung Cancer datasets, both EBCS and FS-BCS achieve better or roughly same DT accuracy and NB accuracy, but EBCS achieves significant size reduction compared to FS-BCS. These results show that both the developed objective function and traditional objective functions [50] have roughly the same efficiency.
for nominal datasets. However, EBCS has better convergence for datasets that have more than 20 features and when size reduction is less than one quarter of available features. See section IV-B.

2) MIXED DATASETS

Table 4 and subfigures 3 (C,D) show our approach (EBCS) achieves better size reduction in all mixed datasets compared to FS-BCS. For the Zoo dataset, EBCS achieves less size reduction and improves classification accuracy (DT accuracy = 13%, NB accuracy = 26.7%). For the Hepatitis and German Credit Data, EBCS achieves the same size reduction and significantly better classification accuracy (both DT and NB) than FS-BCS. However, for the Dermatology dataset, EBCS achieves better size reduction and classification accuracy than FS-BCS. These results show that the developed objective function performs more efficiently than the standard objective function for mixed datasets where the number of distinct feature values is different. Similarly, it shows EBCS is more efficient than FS-BCS when the number of features in datasets (e.g. Dermatology) is more than 20 features.

3) NUMERICAL DATASETS

According to table 4 and subfigures 3 (E,F), EBCS and FS-BCS achieve the same FS for the Breast Cancer Wisconsin (Original) dataset, because the number of available features is 9 (low dimensional dataset) and the number of distinct values of its features is the same. For the Wine dataset, EBCS achieves less size reduction, but it achieves significant improvement of classification accuracy (DT and NB) compared to FS-BCS. This improvement is caused by the newly developed objective function, which is more efficient than the standard objective function for this dataset, given its features have different number of distinct values. For the ISOLET dataset, FS-BCS failed to achieve the FS; it selects around half the number of available features compared to EBCS, which selects 8% from the available features without significant reduction in classification accuracy. This is caused by the high number of distinct values (nearly equal to the total number of objects) in this dataset, and the weak convergence of the FS-BCS. The high number of distinct values and the number of selected features causes RSTDD to produce full dependency (100%) for feature subsets, which subsequently causes FS-BCS’s convergence to use local search only to update the population of solutions. This indicates that FS-BCS is trapped in a local optimal.

For the remaining datasets, EBCS achieves better significant size reduction than FS-BCS (from 15.7% to 38.1%), with significant improvement in classification accuracy (except for the Sonar dataset where there is no significant reduction of NB classification accuracy). EBCS achieves higher efficient convergence than FS-BCS, especially when the number of available features for these datasets are more than 20. Also, the developed objective function is more efficient than the standard objective function when the datasets have different number of distinct values in their features.

**B. ANALYSIS OF COMPUTATIONAL TIME AND NUMBER OF ITERATIONS**

Table 5 shows the computational time, in seconds, of FS-BCS and EBCS for the 16 studied datasets. As expected, most of their computational time was spent in the objective function process. The objective function is run 20 times (population size) for each iteration. The time complexity for each iteration is thus $O(p\cdot f + p\cdot n\log n)$, where $p$ is the population size, $f$ is the number of features and $n$ is the number of objects. In the experiments, $p$ is constant (20) but $f$ and $n$ vary up to 617 and 8124 respectively. The effect of the $f$ factor is linear on the time complexity while the time complexity of input $n$ is logarithmic. This means, the time complexity is the same for FS-BCS and EBCS, although EBCS took less time than FS-BCS in fourteen datasets, see figure 4. The main reason for this difference is that EBCS runs fewer number of iterations to reach the best feature subset, converging much faster to the optimal subset than FS-BCS. Unfortunately, the overall time complexity of both algorithms cannot be determined theoretically because they are non-deterministic algorithms that produce different outputs in different runs with different number of iterations for the same input. Therefore, the paper measures computational time for both algorithms or approaches empirically, please see [101] for details.

The Isolet-test dataset has numerical features that have very high number of distinct feature values (low frequent values) and FS-BCS selects nearly half the available features. This implies that the value of the standard objective function is 100%, which means FS-BCS’s convergence uses local update only to generate new candidate feature subsets after the first iteration (i.e., FS-BCS is trapped in a local optimal).

**TABLE 5. Computational Time of FS-BCS and EBCS and number of iterations. (Time(S): Time, in seconds, needed for number of iterations to find the best feature subset. Itr.#: Number of iterations needed to find the solution).**

| Dataset          | FS-BCS Itr.# | FS-BCS Time(S) | EBCS Itr.# | EBCS Time(S) |
|------------------|-------------|----------------|------------|--------------|
| Breast Cancer    | 7           | 1.68           | 4          | 1.40         |
| Wine             | 10          | 0.64           | 6          | 0.55         |
| Hepatitis        | 12          | 0.75           | 6          | 0.44         |
| Voting           | 4           | 0.62           | 2          | 0.427        |
| Zoo              | 13          | 0.532          | 4          | 0.240        |
| Segment          | 18          | 12.78          | 4          | 5.010        |
| Credit           | 7           | 3.31           | 1          | 1.644        |
| Mushroom         | 12          | 50.46          | 3          | 19.73        |
| Dermatology      | 3           | 1.355          | 2          | 1.496        |
| Soybean (small)  | 8           | 0.403          | 4          | 0.271        |
| Spectf           | 13          | 1.216          | 7          | 0.824        |
| Lung Cancer      | 5           | 0.287          | 1          | 0.184        |
| Sonar            | 19          | 2.93           | 3          | 0.894        |
| Movement         | 9           | 3.793          | 3          | 2.033        |
| Musk (Version 1) | 10          | 6.013          | 5          | 3.0541       |
| ISOLET-test      | 1*          | 16.091         | 3          | 55.398*      |

* FS-BCS is trapped in a local optimal

Table 5 and figure 4 (A) show the number of iterations needed to find the best features subset in both approaches. EBCS needs 56% iterations from FS-BCS iterations to find the best features subset in these datasets. Figure 4(B)
shows the time difference in percentage for convergence for both approaches. EBCS converges in less time compared to FS-BCS in 14 datasets, because EBCS runs significantly fewer iterations to converge in the search space for most datasets, except for the Dermatology dataset where the difference in iterations is minor (and thus its computational time). On the other hand, for the Isolet-test dataset FS-BCS failed to achieve feature reduction.

