A Comprehensive Survey on Recent Metaheuristics for Feature Selection

Tansel Dokeroglu[a], Ayça Deniz[b], Hakan Ezgi Kiziloz[c,d,*]

[a]*Department of Software Engineering, Çankaya University, Ankara, Turkey*
[b]*Department of Computer Engineering, Middle East Technical University, Ankara, Turkey*
[c]*Department of Computing, Sheffield Hallam University, Sheffield, United Kingdom*
[d]*Department of Computer Engineering, University of Turkish Aeronautical Association, Ankara, Turkey*

Abstract

Feature selection has become an indispensable machine learning process for data preprocessing due to the ever-increasing sizes in actual data. There have been many solution methods proposed for feature selection since the 1970s. For the last two decades, we have witnessed the superiority of metaheuristic feature selection algorithms, and tens of new ones are being proposed every year. This survey focuses on the most outstanding recent metaheuristic feature selection algorithms of the last two decades in terms of their performance in exploration/exploitation operators, selection methods, transfer functions, fitness value evaluation, and parameter setting techniques. Current challenges of the metaheuristic feature selection algorithms and possible future research topics are examined and brought to the attention of the researchers as well.

Keywords: Feature selection, Survey, Metaheuristic algorithms, Machine learning, Classification.

1. Introduction

The data generated by contemporary applications are increasing drastically in terms of the number of instances and features. This rapid increase in data sizes brought by Big Data has become an important issue for recent machine learning algorithms [1, 2]. Feature selection is one of the commonly used preprocessing techniques of the machine learning community for the removal of irrelevant, noisy, and redundant data while increasing the learning accuracy and improving the quality of the classification results. Lately, it has become an essential task in the development of efficient data mining and machine learning algorithms. Therefore, feature selection has been the focus of many studies for quite some time [3]. Many feature selection algorithms have been proposed in the literature to obtain the most informative subsets that provide higher quality results for classification and clustering. As of February 2022, one million and 50 thousand related articles are displayed at Google Scholar when the keywords "feature selection" are searched.

*Corresponding author
Email addresses: tdokeroglu@cankaya.edu.tr (Tansel Dokeroglu), ayca.deniz@metu.edu.tr (Ayça Deniz), hakanezgi@etu.edu.tr (Hakan Ezgi Kiziloz)

There are three main methods for feature selection: filter, wrapper, and embedded methods. Filter methods (e.g. information gain) are based on a statistical analysis of the attributes. Wrapper methods utilize a search algorithm along with a classifier and test the performance of each subset of features. In embedded methods, the search for the best performing feature subset and classification are handled simultaneously. There exists a trade-off between the filter and wrapper methods: even though filter methods are easier to calculate, wrapper methods outperform filter methods [4].

Obtaining the optimal subset of features is an NP-Hard problem [5, 6]. Metaheuristic algorithms are one of the best tools to deal with combinatorial problems [7–9]. Moreover, studies show that metaheuristic algorithms perform better than exhaustive or greedy approaches [10]. State-of-the-art metaheuristic algorithms are highly influenced by nature, and today, they are widely used in the feature selection domain [11–18].

In this review, we focus on recent generation metaheuristics of the last two decades that have been proposed for the solution of feature selection other than the classical methods, Genetic Algorithm (GA) [19], Particle Swarm Optimization (PSO) [20, 21], Ant Colony Optimization (ACO) [22], Simulated Annealing (SA) [23], Genetic Programming (GP) [24], Differential Evolution (DE) [25], Tabu Search (TS) [26], and Artificial Immune Systems Algorithm (AIS) [27]. These are the most studied conventional methods, and there are still studies (including hybrid and multiobjective versions) published with these classical metaheuristic algorithms [28]. In Section 2, we provide brief information about previous surveys on these metaheuristics before starting to review our selected recent metaheuristics.

According to the No-Free-Lunch (NFL) theorem, no optimization algorithm is good enough to solve all problems [29]. Therefore, there is no guarantee to find the best set of features on all problem domains using a single metaheuristic. Considering these issues, there will always be a possibility of obtaining better results with new metaheuristics on feature selection. In the literature, hundreds of new articles are being published every year. These studies produce high-quality solutions with metaheuristics on feature selection, and this intense interest attracts the attention of numerous researchers. The reported results of these algorithms are remarkable on huge datasets. In this context, we tried to review the studies of distinguished academicians who received many citations and whose papers are published in the top journals and conferences.

Figure 1 gives the number of available studies related to the classical metaheuristics. The results are obtained from the Google Scholar website using the keywords "feature selection" + "genetic algorithm", and so forth. There are more than 200 thousand papers on feature selection related to classical feature selection algorithms, and the most studied two are observed to be Genetic Algorithm and Particle Swarm Optimization.

In Table 1, our selected recent/new 22 metaheuristics for feature selection are listed chronologically. During the selection of these metaheuristics, we focused on the number of citations, promising results, computation performance, prediction accuracy, and their main contributions. Moreover, we inspected the strengths and weaknesses of these new metaheuristic algorithms. There are more than 59 thousand search

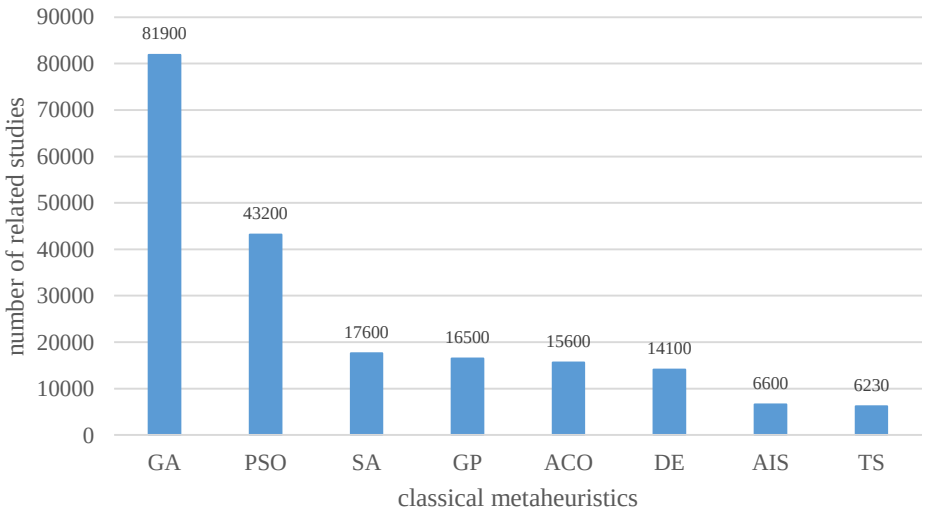

Figure 1: The number of feature selection studies related with classical metaheuristics on Google Scholar. (The results are obtained with keywords "feature selection" + "the name of the metaheuristic" as of February 2022 (searched in the whole document).)

results related to these 22 recent metaheuristics on Google Scholar. The three most cited metaheuristics are Artificial Bee Colony Algorithm (ABC), Firefly Algorithm (FA), and Cuckoo Search (CS) with more than 23 thousand search results. In Figure 2, the search results of the algorithms on Google Scholar can be observed.

Figures 3 and 4 give information about the number of available studies related to the classical and recent metaheuristic feature selection algorithms with Scopus search results respectively. It is seen that the results are proportional to the results of Google Scholar.

A comprehensive review of previous related surveys is well studied in Section 2. Our study is unique when compared with other surveys in terms of selected metaheuristics. Section 3 gives information about the common definitions/structures, operators, solution representations, transfer functions, fitness value evaluations, and the best performing classifiers used by the algorithms. In Section 4, brief information about the recent metaheuristics selected in this survey for the feature selection is given. Section 5 presents multiobjective versions of the metaheuristic algorithms. In Section 6, hybrid metaheuristic and hyperheuristic algorithms that combine recent and classical approaches for the solution of the feature selection are reviewed. Section 7 gives information about the benchmark datasets and search engines that are used by feature selection algorithms in experiments. Our concluding remarks, open problems, and challenging issues of feature selection are presented in Section 8.

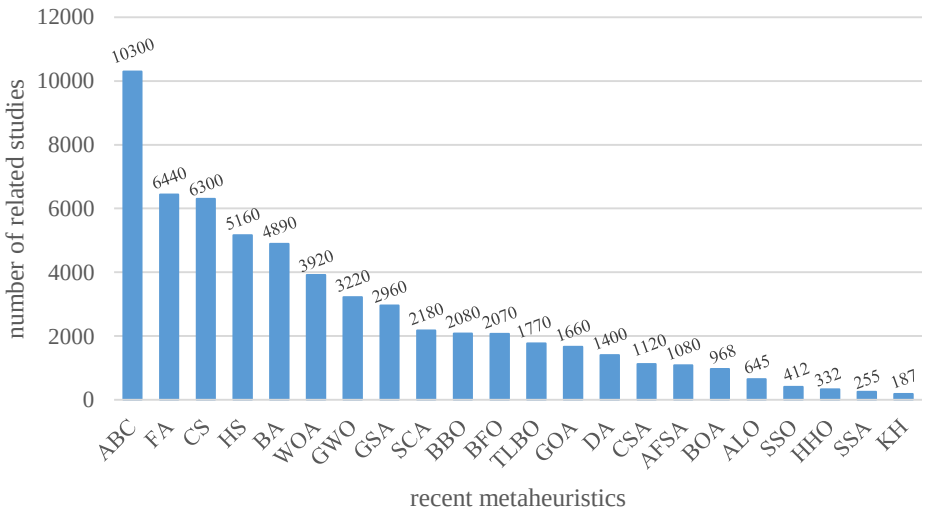

Figure 2: The number of feature selection studies related with recent metaheuristics on Google scholar. (The results are obtained with keywords "feature selection" + "the name of the metaheuristic" as of February 2022 (searched in the whole document).)