To explain the effect of varying number of objects (n) and number of features toward computational time of EBCS and FS-BCS. Both algorithms are run for Musk (version 1) dataset which has good number of features (f) and objects (n) multiple
times to find five feasible feature subsets. Best time, worst time and average time for the five feasible feature subsets are recorded.

**TABLE 6. EBCS and FS-BCS computational time for musk (version 1) dataset with varying number of objects.**

| # of Objects | EBCS time (second) | FS-BCS time (second) |
|--------------|--------------------|---------------------|
|              | Best | Average | Worst | Best | Average | Worst |
| 26           | 0.08 | 0.11    | 0.16  | 0.17 | 0.19    | 0.22  |
| 76           | 0.18 | 0.24    | 0.33  | 0.50 | 0.52    | 0.59  |
| 126          | 0.29 | 0.40    | 0.59  | 0.90 | 0.94    | 1.00  |
| 176          | 0.47 | 0.61    | 0.75  | 1.43 | 1.46    | 1.48  |
| 226          | 0.67 | 0.82    | 1.18  | 1.93 | 2.04    | 2.21  |
| 276          | 0.91 | 1.07    | 1.39  | 2.31 | 2.67    | 3.01  |
| 326          | 1.13 | 1.34    | 1.66  | 3.13 | 3.36    | 3.59  |
| 376          | 1.45 | 1.67    | 2.04  | 3.88 | 4.07    | 4.13  |
| 426          | 1.78 | 2.08    | 2.55  | 4.69 | 4.85    | 5.24  |
| 476          | 2.23 | 2.55    | 3.05  | 5.65 | 5.86    | 6.01  |

**TABLE 7. EBCS and FS-BCS computational time for musk (version 1) dataset with varying number of features.**

| # of Features | EBCS time (second) | FS-BCS time (second) |
|---------------|--------------------|---------------------|
|               | Best | Average | Worst | Best | Average | Worst |
| 18            | 0.16 | 0.39    | 0.45  | 1.11 | 1.12    | 1.14  |
| 48            | 0.62 | 0.78    | 0.90  | 1.88 | 1.98    | 2.02  |
| 78            | 0.96 | 1.17    | 1.40  | 2.85 | 2.94    | 3.01  |
| 108           | 1.46 | 1.70    | 2.01  | 3.83 | 3.93    | 4.04  |
| 138           | 1.84 | 2.10    | 2.47  | 4.72 | 4.80    | 4.85  |
| 168           | 2.23 | 2.55    | 3.05  | 5.65 | 5.86    | 6.01  |

**C. CLASSIFICATION PERFORMANCE BEFORE AND AFTER EBCS**

According to table 4 and figure 7, EBCS achieves significant size reduction (average 79.4%) without significantly reducing or affecting the classification accuracy for all datasets using NB and DT (except Soybean due to its small size).

**D. ANALYSIS OF NEW OBJECTIVE FUNCTION**

To evaluate the efficiency of the newly developed objective function, the standard objective function [50] is used in EBCS instead, named TEBCS to differentiate it. It is implemented using PHP and is run five times for each training set. Classification performance is then measured using NB and DT. Results of EBCS and TEBCS are shown in table 8.
According to table 8 and subfigures 8 (A,B), the new objective function and the standard objective function [50] have the same efficiency when they are applied on nominal datasets that have roughly the same number of distinct feature values.

As shown, EBCS accuracy and TEBCS accuracy have the same values and similarly EBCS SR% and TEBCS SR%, thus two values (EBCS accuracy, EBCS SR%) are visible in Figure 8.

Results in table 8 and subfigures 9 (A,B) shows that the new objective function is more efficient than the standard objective function for mixed and numerical datasets. The new objective function enables the EBCS to achieve size reduction without significant reduction in classification accuracy. On the other hand, the standard objective function enables EBCS to achieve significant size reduction but at the cost of significant reduction in classification accuracy. However for nominal datasets, e.g. the Breast Cancer Wisconsin, that have the same number of distinct feature values, both objective functions show same efficiency. However, as shown, the newly developed objective function is more efficient for mixed and numerical datasets that have different number of distinct feature values.

E. COMPARISON BETWEEN EBCS AND HYBRID NIA WITH FUZZY RST APPROACHES

This section measures the efficiency of EBCS approach by comparing it to five hybrid NIA and fuzzy RST approaches using the experimental results in Master River Multiple Creeks Intelligent Water Drops (MRMC-IWD) [66]. The experimental study compared MRMC-IWD to four state-of-the-art approaches (HS, GA, PSO, and HC) that were published in [68] over five datasets, which were taken from UCI [88]. Table 9 summarizes the main characteristics of the datasets.

The results in table 10 and subfigures 10 (A,B) show EBCS achieves significant improvement of classification accuracy before and after EBCS.

FIGURE 7. Classification accuracy before and after EBCS.

FIGURE 8. EBCS vs. TEBCS for nominal datasets (classification accuracy, SR%).
TABLE 8. Results of EBCS with new objective function and EBCS with traditional objective function [50]. (All: Original datasets. Size: Number of features. Acc: Accuracy. Prec: Precision average. Rec: Recall average.). Highlight corresponds to the best approach based on balancing between classification accuracy and number of selected features.

| Dataset               | Approach | Size | Decision Tree % | Naive Bayes % |
|-----------------------|----------|------|-----------------|--------------|
|                       |          |      | Acc. | Prec. | Rec. | Acc. | Prec. | Rec. |
| **Nominal Datasets**  |          |      |      |       |      |      |       |      |
| Congressional Voting Records | All 16 |   96.7 | 96.8 | 96.8 | 90.1 | 90.5 | 90.1 |
|                       | EBCS 6  | **95.38** | **95.6** | **95.4** | **94.6** | **94.6** | **94.6** |
|                       | TEBCS 6 | 94.6 | 94.8 | 94.6 | 93.8 | 94.1 | 93.8 |
| Soybean (small)       | All 35  | 95.7 | 96.5 | 95.7 | 97.8 | 98.1 | 97.9 |
|                       | EBCS 4  | 85.7 | 74.6 | 85.7 | 100 | 100 | 100 |
|                       | TEBCS 4 | 85.7 | 75.5 | 85.7 | 100 | 100 | 100 |
| Mushroom              | All 22  | 98.4 | 98.5 | 98.4 | **98.3** | **98.4** | **98.3** |
|                       | EBCS 4  | **99.42** | **99.4** | **99.4** | 97.9 | 97.9 | 97.9 |
|                       | TEBCS 4 | 98.4 | 98.5 | 98.4 | 97.9 | 97.9 | 97.9 |
| Lung Cancer           | All 56  | 50  | 50  | 50  | 56.2 | 55.3 | 56.3 |
|                       | EBCS 7  | 66.6 | 75.6 | 66.7 | 66.6 | 66.7 | 66.7 |
|                       | TEBCS 7 | 66.6 | 75.6 | 66.7 | 66.6 | 66.7 | 66.7 |
| **Mixed Datasets**    |          |      |      |       |      |      |       |      |
| Zoo                   | All 17  | 93  | 95  | 93  | 94  | 94.6 | 94.1 |
|                       | EBCS 7  | 93  | 95  | 93  | 96.7 | 97.8 | 96.7 |
|                       | TEBCS 4 | 66.6 | 47.7 | 66.7 | 73.3 | 58.2 | 73.3 |
| Hepatitis             | All 19  | 80  | 80.2 | 80 | 83.2 | 84.1 | 83.2 |
|                       | EBCS 4  | 84.7 | 86.9 | 86.4 | 86.9 | 86.6 | 86.9 |
|                       | TEBCS 4 | 84.4 | 86.6 | 87 | 80.4 | 79.8 | 80.4 |
| German Credit Data    | All 20  | 71.2 | 69.4 | 71.1 | 35.1 | 74.7 | 35.1 |
|                       | EBCS 6  | 68  | **67.3** | **68** | 71.3 | 69.8 | 71.3 |
|                       | TEBCS 5 | 63.6 | 58.3 | 63.7 | 66.7 | 63 | 66.7 |
| Dermatology           | All 34  | 95.9 | 96  | 95.9 | 97.5 | 97.7 | 97.5 |
|                       | EBCS 10 | 93.6 | **94.7** | **93.6** | **98.1** | **98.4** | **98.2** |
|                       | TEBCS 8 | 86.3 | 82.4 | 86.4 | 86.4 | 83.8 | 86.4 |
| **Numerical Datasets**|          |      |      |       |      |      |       |      |
| Breast Cancer         | All 9   | 93.9 | 93  | 93  | 96.1 | 96.3 | 96.1 |
| Wisconsin (Original)  | EBCS 3  | 97.6 | 97.6 | 97.6 | 98.1 | 98.1 | 98.1 |
|                       | TEBCS 3 | 97.6 | 97.6 | 97.6 | 98.1 | 98.1 | 98.1 |
| Wine                  | All 13  | **91.5** | **91.6** | **91.6** | **97.7** | **97.8** | **97.8** |
|                       | EBCS 3  | 87.3 | 87.3 | 87.1 | 94.4 | 94.9 | 94.5 |
|                       | TEBCS 2 | 75.4 | 74.9 | 75.9 | 83 | 82.7 | 83 |
| Segment               | All 19  | 96.4 | 96.4 | 96.4 | 81.2 | 83.3 | 81.3 |
|                       | EBCS 3  | **92.65** | **92.9** | **92.7** | **83.1** | **82.9** | **83** |
|                       | TEBCS 2 | 82.1 | 82.1 | 82.1 | 71.2 | 70.8 | 71.2 |
| Spectf                | All 44  | 66.25 | 67.1 | 66.3 | 80 | 82 | 80 |
|                       | EBCS 6  | 75 | 75.7 | 75 | **83.33** | **84.3** | **83.3** |
|                       | TEBCS 2 | 50 | 25 | 50 | 54.1 | 54.2 | 54.2 |
| Connectionist         | All 60  | 70.2 | 70.2 | 70.2 | 67.3 | 67.3 | 66.9 |
| Bench (Sonar)         | EBCS 13 | 79 | **78.9** | **79** | 67.7 | 67.4 | 67.7 |
|                       | TEBCS 2 | 67.7 | 67.2 | 67.7 | 67.7 | 67.7 | 67.7 |
| Libras Movement       | All 90  | 64.4 | 65.8 | 64.4 | 62.2 | 63.6 | 62.2 |
|                       | EBCS 7  | **60.5** | **60.6** | **60.6** | **62.9** | **63.3** | **63** |
|                       | TEBCS 5 | 36.1 | 36.9 | 36.1 | 44.4 | 43 | 44.4 |
| Musk (Version 1)      | All 168 | 100 | 100 | 100 | 76.8 | 77.5 | 76.9 |
|                       | EBCS 7  | **96.2** | **97.7** | **96.2** | **84.6** | **85.3** | **84.6** |
|                       | TEBCS 4 | 63.6 | 63.8 | 63.6 | 67.8 | 68.3 | 67.8 |
| ISOLT-test            | All 617 | 78 | 78.3 | 78.1 | 83.7 | 84.8 | 83.8 |
|                       | EBCS 51 | **76.2** | **76.4** | **75.9** | **80.3** | **81.4** | **80.4** |
|                       | TEBCS 17 | 32.2 | 33.4 | 32.3 | 49.7 | 50 | 49.8 |