2. Previous surveys for feature selection

In this part of our survey, we aimed to summarize the previous reviews of feature selection in the last 30 years. More than 40 most-cited and recent surveys in the literature are listed chronologically. We believe that it will be a good resource for interested readers. There are many surveys in this area, and they contain good information on the subject. However, there is no detailed research like our review on recent state-of-the-art metaheuristics used for feature selection.

Although the first papers about feature selection date back to the 1970s, the best survey papers of this area started appearing in 1992. Kira et al. analyzed the strengths and weaknesses of feature selection and introduced a new statistical algorithm (Relief) in 1992 [52]. The Relief works in linear time complexity according to the number of given features and training instances. The authors reported the comparison results of Relief and other algorithms, which supports their theoretical analysis. Siedlecki & Sklansky analyzed the feature selection methods, branch-and-bound search, and beam search for multidimensional pattern classification in 1993 [53]. They reported the benefits of using GA and SA. Dash & Liu focused on a set of methods used for the feature selection between 1970 to 1997 [54]. They grouped existing methods in terms of evaluation functions and generation. The advantages and disadvantages of the methods were explained, and research areas were investigated. Martin-Bautista & Vila reviewed data mining and knowledge discovery issues of feature selection [55]. A comprehensive survey about GA to select the most

Table 1: Our selected new/recent metaheuristics for the feature selection problem.

Acronym	Metaheuristic	Year
HS	Harmony Search [30]	2001
BFO	Bacterial Foraging Optimization [31]	2002
AFSA	Artificial Fish Swarm Algorithm [32]	2003
ABC	Artificial Bee Colony Algorithm [33]	2005
BBO	Biogeography-Based Optimization [34]	2008
FA	Firefly Algorithm [35]	2009
GSA	Gravitational Search Algorithm [36]	2009
CS	Cuckoo Search [37]	2009
BA	Bat Algorithm [38]	2010
TLBO	Teaching-Learning-Based Optimization [39]	2011
KH	Krill Herd [40]	2012
SSO	Social Spider Optimization [41]	2013
GWO	Grey Wolf Optimization [42]	2014
ALO	Ant Lion Optimization [43]	2015
SCA	Sine Cosine Algorithm [44]	2016
WOA	Whale Optimization Algorithm [45]	2016
CSA	Crow Search Algorithm [46]	2016
DA	Dragonfly Algorithm [47]	2016
SSA	Salp Swarm Algorithm [48]	2017
GOA	Grasshopper Optimization Algorithm [49]	2018
BOA	Butterfly Optimization Algorithm [50]	2019
HHO	Harris' Hawk Optimization [51]	2019

relevant features was presented. Different fitness values and parameters used by this evolutionary algorithm were reviewed. Philippe & Gallinari prepared a review of neural network approaches to feature selection [56]. The authors introduced baseline statistical methods used in regression and classification in their study. The methods were compared using benchmark problem instances. Molina et al. evaluated the fundamental algorithms in the literature using a controlled scenario, and they developed a scoring framework to analyze the amount of relevancy and redundancy of datasets [57]. Liu & Yu wrote a comprehensive survey on the main approaches and algorithms of feature selection for classification and clustering [6]. A new framework was proposed for search evaluation, strategies, and data mining. They built an integrated method for

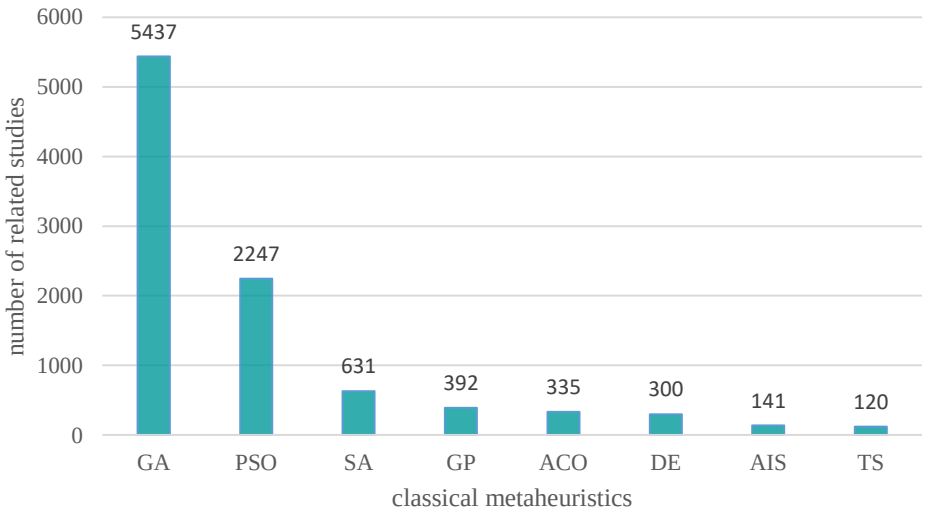

Figure 3: The number of feature selection studies related with classical metaheuristics on Scopus. (The results are obtained from the keywords "feature selection" + "the name of the metaheuristic" as of February 2022 (including the title, abstract and keywords of the documents).)

intelligent feature selection. Examples were given on feature selection to integrate a meta-algorithm. The authors identified the challenges and main trends in feature selection. Saeys et al. presented a survey on feature selection for bioinformatic studies [58]. They tried to catch the reader's attention to the feature selection area by developing a taxonomy of feature selection techniques, their use, and their potential in bioinformatics. Yusta provided an overview of metaheuristics (GRASP, TS, and Memetic Algorithm) that were proposed for the feature selection [59]. The authors observed that the GRASP and TS algorithms could provide better results than a GA.

Zhao et al. mentioned that no repository collects feature selection algorithms, and they proposed a repository with the most popular feature selection algorithms for comparison [60]. The repository provides a more reliable platform to evaluate newly proposed feature selection methods. Liu et al. prepared a survey on the key components and developments of feature selection on data mining [61]. They reviewed research fields of contemporary interests and continuing research activities and defined multidisciplinary research areas. De La Iglesia reviewed the evolutionary algorithms for feature selection [62]. The study reported models for providing computational efficiency and understandable approaches. GA, GP, ACO, and PSO algorithms were investigated in this study. Ganapathy et al. reviewed features for intrusion detection in networks [63]. Smart software agents, GA, neural networks, neuro-GA, fuzzy techniques, rough sets, and PSO intelligence were analyzed. Bolón-Canedo et al. created several synthetic datasets and employed many algorithms to observe their performance on the task of selecting features [64]. Zhai et al. analyzed the origins

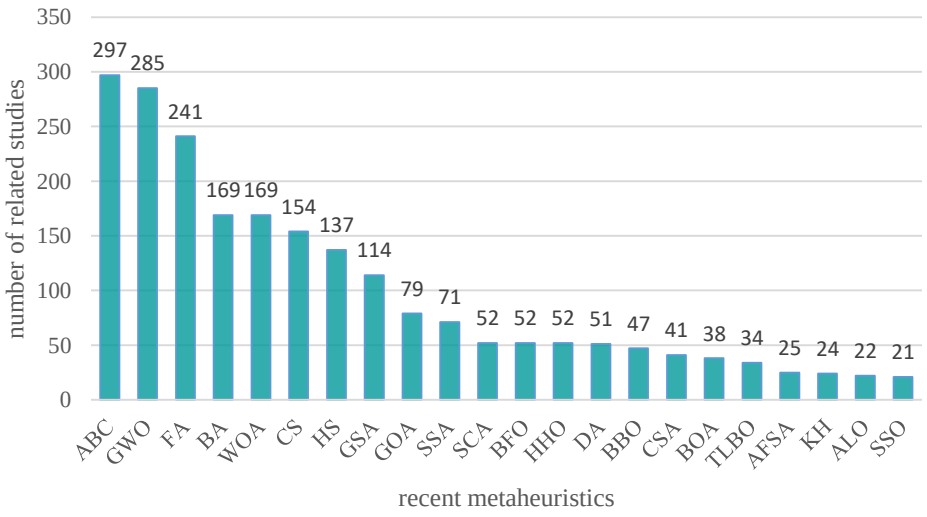

Figure 4: The number of feature selection studies related with recent metaheuristics on Scopus. (The results are obtained from the keywords "feature selection" + "the name of the metaheuristic" as of February 2022 (including the title, abstract, and keywords of the document).)

of Big Data and the evolution of feature selection using popular datasets in analytics and computational intelligence research [65]. The authors reviewed the state-of-the-art feature selection methods to evaluate the existing approaches for Big Data. Kumar & Minz introduced the general procedures, fitness evaluation processes, and the basic structures of feature selection [66]. A comprehensive review and evaluation of feature selection methods were performed. Moreover, they summarized the applications, challenges in this area, and future research directions of the problem. Chandrashekar & Sahin provided an overview of feature selection methods in the literature in 2014 [4]. They gave a generic introduction to the elimination of variables that are used in machine learning. They focused on the methods: filter, wrapper and embedded. Tang et al. prepared a survey about the most-used feature selection algorithms: Linear Discriminant Analysis, Principal Component Analysis, Information Gain, Canonical Correlation Analysis, Relief, Lasso, and Fisher Score [67]. Vergara & Estévez presented a review for the information-theoretic methods [68]. The difficulty of optimal feature selection was described. The feature relevance, redundancy, and synergy, and Markov blanket were defined. A set of related problems were introduced. Khalid et al. prepared a survey on widely used feature selection and extraction techniques for improving the performance of classifiers [69]. The strengths and weaknesses of widely used feature selection methods were discussed. In their survey, Xue et al. indicated that the feature selection problem is very hard due to its complex search space [70]. They gave brief information about successful evolutionary computation methods that have recently gained much attention. They reviewed new guidelines for evaluating the approaches. Current challenges were identified for

future research. Ang et al. presented a taxonomy of supervised, unsupervised, and semi-supervised feature selection alternatives and reviewed the gene selection methods in the literature [71]. The experiments verified that the prediction accuracy of unsupervised and semi-supervised feature selection is promising. Jović et al. mentioned the difficulty of searching exhaustively for the optimal feature subset [72]. In their review, they considered the most common feature selection algorithms (filter, wrapper, and embedded). Hybrid versions of these algorithms were also inspected. Miao & Niu reviewed the feature selection studies on computer vision and text mining [9]. Experiments showed that unsupervised feature selection performs well to improve the performance of clustering. Gnana et al. prepared a survey about various feature selection methods in 2016 [73]. Wang et al. surveyed the principles of feature selection and recent applications in big bioinformatics data [74]. They formalized a combinatorial feature selection problem and classified the methods as exhaustive, heuristic, and hybrid search algorithms. Li et al. surveyed recent advances in feature selection [75]. The applications of feature selection were introduced. The applications were mainly in multimedia retrieval, social media, and bioinformatics. Sheikhpour et al. investigated semi-supervised methods for feature selection [76]. Li et al. provided a comprehensive study in big data feature selection [77]. They reviewed the feature selection from a data perspective for conventional structured, heterogeneous, and streaming data. They classified the feature selection algorithms into similarity, information-theoretical, sparse-learning, and statistical techniques. They provided a repository for open-source feature selection algorithms. Moreover, open problems and challenges were listed in this study.

Cai et al. discussed evaluation measures for feature selection [78]. Urbanowicz et al. focused on Relief-based algorithms (RBAs), which are filter-style feature selection algorithms [79]. The authors included a review of RBA research. Brezovcnik et al. prepared a comprehensive literature review of 64 Swarm intelligence algorithms for feature selection [80]. The datasets used by the algorithms were also listed. Deng et al. gave a review on text classification feature selection techniques [81]. The popular representation of documents was presented, and similarity measures were reported. They reviewed the Nearest Neighbors, Naive Bayes, Support Vector Machines (SVM), Decision Tree, and Neural Networks. They also surveyed state-of-the-art filter, wrapper, embedded, and hybrid methods of feature selection. Venkatesh & Anuradha prepared a survey about dynamic IoT and web-based application data feature selection [82]. The scalability is a critical concern of feature selection methods for this domain. Fernandez et al. provided a comprehensive review of unsupervised feature selection methods [83]. They prepared a taxonomy of the methods, the main characteristics and ideas about the algorithms. Important open challenges in this research area were discussed. Bolon-Canedo et al. prepared a survey about ensemble learning for feature selection [84] . They reviewed the recent advances and future trends. Liu & Wang provided a brief survey on nature-inspired metaheuristics of the last decade [85]. The main challenges were discussed in terms of scalability, stability, accuracy, and computation.

Al-Tashi et al. presented a literature review of the multiobjective feature selection problems and proposed

techniques [86]. They reviewed related studies between 2012 and 2019. The review verified that there exists no perfect method for the multiobjective version of the feature selection problem. Agrawal et al. presented a literature review on binary metaheuristic algorithms developed for feature selection between 2009 and 2019 [87]. The metaheuristic algorithms were classified into four categories. More than a hundred metaheuristic algorithms were presented in this study. Challenges and issues were discussed, and research gaps were highlighted. Moreover, a case study was presented on University of California, Irvine (UCI) Machine Learning Repository datasets.

Some of the state-of-the-art studies on classical metaheuristic algorithms are listed as follows. Unler & Murat proposed a metaheuristic algorithm, i.e., a modified version of PSO, for the binary feature selection problem [88]. Experiments verified that the performance of the algorithm is better than other classical algorithms. There exist many other studies that apply different versions of PSO on feature selection task [89–93]. Deniz et al. proposed a multiobjective GA for the feature selection of binary classification [10]. The algorithm was compared with PSO, Greedy, TS, and SS methods. Xue et al. presented a multiobjective study to generate a Pareto front of feature subsets [94]. Taradeh et al. proposed an algorithm with evolutionary mutation and crossover operators to solve the feature selection problem [95]. Hancer et al. developed two new DE-based filter approaches for the feature selection problem [96].

3. Common structures of recent metaheuristics

In this part of our survey, we give information about the common definitions/structures, solution representation, operators, selection methods, fitness value evaluations, and common machine learning methods (classifiers) used by the recent metaheuristics that we have reviewed in this survey. We focus on common approaches of most of these algorithms, although there are many more structures or techniques applied by the selected metaheuristics. For example, the binary representation of the solutions is the same in the majority of the methods.

3.1. Solution representation

The metaheuristic optimization approaches use a population of candidate solutions. The solutions are usually represented as a vector of values. For metaheuristic feature selection algorithms, the representation of a solution is generally a binary encoding of a selected set of features. In Figure 5, we see a candidate solution with its selected features. In this solution, there are eight features, and four of them are selected. In this representation, 2^n many feature subsets can be generated, where n is the number of features. This is theoretically the same with the formal definition of the number of feature subsets. Therefore, it is possible to find every subset of features with this representation. That is why it is the most common way of representing the information of individuals.

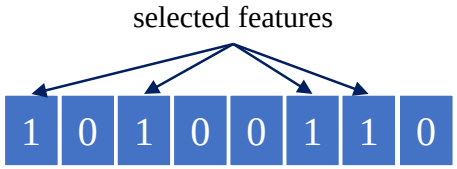

Figure 5: The binary encoding of a solution used by wrapper-based feature selection algorithms.

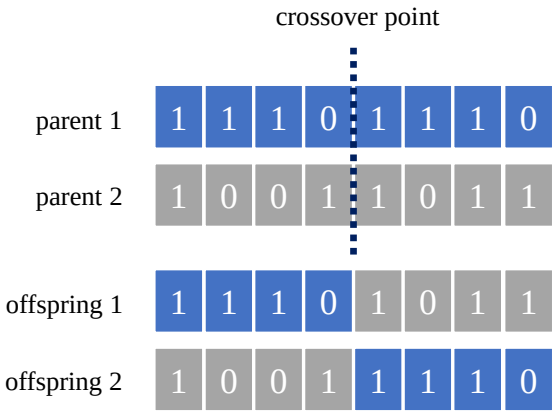

Figure 6: A sample exploration operator for binary feature selection metaheuristics.

3.2. Exploration and exploitation

Balancing the exploration (diversification) and exploitation (intensification) activities is a crucial task for the optimization performance, accuracy prediction, and convergence speed of the metaheuristic algorithms. Detailed research on this issue can be found in a study by Xu & Zhang [97]. There is no clear answer to this question yet. With fitness landscape analysis and information landscape approaches, a better balance between these activities is decided [98]. The balance between exploration and exploitation is not to be 50% of the total optimization time. This is an issue that should be well set by the dynamics of the proposed algorithm [99]. Metaheuristics can adaptively tune these phases and perform considerably better than other algorithms. Unexplored areas should be evenly visited as much as possible, and the search should not get stuck into local optima. Exploration is generally used to get rid of the local optima, whereas exploitation searches for the neighboring alternative of the current solution. For a common exploration operator, e.g., crossover in GA, please see Figure 6. A big part of the solution is changed after applying this technique.

Using exploitation operators, promising regions can be searched to find better results. For exploitation of the solutions, an operator similar to mutation in GA is generally used (see Figure 7). A feature (gene) is

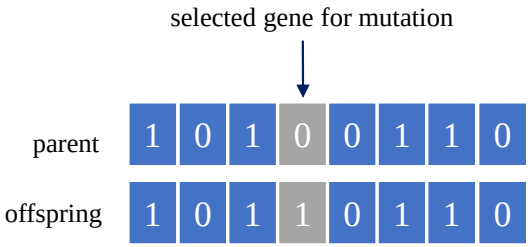

Figure 7: A sample exploitation operator for binary feature selection metaheuristics.

chosen from the array of features and changed with zero if its value is one and vice versa.

3.3. Fitness value evaluation

The fitness function calculates the solution quality of selected features. Different fitness functions may be preferred during the optimization process. Selection of this function can significantly affect the performance of the optimization regarding its speed and accurate prediction capability. Moreover, in the feature selection task, minimizing the number of selected features is also important. Accordingly, a common fitness function used by recent metaheuristics is as given below [12, 100]:

$$Fit = \alpha ER + \beta\left(\frac{|S|}{|O|}\right) \tag{1}$$

where ER is the classification error, |S| is the length of the selected subset of features, and |O| is the length of all features in the original dataset. $\alpha \in [0, 1]$ and $\beta = (1 - \alpha)$ are two values that indicate the impact of classification error and feature size, respectively.