Accuracy with best or same size reduction compared to other approaches on Ionosphere and Water datasets. However, in the Sonar dataset, all approaches minimize the number of selected features more than EBCS, in comparison to others, EBCS improves the classification accuracy significantly. In Libras dataset, EBCS and HS achieve the same and best
results compared to others while GA maximizes the classification accuracy slightly (less than 5%) and number of selected features significantly (more than 100%) compared to EBCS. Finally in Arrhythmia, EBCS achieves the best size reduction and highest classification accuracy compared to PSO, GA, HS, and MRMC-IWD approaches, and the roughly same classification accuracy and size reduction compared to HC approach.

In general, EBCS achieves the best FS compared to PSO, GA, and MRMC-IWD on five datasets. While compared to HC and HS approaches, EBCS improves the FS on four datasets and the same or roughly the same on one dataset.

### F. COMPARISON BETWEEN EBCS, MULTI OBJECTIVES EVOLUTIONARY, PSO WITH CFS, AND GENETIC WITH CFS

In general, subfigure 11 (A) shows that EBCS achieves better size reduction and classification accuracy (in DT and NB) compared to both PSO, multi-objectives evolutionary, and genetic. EBCS removes nearly 79% from all features and improves classification accuracy (DT and NB). On the other hand, PSO removes only 59.8% from all features with significant reduction of DT classification accuracy (-8.1%) and of NB classification accuracy (-7.2%).

Similarly, Genetic removes only 58.8% from all features with significant reduction of DT classification accuracy (-5.4%) and of NB classification accuracy (-5.3%). Also multi objectives evolutionary removes 66% from original data with reduction of DT classification accuracy (-4.4%) and of NB classification accuracy (-2.9%).

Table 11, subfigure 12 (A,B) and figure 11 (B) show EBCS achieves best size reduction while improving the...
FIGURE 10. EBCS vs. hybrid NIA and fuzzy RST.

TABLE 11. Results of EBCS, genetic with CFS, multi objectives evolutionary with CFS and PSO with CFS. (Size: Number of selected features. DT: Classification accuracy based on decision. NB: Classification accuracy based on Naive Bayes. Highlight corresponds to the best approach/s based on balancing between classification accuracy and number of selected features.)

| Dataset                                      | Size | Genetic DT% | NB% | PSO DT% | NB% | Evolutionary [91] | EBCS DT% | NB% |
|----------------------------------------------|------|-------------|-----|---------|-----|-------------------|----------|-----|
| Congressional Voting Records                 | 3    | 95.38       | 96.1| 3       | 95.38| 6 95.38           | 95.38    | 94.6|
| Soybean small                                | 21   | 85.7        | 100 | 21      | 64.2 | 100               | 94.8     | 98.3|
| Lung Cancer                                  | 6    | 98.7        | 98.7| 6       | 98.85| 98.7             | 98.8      | 98.4|
| Zoo                                          | 2    | 62.5        | 37.5| 5       | 62.5 | 37.5             | 62.5     | 66.6|
| Hepatitis                                    | 10   | 86.6        | 86.6| 9       | 86.6 | 76.6             | 90       | 86.6|
| German Credit Data                           | 8    | 78.2        | 78.2| 9       | 78.2 | 78.2             | 93.6     | 86.6|
| Dermatology                                  | 21   | 95.4        | 97.2| 20      | 75.8 | 68.9             | 93.6     | 98.1|
| Breast Cancer Wisconsin (Original)           | 9    | 92.8        | 97.1| 9       | 92.8 | 97.1             | 93.6     | 98.1|
| Wine                                         | 11   | 86.7        | 92.4| 11      | 86.7 | 92.4             | 93.6     | 94.4|
| Segment                                      | 9    | 91.9        | 80.8| 8       | 91.5 | 82.14            | 92.65    | 83.1|
| Spectf                                       | 6    | 75          | 79.16| 6       | 75       | 79.16| 6 75 | 75.8  | 75.83| 75.83|
| Connectionist Bench (Sonar)                  | 13   | 67.7        | 66.1| 13      | 67.7 | 67.7             | 67.7     | 67.7|
| Libras Movement                              | 18   | 43.5        | 43.5| 25      | 38.8 | 49.2             | 43.5     | 62.9|
| Musk (Version 1)                             | 62   | 56.6        | 75.5| 35      | 56.6 | 72.7             | 56.6     | 72.7|
| ISOLET-test                                   | 2    | 67.3        | 73.7| 236     | 69.4 | 76.1             | 58.8     | 75.6|

FIGURE 11. General comparison and size reduction between EBCS, PSO, multi objectives evolutionary, and genetic approaches.

classification accuracy (DT and NB) when compared to others in Breast Cancer Wisconsin (original), Zoo, Hepatitis, Soybean(small), Dermatology, Libras Movement, Musk (version 1), Wine and Segment datasets.

For Spectf dataset, all approaches achieve the same size reduction while EBCS achieves the same or better classification accuracy compared to others. In Mushroom dataset, EBCS improves the size reduction with the same
classification accuracy compared to PSO and genetic approaches, and the same size reduction and classification accuracy compared the multi objectives evolutionary approach.

In the Vote dataset; PSO, Genetic and multi objectives evolutionary achieve better FS than the EBCS approach. For the other datasets; PSO, Genetic and multi objectives evolutionary reduce the number of selected features and classification accuracy more than EBCS while EBCS reduces the number of selected features without significant reduction of classification accuracy using both DT and NB classifiers.

Considering results of the four approaches on all the tested datasets, EBCS achieves better FS than PSO and Genetic in 15 out of the 16 datasets. In 14 datasets, EBCS achieves better FS than multi objectives evolutionary.

G. COMPARISON EBCS WITH POPULAR TRADITIONAL FS APPROACHES

To evaluate the performance of EBCS approach, it is compared with two popular traditional FS approaches that are implemented in Weka for nominal and numerical features [97]. The first approach is best first with CFS [90], and the second is linear forward selection with Consistency Subset Evaluator (CON) [41].

In general, table 12, figure 13 and subfigures 14 (A,B) show that EBCS achieves the best classification accuracy based on DT and NB classifiers with significant size reduction compared to CFS and CON approaches. EBCS, CFS, and CON achieve 95.52%, 79.1%, and 69.8% respectively based on DT classifier, and also, EBCS, CFS and CON achieve 91.3%, 81.7% and 71.3% respectively according to NB classifier. EBCS reduces the number of feature by 79.4%
from the original datasets, but CFS and CON achieve 63.5% and 70.2% respectively for size reduction. On Zoo, Hepatitis, Dermatology, Soybean(small), Breast-W, Wine, Segment, Spectf, MoveLib, Musk, and Isolet datasets; EBCS minimizes the number of features significantly and maximizes the classification accuracy based on both DT and NB classifiers compared to CFS. On Mushroom dataset, EBCS and CFS achieve roughly the same FS. CFS achieves better size reduction, but it fails to improve the classification accuracy based on both DT and NB classifiers compared to EBCS on Credit, Lung and Sonar datasets. Only on Vote dataset, CFS achieves better FS than EBCS.

When, EBCS is compared to CON, EBCS maximizes the classification accuracy based on both DT and NB, and it minimizes the features subset significantly more than CON on Credit, Hepatitis, Vote, Soybean(small), Breast-W, Wine, Segment, Spectf, and MoveLib datasets. However, CON fails to achieve any result on Zoo and Musk datasets. On Mushroom dataset, EBCS reduces the number of features with roughly same classification accuracy (DT and NB) compared to CON. In the remaining datasets, EBCS achieves better FS than CON, which reduces the number of features and classification accuracy on both DT and NB.

Finally, EBCS achieves better FS in 15 out of the 16 tested datasets compared to CFS, and on all datasets compared to CON.

VII. CONCLUSION AND FUTURE WORK
FS is an important process for classification that aims to improve the accuracy and reduce the complexity of the classification model by selecting the relevant features. Hybrid approaches using NIAs (such as ACO, ABC, and PSO) and RST are widely used to solve FS, however they are inefficient for high dimensional, mixed, and numerical datasets.

BCS, on the other hand, is fast and efficient, less complex, easy to implement, has fewer parameters, and has a more efficient search strategy compared to other NIAs including ACO, PSO, and ABC. The main drawback of these algorithms, however, is that the efficiency of convergence decreases as the number of features increases. To improve convergence, RSTDD is used in many filter FS approaches for classification as an objective function. It offers several advantages, including easy to implement and does not need any preliminary or additional information about data, but performs inefficiently for mixed and numerical datasets.

To address these drawbacks, a new filter FS approach for classification (EBCS) is proposed. EBCS solves FS for nominal, mixed, and numerical datasets, that have different number of features, with improved computational time and classification accuracy. To achieve, EBCS developed a new initialization and search strategy, and a new more efficient objective function using RSTDD and frequent values.

EBCS was evaluated on 16 datasets including 4 nominal, 4 numerical and 8 mixed. The results show EBCS achieved better feature selection on 14 datasets, and the same feature selection on 2 datasets compared to the standard. It took less computational time when compared to the standard BCS on all datasets, and it also achieved significantly improved feature selection without significant reduction in classification accuracy for all features for all tested datasets. It also achieved better feature selection on 15 datasets compared to
the genetic, PSO, CFS, and CON approaches. In addition, it improved the FS on 14 datasets and had the same FS on one dataset compared to multi-objectives evolutionary. Finally, EBCS achieved better FS compared to experimental results in MRMCl-IWD on four out of five datasets that were used in the experiments [66].