Multiobjective versions of the functions are also possible. In multiobjective versions of this problem, the fitness function tries to maximize the classification accuracy while minimizing the number of features. Accordingly, the multiobjective formula with both objectives is modeled as follows:

$$\begin{aligned} &\textit{minimize } f_1,\ f_2 \\ &\textit{subject to} \\ &\qquad f_1 = |S| \\ &\qquad f_2 = ER \end{aligned} \tag{2}$$

The length of the optimal subset of features is initially unknown, and exploring/exploiting the sets of features is the best way for minimizing $|S|$. In case it is known prior to the study, it is still hard to obtain the optimal subset as it requires a search over $\binom{|O|}{|S|}$ combinations. Second step is to evaluate the quality of the selected features. *Accuracy* and F_1*-measure* are the common metrics used to compare the classification

quality. *Accuracy* is calculated with dividing correctly classified instances by all instances. F_1*-measure* gives the balance between *precision* and *recall*. These metrics evaluate correctness of true predictions and ability to detect true instances, respectively. When the dataset consists of imbalanced classes, F_1*-measure* is preferred over *Accuracy* for evaluation.

3.4. Transfer function operators

The metaheuristics are mostly developed for solving continuous optimization problems. Later, their binary (discrete) versions are proposed. During this transformation, the continuous approaches are converted into a binary version utilizing transfer functions. Mostly, S and V-shaped transfer functions are used in the literature. A transfer function shows the probability of selecting a feature or not. Kennedy & Eberhart proposed a transfer function for PSO as given in Equation 3 [101]:

$$T(x_j^i(t)) = \frac{1}{1 + exp^{-x_j^i(t)}} \tag{3}$$

where x_j^i is the j-th feature in solution x, in the i-th dimension, and t shows the current iteration. In S-shaped transfer functions, an element can be updated using Equation 4:

$$x_j^i(t+1) = \begin{cases} 1, & r < T(x_j^i(t+1)) \\ 0, & otherwise \end{cases} \tag{4}$$

where r is a random value between 0 and 1. In V-shaped transfer functions, a feature can be updated using Equation 6, depending on the probability values given in Equation 5 [36]:

$$T(x_j^i(t)) = \mid tanh(x_j^i(t)) \mid \tag{5}$$

$$x_j^i(t+1) = \begin{cases} \neg x_j^i(t), & r < T(x_j^i(t+1)) \\ x_j^i(t), & otherwise \end{cases} \tag{6}$$

where the $\neg$ operator negates the current value.

3.5. The main activities of a metaheuristic algorithm

Figure 8 gives the generic flowchart diagram of the main activities performed by metaheuristic algorithms. First, an initial population is created and the fitness values of the candidates are calculated. Later, the iterations start. Given a termination condition, new candidate solutions are generated by the exploration and exploitation operators of the metaheuristics. It is important not to evaluate the same solutions repeatedly during the optimization. Because it is highly possible that the recombination operators of the metaheuristics are going to generate the same candidates repeatedly, and there is no need to spend time to recalculate them. Moreover, since these algorithms are computationally expensive, their faster versions, e.g., parallel

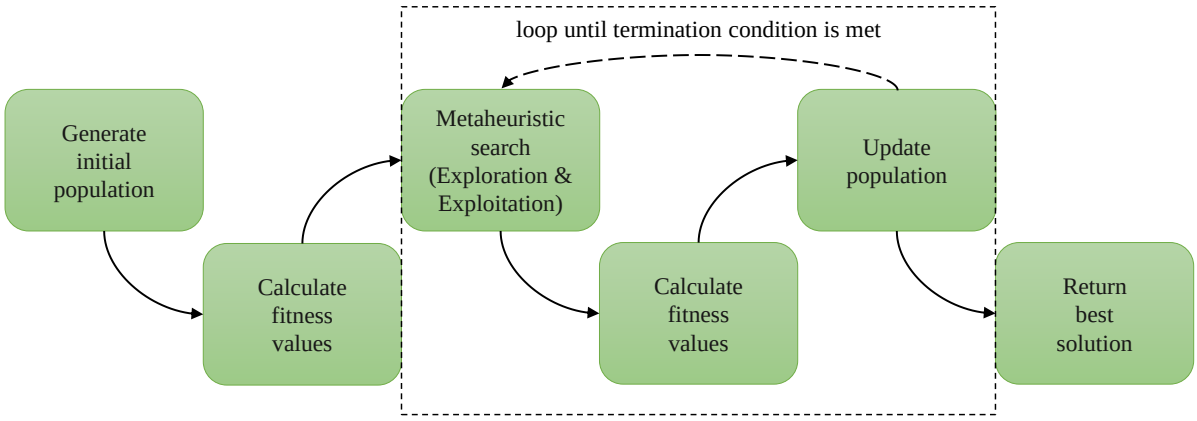

Figure 8: The main steps performed by metaheuristic algorithms during feature selection optimization.

or dynamic programming, can obtain better results due to their increased number of fitness evaluations in less time.

3.6. Setting the parameters of metaheuristics

Controlling the parameters of a metaheuristic algorithm is one of the most important research areas [102]. This has a great impact on the performance of the algorithm. Moreover, the size of the individuals in the population and the number of iterations (generations) should be decided since the metaheuristics are population-based algorithms. The number of iterations, selection method of the parents, mutation ratio, and the convergence ratio are critical parameters that should be well-tuned to provide better solutions in terms of computation time and solution quality. These parameters are common for all population-based metaheuristic algorithms [103].

There are also algorithm-specific parameters of the recent metaheuristic algorithms. Some of the algorithms report the minimum number of parameters to be tuned for the optimization, whereas there are algorithms with many parameters to be set. Considering the difficulty of adjusting the parameters, their effects on performance can be observed in many recent articles. Algorithms having fewer number of parameters may be considered better, however, higher number of parameters can help by directing and improving the optimization process in a better way by giving the chance of tuning these possibilities.

3.7. Common classifiers used in feature selection algorithms

In this part, we give brief information about common supervised machine learning techniques used by wrapper feature selection algorithms. Commonly, the datasets used in the experiments are divided into two parts, namely, training and validation parts. Using the cross-validation method, the accuracy of the

classifiers is tested with selected features. The training set trains the classifier and the remaining instances are used to assess the selected features. The K-Nearest Neighbors (KNN) is among the most used classifiers in feature selection algorithms. The KNN is easy to implement and its computational cost is lower than most of the classifiers. The speed of the classifier is a serious criterion while selecting the learning algorithm as thousands of fitness evaluations are performed during the experiments. Better results can be observed with faster machine learning algorithms such as Extreme Learning Machines [104, 105]. SVM can achieve better classification performance, but it is computationally an expensive classifier. Deep learning shows excellent performance on the classification problems, and it is a very hot research area lately [106]. Logistic Regression, Naive Bayes, Random Forest, Artificial Neural Networks, and Optimum Path Forest are other common classifiers used for the feature selection [87]. The classifiers may perform differently in various domains. Therefore, instead of selecting a single classifier, the researchers should test multiple classifiers in their experiments.

3.8. Evaluation metrics

The performances of the algorithms are evaluated depending on the fitness values. As discussed in Section 3.3, fitness value calculation relies on the prediction performance and the number of selected features in the subset. In addition, the execution times of the algorithms also play a non-negligible role.

Accuracy and F-measure are common metrics for measuring the prediction performance, but other metrics also exist. Precision, Recall, Area Under the Curve, Mean Absolute Error, Kappa statistic [107], Correlation Coefficient [108], Root Mean Square Error, Relative Absolute Error, and Root Relative Squared Error [109] are some examples of other metrics. In addition, Carrasco et al. [110] presented a survey on the recent trends of statistical analyses for the computational intelligence algorithms and prepared the statistical background of tests.

4. Selected recent metaheuristics in literature

This section gives brief information about recent metaheuristic algorithms (in alphabetical order) that we have selected/reviewed in this study. A formal description of the metaheuristics, the exploration and exploitation approaches, and some related feature selection studies that have been highly cited are given in this part. Most of them (and their hybrid versions) are reported to obtain competitive results with the classical metaheuristic algorithms.

4.1. Ant Lion Optimization (ALO)

Mirjalili proposed the ALO that is inspired by the hunting actions of ant lions [43]. The ALO algorithm simulates the interactions between ant lions and the prey (e.g. ants) in the trap. In their search for food,

ants move around randomly. Ant lions hunt them with their traps as ants wander. The random movement of ants is modeled as given below:

$$X(t) = \sum_{i=1}^{t} 2r(t_i) - 1 \tag{7}$$

where t is the number of random walk steps (iterations), and $r(t)$ is a random value between 0 and 1.

The optimization mechanism of ALO is based on the following rules. Ants use different random walks for all their dimensions (exploration). Ant lions build traps according to their fitness values. A higher fitness value helps build a better trap. Ant lions with better traps catch ants with a higher probability. When an ant enters a trap, its random walk range is reduced (exploitation), and the ant is pulled under the sand when its fitness value is better than the ant lion's fitness value. Ant lions change their positions and build new traps at the position where their prey is caught.