As a future work, we plan to improve the developed approach based on two points: First, the approach used distinct values to develop the new objective function, but the unique features, and the big difference between the numbers of distinct values for each feature decrease the performance of our approach. Therefore, we need to investigate the performance of our work by dealing with these two limitations of the proposed objective function. Second, the maximum number of features that are used in this work is 617, but in cases some datasets have even larger number of features. However, to investigate the capability of our work on datasets that has greater than 1000 features, alternative initialization and global search mechanisms would need to be further investigated; to develop a more dynamic approach to automatically subdivide the datasets to search groups that provide optimal efficient computation.

REFERENCES

[1] H. F. Adam, E. E.-D. Hemdan, A. Alenezi, M. O. Alassaf, and G. B. Wills, “Internet of Things forensics: A review,” Internet Things, vol. 11, Sep. 2020, Art. no. 100220.

[2] S.-K. S. Fan, C.-J. Su, H.-T. Nien, P.-F. Tsai, and C.-Y. Cheng, “Using machine learning and big data approaches to predict travel time based on historical and real-time data from Taiwan electronic toll collection,” Soft Comput., vol. 22, no. 17, pp. 5707–5718, 2018.

[3] I. M. El-Hasnony, S. I. Barakat, M. Elhosny, and R. R. Mostafa, “Improved feature selection model for big data analytics,” IEEE Access, vol. 8, pp. 66989–67004, 2020.

[4] S. Bandaru, A. H. C. Ng, and K. Deb, “Data mining methods for knowledge discovery in multi-objective optimization: Part A—Survey,” Expert Syst. Appl., vol. 70, pp. 139–159, Mar. 2017.

[5] M. Rostami, K. Benahmoud, E. Nasiri, and S. Forouzandeh, “Review of swarm intelligence-based feature selection methods,” Eng. Appl. Artif. Intell., vol. 100, Apr. 2021, Art. no. 104210.

[6] M. Prakash, S. Karthika, and S. Ramanan, “An analysis for classification of soil data in data mining using R,” Int. J. Control Theory Appl., vol. 10, no. 2, pp. 91–98, 2017.

[7] C. Yan, J. Ma, H. Luo, and A. Patel, “Hybrid binary coral reefs optimization algorithm with simulated annealing for feature selection in high-dimensional biomedical datasets,” Chemometric Intell. Lab. Syst., vol. 184, pp. 102–111, Jan. 2019.

[8] S. M. Raza and U. Qamar, Understanding and Using Rough Set Based Feature Selection: Concepts, Techniques and Applications. Singapore: Springer, 2019.

[9] W. Zhao, S. Han, W. Meng, D. Sun, and R. Q. Hu, “BSDP: Big sensor data preprocessing in multi-source fusion positioning system using compressive sensing,” IEEE Trans. Veh. Technol., vol. 68, no. 9, pp. 8866–8880, Sep. 2019.

[10] Q. Al-Tashi, S. J. Abdulkadir, H. M. Rais, S. Mirjalili, and H. Alhussian, “Approaches to multi-objective feature selection: A systematic literature review,” IEEE Access, vol. 8, pp. 125076–125096, 2020.

[11] U. M. Khare and R. Dhanalakshmi, “Stability of feature selection algorithm: A review,” J. King Saud Univ.-Comput. Inf. Sci., Jun. 2019.

[12] H. Liu and H. Motoda, Feature Selection for Knowledge Discovery and Data Mining. New York, NY, USA: Springer, 1998.

[13] A. N. M. B. Rashid, M. Ahmed, L. F. Sikos, and P. Haskell-Dowland, “Cooperative co-evolution for feature selection in big data with random feature grouping,” J. Big Data, vol. 7, no. 1, pp. 1–42, Dec. 2020.

[14] D. Olešký and T. Novošel, “Comparison of feature selection techniques in knowledge discovery process,” TEM J., vol. 3, no. 4, pp. 285, 2014.

[15] B. Martin, A. Goldszejt, L. Granvilliers, and C. Jermann, “Meta-heuristic methods such as particle swarm optimization and genetic algorithms have been applied in solving multi-objective optimization problems, and have been observed to be useful for generating a good approximation of Pareto optimal solutions. This paper suggests a multi-objective particle swarm optimization (MOPSO) utilizing generalized data envelopment analysis (GDEA) in order to decide adaptively,” J. Global Optim., vol. 64, no. 1, pp. 117–139, 2016.

[16] H. Zhao, A. P. Sinha, and W. Ge, “Effects of feature construction on classification performance: An empirical study in bank failure prediction,” Expert Syst. Appl., vol. 36, no. 2, pp. 2633–2644, Mar. 2009.

[17] M. Sharma and P. Kaur, “A comprehensive analysis of nature-inspired meta-heuristic techniques for feature selection problem,” Arch. Comput. Methods Eng., vol. 28, pp. 1–25, May 2020.

[18] B. H. Nguyen, B. Xue, and M. Zhang, “A survey on swarm intelligence approaches to feature selection in data mining,” Swarm Evol. Comput., vol. 54, May 2020, Art. no. 100663.

[19] P. D. Sheth, S. T. Patil, and M. L. Dhore, “Evolutionary computing for clinical dataset classification using a novel feature selection algorithm,” J. King Saud Univ.-Comput. Inf. Sci., Dec. 2020.

[20] P. Shrivastava, A. Shukla, P. Vepakomma, N. Bhansali, and K. Verma, “A survey of nature-inspired algorithms for feature selection to identify Parkinson’s disease,” Comput. Methods Programs Biomed., vol. 139, pp. 171–179, Feb. 2019.

[21] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou, “Multi-verse optimizer: A nature-inspired algorithm for global optimization,” Neural Comput. Appl., vol. 27, no. 2, pp. 495–513, 2016.

[22] H. Peng, C. Ying, S. Tan, B. Hu, and Z. Sun, “An improved feature selection algorithm based on ant colony optimization,” IEEE Access, vol. 6, pp. 69203–69209, 2018.

[23] M. Dorigo, V. Maniezzo, and A. Colomi, “Ant system: Optimization by a colony of cooperating agents,” IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 26, no. 1, pp. 29–41, Feb. 1996.

[24] R. Poli, J. Kennedy, and T. Blackwell, “Particle swarm optimization,” Swarm Intell., vol. 1, no. 1, pp. 33–57, Jun. 2007.

[25] D. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, and M. Zaidi, “The bees algorithm—A novel tool for complex optimisation problems,” in Proc. 2nd Virtual Int. Conf. Intell. Prod. Mach. Syst. (IPROMS), 2006, pp. 454–459.

[26] X.-S. Yang and S. Deb, “Cuckoo search via Lévy flights,” in Proc. World Congr. Nature Biologically Inspired Comput. (NaBIC), Dec. 2009, pp. 210–214.

[27] D. Rodrigues, L. A. M. Pereira, T. N. S. Almeida, J. P. A. Papa, A. N. Souza, and O. R. Ramos, “‘X-S. Yang. ’BSC: A binary cuckoo search algorithm for feature selection,’” in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), May 2013, pp. 465–468.

[28] A. Alia and A. Taweel, “Hybrid nature inspired algorithms and rough set theory in feature selection for classification: A review,” Algorithms, vol. 3, p. 7, Jul. 2016.

[29] X.-F. Song, Y. Zhang, D.-W. Gong, and X.-Z. Gao, “A fast hybrid feature selection based on correlation-guided clustering and particle swarm optimization for high-dimensional data,” IEEE Trans. Cybern., early access, Mar. 17, 2021, doi: 10.1109/TCYB.2021.3061152.

[30] E. Costa, A. Lorena, A. Carvalho, and A. Freitas, “A review of performance evaluation measures for hierarchical classifiers,” in Proc. Eval. Methods Mach. Learn. II, Papers From AAAI Workshop, 2007, pp. 1–6.

[31] X. Guo, F.-L. Chung, G. Li, and L. Zhang, “Multi-label bioinformatics data classification with ensemble embedded feature selection,” IEEE Access, vol. 7, pp. 103863–103875, 2019.

[32] D. Zouache and F. B. Abdelaziz, “A cooperative swarm intelligence algorithm based on quantum-inspired and rough sets for feature selection,” Comput. Ind. Eng., vol. 115, pp. 26–36, Jan. 2018.

[33] L. Yu and H. Liu, “Efficient feature selection via analysis of relevance and redundancy,” J. Mach. Learn. Res., vol. 5, no. 10, pp. 1205–1224, 2004.

[34] P. A. Estévez, M. Tesmer, C. A. Perez, and J. M. Zurada, “Normalized mutual information feature selection,” IEEE Trans. Neural Netw., vol. 20, no. 2, pp. 189–201, Feb. 2009.

[35] S. García, J. Luengo, and F. Herrera, “Feature selection,” in Data Preprocessing in Data Mining. Cham, Switzerland: Springer, 2015, pp. 163–193.
Z. Beheshti and S. M. H. Shamsuddin, “Modified cuckoo search algorithm with rough sets for feature selection,” Neural Comput. Appl., vol. 29, pp. 1–10, Jul. 2018.

A. F. Aliya and A. Taweel, “Feature selection based on hybrid binary cuckoo search and rough set theory in classification for nominal datasets,” Int. J. Inf. Technol. Comput. Sci., vol. 9, no. 4, p. 63, 2017.

R. Salgotra, U. Singh, and S. Saha, “New cuckoo search algorithms with enhanced exploration and exploitation properties,” Expert Syst. Appl., vol. 95, pp. 384–420, Apr. 2018.

T. Karthikeyan and P. Thangaraju, “Best first and greedy search based CFS-Naive Bayes classification algorithms for hepatitis diagnosis,” Biosci. Biotechnol. Res. Asia, vol. 12, no. 1, pp. 983–990, Apr. 2015.

M. Dash, H. Liu, and H. Motoda, “Consistency based feature selection,” in Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining, Berlin, Germany: Springer, 2000, pp. 98–109.

Y. Zhang, S. Cheng, Y. Shi, D.-W. Gong, and X. Zhao, “Cost-sensitive feature selection using two-archive multi-objective artificial bee colony algorithm,” Expert Syst. Appl., vol. 137, pp. 46–58, Dec. 2019.

Y. Zhang, D.-W. Gong, X.-Z. Gao, T. Tian, and X.-Y. Sun, “Binary differential evolution with self-learning for multi-objective feature selection,” Inf. Sci., vol. 507, pp. 67–85, Jan. 2020.

R. Tiwari and M. P. Singh, “Correlation-based attribute selection using genetic algorithm,” Int. J. Comput. Appl., vol. 4, no. 8, pp. 28–34, Aug. 2010.

S. Singh and A. K. Singh, “Web-spam features selection using CFS-PSO,” Procedia Comput. Sci., vol. 125, pp. 568–575, Jan. 2018.

L. Ke, Z. Feng, and Z. Ren, “An efficient ant colony optimization approach to attribute reduction in rough set theory,” Pattern Recognit. Lett., vol. 29, no. 9, pp. 1351–1357, Jul. 2008.