Emary et al. developed ALO algorithms for wrapper-based feature subset selection task in classification [11]. They indicated that the ALO uses a single operator to find a balance between the exploration and exploitation operations. Binary and transfer functions were used in the algorithms. They compared the proposed ALO algorithms' results with three other optimization algorithms (PSO, GA, and binary BA). They evaluated all these methods over 20 datasets retrieved from the UCI repository. Results showed that the proposed ALO algorithms are effective in search for the optimal feature subset regardless of the initial population generation techniques or other operators used in the algorithms. Wang et al. proposed an ALO algorithm with wavelet SVM for reducing the hyperspectral image using Levy flights to deal with local optima [111]. A new criterion was developed to estimate the classification accuracy. The proposed method outperformed other algorithms and found the optimal solution, and its classification accuracy is provided with fewer bands. Zawbaa et al. presented a chaotic ALO for the feature selection [112]. They proposed a parameter to control the balance between exploration and exploitation. This parameter changes iteratively to limit the random walk of the ants and to reduce the amount of exploration as the optimization gets closer to the optimum result. They tested their method on ten biological datasets to prove that their method generalizes well. They compared their results with PSO and GA algorithms using several quality metrics. Mafarja et al. employed the ALO algorithm for the wrapper-based feature selection [113]. They developed six variants of ALO. They set different transfer functions to each variant for mapping the continuous search space to a discrete one. They tested all variants' performance on 18 UCI datasets. Moreover, they compared the results with other available approaches in the literature.

4.2. Artificial Bee Colony Algorithm (ABC)

ABC was proposed by Karaboga in 2005 [33]. ABC is the most cited metaheuristic algorithm for feature selection among our selected set of algorithms. It is inspired by the social cooperation of bees for

food procurement. In ABC, candidate solutions represent the bees that are searching for food resources, and every solution represents an available food resource [114, 115]. The quality of each solution in ABC indicates the nectar amount of the resource. *Employed*, *onlooker*, and *scout* are the types of bees in the hive. In nature, employed bees exploit a known food source and deliver their information to others by dancing at the hive. After collecting the nectar of a food source, an employed bee becomes a scout bee and starts exploring new food resources. Finally, onlooker bees observe the employed bees' dance and decide the most promising food sources.

The behaviors of a bee colony are computationally represented as follows [116]. First, an initial population (initial set of food sources) is generated by randomly creating solutions in the search space. A new solution within the decision boundaries is produced with the following equation:

$$X_{ij} = X_j^{min} + rand(0,1)(X_j^{max} - X_j^{min}), \qquad \forall i \in SN, \forall j \in D \tag{8}$$

where X_{ij} is the jth dimension value of the new solution (food source) i, SN is the population, and D is the number of dimensions in the problem.

The fitness of every individual in the colony becomes the obtained resources. After the initial population is evaluated, new solutions are produced with the equation given below. In ABC, to produce new solutions, local information is utilized to find the food sources in the neighbourhood of existing food sources:

$$V_{ij} = X_{ij} + \phi_{ij}(X_{ij} - X_{kj}), \qquad \forall k \in BN, k \neq i \tag{9}$$

where V_i is the new food source at a randomly selected dimension j, BN is the set of employed bees, and ϕ_{ij} is a real number between -1 and 1. If the calculated value does not lie within the boundaries, the value is moved to the acceptable range. When the distance between the jth dimension of kth and ith solutions is small, the new solution becomes closer to the ith solution. As the solutions get closer to the optimum, the perturbation of the ith solution gets smaller.

Schiezaro & Pedrini developed an ABC algorithm to investigate and analyze a feature selection to classify different datasets [117]. Many UCI datasets were tested to validate the effectiveness of the proposed method against other relevant approaches available in the literature. Zhang et al. developed a multiobjective feature selection ABC algorithm, and two new operators were proposed for obtaining a group of non-dominated feature subsets with good distribution and convergence [118]. The algorithm was evaluated on UCI datasets and was compared with multiobjective algorithms. Results showed that the algorithm is robust for solving feature selection. Rao et al. developed an ABC algorithm to eliminate redundant features according to their contribution to the decision making [119]. The algorithm was verified with datasets from the public data repository. Mohammadi & Abadeh proposed a new ABC-based feature selection technique for blind steganalysis which detects images from cover images [120]. Wang et al. [121] developed an ABC algorithm

for filtering the redundant information to obtain the best-predetermined thresholds. Fuzzy SVM and Naive Bayes classifiers were used on datasets. Results of the experiments verified that the proposed method achieves better accuracy values than many representative feature selection techniques.

4.3. Artificial Fish Swarm Algorithm (AFSA)

AFSA metaheuristic was proposed by Li in 2003 [32]. It imitates the fish behaviors of swarming and preying. In AFSA, every fish is a possible solution in the solution space. The behavior of the fish is determined by two factors: its current state and local environmental state, i.e. its companion fish. Therefore, the fish affect and are affected by their environment.

In AFSA, every AF inspects its current environment and moves to a better state with the help of visual cues. Let $X = (x_1, x_2, ..., x_n)$ be the current state of an AF where n is the number of features in the problem domain, and Y is the value of the objective function (fitness value). AFSA is computationally based upon the following behaviors (random moving, preying, swarming, and following):

Random moving: In nature, fish moves randomly in its visual distance, V, to search for food or members of the colony. A random moving to a state, X_j, from the current state, X_i, is denoted as follows:

$$X_j = X_i + rand().V \tag{10}$$

where $rand()$ is a function to generate numbers between 0 and 1.

Preying: A fish selects a random state, X_j, in its visual distance, V, and it moves to that state if the food amount is higher in that direction. If it cannot find a better state after multiple attempts, it chooses a random state in its largest step length, S, to move forward. The random preying behavior of the fish can be increased by decreasing the predetermined attempt count. Preying behavior of an AF is described as given in the equation below:

$$X_i^{next} = \begin{cases} X_i + rand().S.\frac{X_j - X_i}{\|X_j - X_i\|}, & Y_i < Y_j \\ X_i + rand().S & otherwise \end{cases} \tag{11}$$

Swarming: The fish always tries to move to the centre of the colony for many reasons, such as avoiding dangers. Let X_c be the centre of the adjacent fish inside the visual distance of the fish X. The fish moves to the centre of the adjacent fish if the food amount for every fish in the centre area, n_f, is higher than the amount in the current state. Otherwise, the fish continues with the preying behavior. The swarming behavior of an AF is described as given in the equation below:

$$X_i^{next} = \begin{cases} X_i + rand().S.\frac{X_c - X_i}{\|X_c - X_i\|}, & \frac{Y_c}{n_f} > \delta.Y_i \\ prey\ (11) & otherwise \end{cases} \tag{12}$$

where δ is the crowd factor. It ensures that AFs cluster at the best possible state and move to the global optimum.

Following: The fish explores the adjacent fish, and it follows them when they find a better food source. It is denoted the same as the swarming behavior. The only difference is the fish checks the adjacent fish instead of the centre of visual distance. If the food amount is high and the neighbourhood is not crowded, then it moves to that direction to reach the food.

Manikandan & Kalpana proposed an AFSA algorithm for feature selection tasks as AFSA has proven to be highly efficient in solving combinatorial optimization problems [122]. The experiment results showed that the proposed method achieves good performance in finding the best subset of features. Nalluri et al. developed an AFSA algorithm with SVM that finds the most valuable subset of features in a dataset [123]. They evaluated their approach on datasets having binary- and multi-labelled classes. The experiment results showed that the proposed approach increases the classification accuracy while lowering the number of features. Zhang et al. proposed a novel AFSA algorithm that builds a neural network and performs feature selection while tuning the parameters [124]. The results of the experiments presented that the proposed algorithm achieves comparable results with a complex neural network that uses all features as input. Moreover, it requires fewer features and hidden nodes.

4.4. Bacterial Foraging Optimization (BFO)

Passino proposed the BFO metaheuristic in 2002 [31]. The foraging behavior of bacteria is simulated by BFA. The locomotion (movement) is made by a number of tensile flagella that assists an E.coli bacterium to swim. The flagella behave independently, and the bacterium rolls. The bacterium may tumble drastically to acquire a nutrient gradient. The bacterium can travel longer distances in a friendly place. The bacteria can generate a replica of themselves. A swarm of bacteria can migrate to other places.

Let Θ be a multidimensional vector, and $J(\Theta)$ is the optimization problem. BFO uses swarming, reproducing, chemotaxis, and elimination-dispersal behaviors to solve a combinatorial optimization problem [125].

Chemotaxis: The bacteria move either by swimming or tumbling with flagella. Let $\Theta^i(j,k,l)$ is the ith bacterium at the jth chemotactic step, kth reproduction step, and lth elimination-dispersal step. The chemotaxis movement is computationally represented as follows:

$$\Theta^i(j+1,k,l) = \Theta^i(j,k,l) + C(i)\frac{\Delta i}{\sqrt{\Delta^T(i)\Delta i}} \tag{13}$$

where $C(i)$ is the step size of the bacterium and Δi indicates the direction vector.

Reproduction: The population is sorted according to the health status of bacteria. The unhealthy half die, and the remaining ones split into two to fill up the population again. The health of a bacterium, i.e., the fitness value, is denoted as follows:

$$\sum_{j=1}^{N_c} J(i,j,k,l) \tag{14}$$

where N_c is the chemotaxis step size, and $J(i,j,k,l)$ is the fitness value evaluated after each chemotaxis step.

Swarming: Bacteria release a chemical that helps others to group and move to the nutrient gradient. BFO simulates this cell-to-cell signalling which helps to increase the search space of the algorithm.

Elimination-dispersal: Changes in the environment, such as high temperature, may kill a bacteria group or make them disperse to a new location. To mimic this behavior in BFO, after a couple of reproduction operations, some bacteria are killed and moved to some other place. This operation helps the algorithm to leave the local optima and increase the search space.

Wang et al. proposed a set of new BFO algorithms for feature selection with control mechanisms and population updating techniques [126]. The algorithms use three parameters to regulate the diversity of the population and decrease the computational complexity. The studies indicated that the BFO outperformed other algorithms achieving higher classification accuracy. Pal et al. used a BFO and Learning Automata to decide the minimum number of features in an electroencephalography brain-computer interfacing dataset [127]. The authors classified the dataset by Distance Likelihood Ratio Test. The algorithm showed 80.291% prediction accuracy. Niu et al. designed a multiobjective feature selection problem and developed a multiobjective version of the BFO algorithm with KNN as the classifier [128]. The wheel roulette selection was introduced to handle the duplicated features. Various information exchange techniques were combined with the BFO to avoid local optima and achieve better solutions. Comparative experiments with many other evolutionary algorithms verified the performance of the proposed BFO algorithm in feature selection problems.

4.5. Bat Algorithm (BA)

The BA metaheuristic was developed by Yang in 2010 [38]. It is inspired by the echolocation (sonar) technique used by bats while avoiding obstacles, detecting prey, and locating their nests. The sound emitted by bats reflects from the items in the environment, and bats can detect the differences easily. Echolocation can be formalized as a technique to optimize an objective function [129].

Bats have a velocity v_i at position x_i with a frequency range between f_{min} and f_{max}. The bats can set the frequency of the emitted pulse. The wavelength/frequency has a great impact on the convergence of the BA. Higher frequencies can reach short distances and have short wavelengths. The opposite is valid for the lower frequencies. In BA implementation, the frequency can be adjusted to increase the efficiency of the algorithm. The locations of bats x_i and their velocities v_i in a d-dimensional domain change iteratively. For the timestamp t, x_i^t and v_i^t are updated with the equations given below:

$$f_i = f_{min} + (f_{max} - f_{min})\beta \tag{15}$$

$$v_i^t = v_i^{t-1} + (x_i^t - x_*)f_i \tag{16}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{17}$$

where β is a random vector between the range [0, 1] and x_* is the available best solution. During the local search, a random solution is generated using a random walk with the formula given below:

$$x_{new} = x_{old} + \alpha A^t \tag{18}$$

where α is a random number between -1 and 1, and A^t is the average loudness at timestamp t of all the bats in the population.

Rodrigues et al. presented a binary wrapper BA for feature selection [130] . A new methodology was developed to estimate the quality of smaller feature sets. Experiments on public datasets showed that the approach improves classification effectiveness. Jeyasingh & Veluchamy proposed a modified BA for feature selection for breast cancer datasets [131]. The BA used random sampling and achieved better performance. Taha et al. developed a hybrid BA with a Naive Bayes classifier for feature selection [132]. The results verified that the BA outperformed other algorithms in selecting the minimum number of features while keeping the accuracy of classification. Nakamura et al. leveraged the exploration skills of the bats with the speed of the Optimum Path Forest classifier and developed a new BA for feature selection [133]. Experiments verified that the proposed BA could improve the efficiency of Optimum Path Forest and outperform other metaheuristic techniques.

4.6. Biogeography-Based Optimization (BBO)

The BBO metaheuristic was developed by Simon in 2008 [34]. According to the BBO, each individual lives in a habitat, and it has a Habitat Suitability Index (HSI) value that shows the fitness of the individual. Suitability Index Variable (SIV) is a search parameter that characterizes the habitat. The HSI is defined by the SIVs.

Migration and mutation are the operators of the BBO. As the number of species increases in the habitat, the immigration rate decreases and the emigration rate increases as the species start exploring other residences. Exactly S species exist in the habitat according to the probability calculation given below:

$$P_s = \begin{cases} -(\lambda_s + \mu_s)P_s + \mu_{s+1}P_{s+1}, & S = 0 \\ -(\lambda_s + \mu_s)P_s + \lambda_{s-1}P_{s-1} + \mu_{s+1}P_{s+1}, & 1 \leq S \leq S_{max} - 1 \\ -(\lambda_s + \mu_s)P_s + \lambda_{s-1}P_{s-1}, & S = S_{max} \end{cases} \tag{19}$$

where μ is the emigration rate, λ is the immigration rate, and S_{max} is the maximum number of species that the habitat can contain.

Each solution is updated with regard to the habitat modification probability. If a solution S_i is to be modified, its immigration rate is utilized to decide which SIV will be modified. Then, using the emigration rate of other solutions, an SIV migrates to the selected solution S_i. The habitat's HSI can be updated with mutations also. Mutation gives a chance to both the high and low HSI solutions to explore a different search space and improve themselves.

Albashish et al. developed a hybrid metaheuristic BBO with SVM [134]. The proposed method was evaluated on benchmark datasets. The experiment results revealed the high potential of the new algorithm. Liu et al. proposed a discrete BBO to select the best subset of informative genes related to the classification [135]. The Fisher-Markov Selector was used in the algorithm. In addition to this, new mutation and migration operators were proposed to set the balance between exploration and exploitation. Comparison with GA, PSO, DE algorithms and hybrid BBO showed that the proposed algorithm is comparable with others.

4.7. Butterfly Optimization Algorithm (BOA)

Arora & Singh proposed the BOA in 2019 [50]. It simulates the search for food or mating behavior of butterflies. Butterflies emit some fragrance to attract the other butterflies. Each butterfly moves toward the best possible butterfly with more fragrance or moves randomly. The butterflies emit fragrance using Equation 20.

$$f = cI^a \tag{20}$$

where f is the received quantity of fragrance, c is the sensory modality, I is the intensity of the stimulus, and a is the power exponent dependent on modality.

The butterflies are located randomly with their fragrance. At each iteration, butterflies move to new positions to search for food or mate. There are two phases for search operation: global and local. The butterfly can take a step toward the best available butterfly g^* as given in the equation below:

$$x_i^{t+1} = x_i^t + (r^2 \times g^* - x_i^t) \times f_i \tag{21}$$

where x_i^t is the ith butterfly (solution) in iteration t and r is a random number between 0 and 1. Similar to global search, local search is formulated as follows:

$$x_i^{t+1} = x_i^t + (r^2 \times x_j^t - x_k^t) \times f_i \tag{22}$$

If j and k belong to the same group, then Equation 22 defines a local random walk. The search for food and mating partners can happen local and global levels. BOA algorithm uses a switch probability p to swap between global and local search. The iterations continue until the stopping criterion is satisfied.

Arora & Anand developed variants of the BOA for selecting the optimal feature subset [136]. The algorithm uses a threshold function to move in the discrete space of the problem. The results confirmed the efficiency of the proposed algorithms as they can explore a near-optimal subset of features. Sadeghian et al. proposed Information Gain binary BOA for the feature selection problem [137]. The algorithm was able to eliminate a high amount of irrelevant and redundant features. The experimental results were performed on datasets retrieved from the UCI repository. The results confirmed the efficiency of the proposed method selecting the best subset of features. Alweshah et al. proposed a recent monarch BOA that uses the KNN [138] . Experiments on benchmark datasets showed that the algorithm could give a high prediction accuracy for all datasets while reducing the number of features significantly.

4.8. Crow Search Algorithm (CSA)

Askarzadeh proposed CSA in 2016 [46]. The crows are intelligent birds that have large brains. They can remember faces, communicate in different ways, and remember their hiding places. They live in flocks, memorize their hiding places, follow each other for searching food. The CSA simulates the behavior of crows that store food in secret locations and get it back when necessary. The CSA is observed to obtain good results compared to other algorithms on constrained engineering problems.

In CSA, every crow knows its secret location for hiding foods. The secret location refers to the best solution that a particular crow could find so far, denoted with $m^{i,iter}$ for crow i at iteration $iter$. Throughout the iterations, crows search for new food sources. At some point, crow j may decide to see its hiding place, i.e., $m^{j,iter}$. Another crow, crow i, may spot crow j and decide to chase it to learn its hiding place for pilfering. If crow j notices crow i, it will fly to a random position to mislead its follower. Accordingly, the position of crow i will be updated as given below:

$$x^{i,iter+1} = \begin{cases} x^{i,iter} + r_i \times fl^{i,iter} \times (m^{j,iter} - x^{i,iter}), & r_j \geq AP^{j,iter} \\ a\ random\ position, & otherwise \end{cases} \tag{23}$$

where r_i and r_j are two random numbers between 0 and 1, $fl^{i,iter}$ is the flight distance of crow i, and $AP^{j,iter}$ is crow j's awareness probability. Small values of fl are for local search, and large values are for global search.

Ouadfel & Abd Elaziz deals with the premature convergence problem of the CSA [139]. In their study, they proposed an enhanced version of CSA for feature selection. Moreover, they developed an adaptive

awareness probability to improve the balance between exploration and exploitation. In addition, they proposed a novel global search method to improve the exploration talent of the CSA. The obtained results on UCI datasets showed a better speed for convergence. Sayed et al. proposed a chaotic CSA to optimize the feature selection problem [140]. The performance of the algorithm was compared with recent optimization algorithms and observed to be superior to classical CSA and the other algorithms. The experiments showed that the performance of CSA could be improved with a sine chaotic map. Gupta et al. presented a modified CSA for feature selection [141]. They compared their obtained results with the bat, CSA, and modified whale optimization algorithms. The proposed algorithm outperformed the standard bat algorithm and CSA. De Souza et al. proposed a V-shaped binarization of the CSA. They reported encouraging results on benchmark datasets [142] .