X. Wang, J. Yang, X. Teng, W. Xia, and R. Jensen, “Feature selection based on rough sets and particle swarm optimization,” Pattern Recognit. Lett., vol. 28, no. 4, pp. 459–471, 2007.

H. Shen, S. Yang, and J. Liu, “An attribute reduction of rough set based on PSO,” in Rough Set and Knowledge Technology, Berlin, Germany: Springer, 2010, pp. 695–702.

M. Mafarja and D. Eleyan, “Ant colony optimization based feature selection in rough set theory,” Int. J. Comput. Sci. Electron. Eng., vol. 1, no. 5, pp. 244–247, 2013.

R. Jensen and Q. Shen, “Finding rough set reducts with ant colony optimization,” in Proc. UK Workshop Comput. Intell., vol. 1, 2003, pp. 15–22.

S. Chebrolu and S. G. Sanjevi, “Attribute reduction on continuous data in rough set theory using ant colony optimization metaheuristic,” in Proc. Int. Conf. Comput. Inform. Sci. (ICCI), 2015, pp. 17–22.

Z. Beheshti and S. M. H. Shamsuddin, “A review of population-based meta-heuristic algorithms,” Int. J. Adv. Soft Comput. Appl., vol. 5, no. 1, pp. 1–35, 2013.

E.-G. Talbi, Metaheuristics: From Design to Implementation, vol. 74, Hoboken, NJ, USA: Wiley, 2009.

H. H. Inbarani, A. T. Azar, and G. Jothi, “Supervised hybrid feature selection via submodular function,” Soft Comput., vol. 21, pp. 1–27, Aug. 2016.
[87] J. D. Rodriguez, A. Perez, and J. A. Lozano, “Sensitivity analysis of k-fold cross validation in prediction error estimation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, no. 3, pp. 569–575, Mar. 2010.

[88] C. Blake, E. Keogh, and C. Merz. (2018). UCI Repository of Machine Learning Databases. [Online]. Available: http://www.ics.uci.edu/~mlearn/mlrepository.html

[89] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA, USA: Addison-Wesley, 1989.

[90] M. A. Hall, “Correlation-based feature selection for machine learning,” Ph.D. dissertation, Univ. Waikato, Hamilton, New Zealand, 1999.

[91] F. Jiménez, G. Sánchez, J. M. García, G. Sciavicco, and L. Miralles, “Multi-objective evolutionary feature selection for online sales forecasting,” Neurocomputing, vol. 234, pp. 75–92, Apr. 2017.

[92] A. Moraglio, C. Di Chio, and R. Poli, “Geometric particle swarm optimisation,” in Genetic Programming. Berlin, Germany: Springer, 2007, pp. 125–136.

[93] L. Xu, P. Yan, and T. Chang, “Best first strategy for feature selection,” in Proc. 9th Int. Conf. Pattern Recognit., 1988, pp. 706–708.

[94] M. Gutlein, E. Frank, M. Hall, and A. Karwath, “Large-scale attribute selection using wrappers,” in Proc. IEEE Symp. Comput. Intell. Data Mining, Mar. 2009, pp. 332–339.

[95] G. Holmes, A. Donkin, and I. H. Witten, “WEKA: A machine learning workbench,” in Proc. Austral. New Zealand Intell. Inf. Syst. Conf. (ANZIS), 1994, pp. 357–361.

[96] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, “Top 10 algorithms in data mining,” Knowl. Inf. Syst., vol. 14, no. 1, pp. 1–37, 2008.

[97] N. Parthalain, Q. Shen, and R. Jensen, “A distance measure approach to exploring the rough set boundary region for attribute reduction,” IEEE Trans. Knowl. Data Eng., vol. 22, no. 3, pp. 305–317, Mar. 2010.

[98] B. Lindgren, Statistical Theory. Evanston, IL, USA: Routledge, 2017.

[99] J. Cortes, A. Caris, and B. Depaire, “Evaluating and optimizing meta-heuristic algorithms: A step ahead in VRP research,” in Proc. 29th Annu. Conf. Belgian Opér. Res. Soc., Antwerp, Belgium, 2015.

AHMED ALIA received the B.Sc. degree in computer science from An-Najah National University, Palestine, in 2006, and the M.Sc. degree in computer science and engineering (computing specialization) from Birzeit University, Palestine, in 2015. He is currently pursuing the Ph.D. degree in information technology with the Institute for Advanced Simulation, Forschungszentrum Jülich, in collaboration with the University of Wuppertal, Germany. From 2008 to 2020, he was a Research and Teaching Assistant at the Faculty of Engineering and Information Technology, An-Najah National University. His research interests include feature selection, machine learning, deep learning, data analysis, and crowd behavior analysis.

ADEL TAWEEL (Member, IEEE) is a Faculty Member the Department of Computer Science, Birzeit University. He has previously held academic posts at the Universities of Keele, Manchestester, and Birmingham, and King’s College London, where he still holds an academic position. He has led, managed, and worked on several EU, USA, and U.K. funded projects, including ePCRN, CLEF, CLEFServices, CLARiFi, FP7 TRANSFoRm, PEARL, ePCRN, ENJECT, and Health Connect. He has published more than 100 peer-reviewed scientific publications in distributed software engineering, component based, service-based and agent-based systems, and medical informatics. He is a Charted Engineer and a member of a number of professional institutions (BCS and IET), and has served as a member of several U.K. national committees and scientific consultant for several companies and government ministries, including U.K. Department of Health. He has served as the chair and the session chair of several international conferences. He is also a Standing Committee Member, Editorial Member, and a reviewer of several books, and international journals and conferences.

**VOLUME 9, 2021**

119453