4.9. Cuckoo Search (CS)

Yang proposed CS in 2009 [37]. The CS imitates the cuckoo birds with the Lévy flights. Cuckoo birds leave their eggs into the nests of other birds. Only a single egg can be left, and a cuckoo bird can leave its egg into any nest. Nests with high-quality eggs continue to live in the next generation. The probability of detecting an egg by the host bird is $p_a \in [0, 1]$. In case it is detected, either the egg is thrown away or the bird leaves the nest to set up a new nest. The CS does not have many parameters to be set when compared with other metaheuristics. It can be applied to a wide set of optimization problems [143–145]. Shehab et al. prepared a comprehensive review about the CS [146]. The architecture of the CS and its variants were surveyed in their study.

When generating new solutions (eggs) in the CS, a Lévy flight is utilized as given below:

$$x_i^{t+1} = x_i^t + \alpha \oplus L\acute{e}vy(\lambda) \tag{24}$$

where x_i is a solution, and α is the step size. The equation provides a global random walk. The product with Lévy flight ensures efficiency during the exploration. The random steps are generated by a Lévy distribution with infinite variance as given below:

$$L\acute{e}vy \sim u = t^{-\lambda} \qquad (1 < \lambda \leq 3) \tag{25}$$

Pandey et al. developed a binary binomial CS for feature selection in 2020 [147]. The proposed method tries to minimize the number of selected features while maximizing the classification accuracy. The method's performance was compared with binary CS, GWO, GSA, BA, and SA. Abd & Mohamed proposed a modified CS with rough sets to deal with feature selection [148]. The algorithm was tested on several benchmark datasets retrieved from the UCI repository and compared with the existing algorithms on discrete datasets. KNN and SVM were used to evaluate the proposed approach's performance. The proposed algorithm

could significantly improve the classification performance. Rodrigues et al. proposed a binary CS [149]. The experiments were carried out for the detection of thieves in power distribution systems. The results demonstrated the robustness of the algorithm.

4.10. Dragonfly Algorithm (DA)

Mirjalili proposed the DA that is inspired by the behaviors of dragonflies [47]. The dragonflies' static and dynamic swarming behaviors resemble the exploration and exploitation phases of an optimization process. In a static swarm, dragonflies move to different directions in sub-swarms (exploration), and in a dynamic swarm, they move to one direction in bigger swarms (exploitation). Collision avoidance (separation), velocity matching of individuals (alignment), and the tendency towards the centre of the swarm (cohesion) are the main activities of the DA. Other than these, as in every swarm, individuals aim to move towards the food sources and move away from the enemies. The separation is computationally modeled as below:

$$S_i = -\sum_{j=1}^{N} X - X_j \tag{26}$$

where X is the current position of the dragonfly, X_j is the jth neighbour, and N is the number of neighbours. Alignment is given as follows:

$$A_i = \frac{\sum_{j=1}^{N} V_j}{N} \tag{27}$$

where V_j shows the velocity of the jth individual. The cohesion is modeled as follows:

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} - X \tag{28}$$

Moving towards to the food is calculated as given below:

$$F_i = X^+ - X \tag{29}$$

where X^+ is the position of the food. Moving away from the enemy is given as follows:

$$E_i = X^- + X \tag{30}$$

where X^- is the position of the enemy. The direction and the next position of the dragonflies are kept in step, Δ, and position, X, vectors, respectively. The dragonflies' positions are updated with these vectors. The step vector is defined as follows:

$$\Delta X_{t+1} = sS_i + aA_i + cC_i + fF_i + eE_i + w\Delta X_t \tag{31}$$

where s is the separation weight, a is the alignment weight, c is the cohesion weight, f is the food factor, e is the enemy factor, w is the inertia weight, and t is the iteration. Accordingly, the position vector is calculated as follows:

$$X_{t+1} = X_t + \Delta X_{t+1} \tag{32}$$

Mafarja et al. proposed a binary version of DA for feature selection [150] . The proposed DA was tested on UCI datasets. The results were compared with those of PSO, GA in terms of classification accuracy and the number of selected attributes. The results showed that the binary DA is very efficient in feature selection. Similarly, Hammouri et al. proposed an enhanced version of DA for the feature selection task [151]. Too & Mirjalili proposed a novel Hyper Learning Binary DA to obtain the optimal feature subset [100]. Sayed et al. developed a variant of DA with chaotic maps for searching iterations of the DA [152]. Chaotic maps can adjust the parameters and accelerate the convergence rate. The algorithm was employed on a Drug bank database. The experiments showed that Gauss chaotic map could significantly boost the performance of DA. Mafarja et al. developed a wrapper DA in 2018 [16]. Eight different transfer functions were tested during the experiments. S and V-shaped transfer functions were used for balancing the exploration and exploitation steps. The S-shaped DA was observed to outperform the classical version of the algorithm.

4.11. Firefly Algorithm (FA)

Yang proposed the FA that is inspired by the fireflies' flashing characteristics [153]. The flashes attract partners or alert the predators. The flashing characteristics are formulated and used as functions to optimize combinatorial problems [154]. Fireflies attract others proportional to their brightness amounts. They move towards the fireflies that are brighter. Fireflies' attractiveness drops as the distance between them increases. When there is no brighter one, the fireflies move randomly. Therefore, the intensity of the light and the attractiveness amount are the main factors for the FA. The brightness at a specific location can be set with a designated function. However, the attractiveness is decided by other fireflies as it depends on the distance and absorption coefficient. The light intensity of a firefly is calculated as follows:

$$I = I_0 e^{-\gamma r} \tag{33}$$

where I_0 is the default light intensity of the firefly, γ is the light absorption constant, and r is the distance. The attractiveness of a firefly, β, is calculated as follows:

$$\beta = \beta_0 e^{-\gamma r^2} \tag{34}$$

where β_0 is the attractiveness amount when r is 0. When a firefly is attracted and moves towards a brighter firefly, its next position is calculated as follows:

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha^t \epsilon_i^t \tag{35}$$

where j is the brighter (higher light intensity) firefly, and r_{ij} is the Cartesian distance between the two fireflies. α and ϵ_i are the randomization parameter and vector, respectively. When β_0 equals 0, the firefly makes a random walk.

Emary et al. developed a system for feature selection using a modified FA [155] . The algorithm provides a good setting between exploration and exploitation to get the optimal solution. It searches the subset of features for (near)-optimal solutions quickly. The proposed algorithm was compared to PSO and GA. Selvakumar & Muneeswaran developed a network intrusion detection system using a FA [156]. The features were applied to KDD CUP 99 dataset. The results verified that fewer features could detect the intrusion with improved accuracy. The proposed algorithm showed promising improvements. Zhang et al. proposed a FA for feature selection [18]. The proposed method prevented premature convergence. The results on datasets showed that the method is competitive to GA, PSO, and FA. Zhang et al. proposed a variant of FA for feature selection in classification [157]. The FA uses SA for local and global solutions. The parameters of chaotic attractiveness and diversion techniques were used to escape from the local optima as chaotic FA has proven to be more reliable in terms of finding the global optimum [158].

4.12. Grasshopper Optimization Algorithm (GOA)

The GOA simulates the behavior of grasshopper swarms to solve optimization problems [159, 160]. Millions of grasshoppers can migrate over large distances. For performing exploration, the grasshopper moves abruptly, whereas, for exploitation, they move locally. In GOA, the position of every grasshopper denotes a candidate solution. A grasshopper's position is mathematically modeled as follows:

$$X_i = S_i + G_i + A_i \tag{36}$$

where X_i is the location of the ith grasshopper, S_i is the social interaction, G_i is the gravity force, and A_i is the wind advection. For a random walk, the equation becomes $X_i = r_1 S_i + r_2 G_i + r_3 A_i$ where r_1, r_2 and r_3 are random numbers between 0 and 1.

The first of the three main components of GOA, social interaction, is calculated as follows:

$$S_i = \sum_{j=1, j\neq i}^{N} s(d_{ij})\widehat{d_{ij}} \tag{37}$$

where N is the population size, s is a function that gives the strength of social forces, d_{ij} is the distance between X_i and X_j, and $\widehat{d_{ij}}$ is the unit vector for the distance. The strength of social forces function is given below:

$$s(d) = fe^{\frac{-d}{l}} - e^{-d} \tag{38}$$

where f is the intensity of attraction and l is the attractive length scale. The second component, gravity force, is calculated as given below:

$$G_i = -g\widehat{e_g} \tag{39}$$

where g is the gravitational constant, and $\widehat{e_g}$ is the vector to the centre of the earth. Finally, the third component, wind advection, is calculated as given below:

$$A_i = u\widehat{e_w} \tag{40}$$

where u is the constant drift and $\widehat{e_w}$ is the vector in the wind direction.

Mafarja et al. employed GOA with new selection operators and population dynamics for the feature selection task [161]. Moreover, they utilized tournament and roulette wheel selection methods in their implementation. Experiments performed on UCI datasets demonstrated the superiority of the proposed algorithms when compared with other methods. Zakeri and Hokmabadi proposed a GOA-based feature selection method by using the grasshoppers' simulations in finding food sources [162]. Mafarja et al. proposed binary versions of GOA for feature selection [15]. They applied sigmoid and V-shaped transfer functions to the developed algorithms. Moreover, they employed a mutation function to improve the exploration phase of the algorithm. The comparative results on UCI datasets showed that the proposed algorithms outperform other similar algorithms in the literature.

4.13. Gravitational Search Algorithm (GSA)

Rashedi et al. proposed the GSA in 2009. GSA is designed based on the laws of gravity and motion [36] . In the algorithm, the solutions are objects, pulling each another by the force of gravity. The success of the objects is related to their masses. The heaviest object is the best solution, and the lightest object is the worst solution. The gravitational force between the objects determines the next positions of the objects. In GSA, each object attracts all others with a force relative to their masses and the distance. All the objects move to a direction with respect to the applied cumulative force. As similar to nature, lighter objects move faster than heavier objects when applied with the same amount of force. In GSA, the fast movement of the lighter objects is considered as the exploration phase, and the slow movement of the heavier objects is the exploitation phase. Similarly, the attraction force reduces as the distance between the objects increase.

In GSA, the attraction force from an object j to the object i is calculated as follows:

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) * M_{aj}(t)}{R_{ij}(t) + \varepsilon}(X_j^d(t) - X_i^d(t)) \tag{41}$$

where X_i is the position of object i, d is the dimension, t is the iteration, M_{pi} is the passive gravitational mass of object i, M_{aj} is the active gravitational mass of object j, $G(t)$ is a gravitational constant with diminishing values through iterations, $R_{ij}(t)$ is the distance between the two objects, and ε is a very small constant value to eliminate a possible division by zero error when the two objects are at the same position.

Accordingly, the total amount of force applied to object i and its acceleration are calculated as follows:

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand * F_{ij}^d(t) \tag{42}$$

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}^d(t)} \tag{43}$$

where $rand$ is a random value between 0 and 1 to obtain a stochastic algorithm, and M_{ii} is the inertial mass of object i. Now it is possible to update the velocity and the new position of the objects as given below:

$$v_i^d(t+1) = rand * v_i^d(t) + a_i^d(t) \tag{44}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{45}$$

Finally, the gravitational and inertial masses of the objects are updated as given below:

$$M_i = M_{ai} = M_{pi} = M_{ii}, \forall i \tag{46}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{47}$$

where $m_i(t)$ is the normalized fitness value of the object i at iteration t.

Taradeh et al. proposed a GSA based algorithm for feature selection by employing new mutation and crossover operators [95]. The KNN and Decision Tree classifiers were implemented during the experiments. The algorithm was compared to GA, PSO, and a recent GWO. The comparisons showed the high performance of the proposed algorithm. Papa et al. proposed a GSA model for feature selection [163]. In the experiments, they used the Optimum-Path Forest classifier on image classification and fraud detection. The algorithm performed better than Principal Component, Linear Discriminant Analysis and a PSO. Xiang et al. proposed a hybrid GSA model to boost the classification accuracy using feature subset selection [164]. Their algorithm uses chaotic maps to improve the performance of local search with sequential quadratic programming. Experiments on UCI datasets verified the performance of the GSA. Nagpal et al. explored GSA with KNN for feature selection on medical datasets having a huge number of features [165]. Experiments showed that the number of features was reduced by 66%, and the accuracy of prediction was improved.

4.14. Grey Wolf Optimization (GWO)

Mirjalili et al. proposed the GWO in 2014 and its multiobjective version in 2016 [42, 166]. The grey wolves are predator animals, and they live as a pack. In general, a pack consists of 5-12 grey wolves, and the wolves live with a social structure. In this social structure, the members of the pack are referred to as *alpha*, *beta*, *omega*, or *subordinates*, with respect to their dominance factor over others. Every pack have a single alpha wolf, i.e., the most dominant wolf in the pack, hence, the leader. Accordingly, the alpha has most of the responsibilities. Beta is the second most dominant wolf. It is expected to be the alpha wolf in the future. The beta helps the alpha make decisions, conveys the alpha's commands to the pack, and sees them through. Omega is the lowest rank wolf in the pack that is dominated by all other wolves. All remaining wolves are referred to as subordinate or delta. In addition to the social structure, the grey wolves hunt as a pack. When the pack is nearby a prey, they track and chase it. Whenever possible, they encircle the prey and attack in respective order.

GWO is modeled considering the nature of the grey wolves. The social hierarchy and prey hunting strategies constitute the mathematical model of GWO. In GWO, the best three solutions are determined as the *alpha*, *beta*, and *delta* wolves. The other members in the pack are considered *omega* wolves, and they have no contribution to the decision-making process of the next iteration. Given these statements, the encircling activity of the wolves is modeled as follows:

$$\overrightarrow{D} = |\overrightarrow{C} \cdot \overrightarrow{X}_p(t) - \overrightarrow{X}(t)| \tag{48}$$

$$\overrightarrow{X}(t+1) = \overrightarrow{X}_p(t) - \overrightarrow{A} \cdot (\overrightarrow{D}) \tag{49}$$

where $\overrightarrow{A}$ and $\overrightarrow{C}$ are vectors of coefficients used for exploitation and exploration, respectively. t is the iteration, X_p is the prey, $\overrightarrow{X}$ is the position vector of the grey wolf, and $\cdot$ is the multiplication of elements. The coefficient vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ are calculated with Equations 50 and 51, respectively.

$$\overrightarrow{A} = 2 \cdot \overrightarrow{a} \cdot \overrightarrow{r_1} - \overrightarrow{a} \tag{50}$$

$$\overrightarrow{C} = 2 \cdot \overrightarrow{r_2} \tag{51}$$

where $\overrightarrow{r_1}$ and $\overrightarrow{r_2}$ are two randomly filled vectors in the interval [0,1]. $\overrightarrow{a}$ is also a vector with identical elements. To simulate the encircling behavior, the value of its elements gradually diminishes from 2 to 0 through iterations. Accordingly, the vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ has elements in the ranges of [-a, a] and [0, 2], respectively.

Since the location of the prey ($\overrightarrow{X}_p(t)$), i.e., the global optimum, is unknown in the abstract search space, Equations 48 and 49 are approximated with the locations of the *alpha*, *beta*, and *delta* wolves. Accordingly, the positions of the wolves are updated as follows:

$$\begin{aligned} \overrightarrow{D}_\alpha &= |\overrightarrow{C}_1 \cdot \overrightarrow{X}_\alpha - \overrightarrow{X}| \\ \overrightarrow{D}_\beta &= |\overrightarrow{C}_2 \cdot \overrightarrow{X}_\beta - \overrightarrow{X}| \\ \overrightarrow{D}_\delta &= |\overrightarrow{C}_3 \cdot \overrightarrow{X}_\delta - \overrightarrow{X}| \end{aligned} \tag{52}$$

$$\begin{aligned} \overrightarrow{X}_1 &= \overrightarrow{X}_\alpha - \overrightarrow{A}_1 \cdot (\overrightarrow{D}_\alpha) \\ \overrightarrow{X}_2 &= \overrightarrow{X}_\beta - \overrightarrow{A}_2 \cdot (\overrightarrow{D}_\beta) \\ \overrightarrow{X}_3 &= \overrightarrow{X}_\delta - \overrightarrow{A}_3 \cdot (\overrightarrow{D}_\delta) \end{aligned} \tag{53}$$

$$\overrightarrow{X}(t+1) = \frac{\overrightarrow{X}_1 + \overrightarrow{X}_2 + \overrightarrow{X}_3}{3} \tag{54}$$

Emary et al. proposed a binary version of GWO to extract the optimal features for data classification [12]. Similarly, Al-Tashi et al. proposed a binary variant of the GWO combined with PSO for the feature selection task [167]. Tu et al. proposed an ensemble GWO for feature selection [168]. The proposed algorithm achieved higher accuracy and better convergence speed than other variants of GWO. It was verified that the algorithm is reliable for real-world optimization problems. Recently, Hu et al. proposed another binary variant of GWO by mapping the transfer functions to binary representations [169]. The algorithm had a good convergence speed and implemented feature selection on the UCI datasets successfully with small deviations. Chantar et al. studied feature selection in the natural language processing domain [170]. They proposed an enhanced GWO for text classification problems. The proposed algorithm with an elite crossover scheme improved efficacy when compared to other algorithms.

4.15. Harmony Search (HS)

Geem et al. proposed the HS in 2001 [30]. It is inspired by the harmony of musical compositions. It has been widely used to solve optimization problems in different domains such as engineering, telecommunications, and health [171]. The HS formalizes the methods that musicians use when they are improvising. These methods are playing a known piece, playing a piece similar to a known piece, and playing a random piece [172]. In HS, these components correspond to harmony memory usage, pitch adjusting, and randomization.

The harmony memory ensures that the best individuals (harmonies) in the population are carried over the generations. The amount of best individuals to be kept in the memory is decided with the harmony memory's acceptance rate, $r_{accept} \in [0, 1]$. When r_{accept} is small, slower convergence is provided with small memory. As the acceptance rate gets closer to 1, almost all harmonies are kept in the memory, leaving no

exploration space to the algorithm. Therefore, the parameter r_{accept} is kept between [0.7, 0.95] to provide a balance.

The pitch adjustment provides diversity in the population. It resembles the mutation operation in genetic algorithms. The existing pitch x_{old} is adjusted, and a new pitch x_{new} (candidate solution) is generated as follows:

$$x_{new} = x_{old} + b_{range} * \varepsilon \tag{55}$$

where b_{range} is the pitch bandwidth, and ε is a random number between -1 and 1. To change the space amount for exploration, pitch-adjusting rate r_{pa} can be used. Slow convergence can be provided by lowering this parameter. The probability of pitch adjustment becomes as follows:

$$p_{pitch} = r_{accept} * r_{pa} \tag{56}$$

The randomization increases diversity in the population by providing local search. The probability of this operation is given below:

$$p_{random} = 1 - r_{accept} \tag{57}$$

Gholami et al. proposed an HS effective metaheuristic algorithm to solve the feature selection problem [173]. Ramos et al. developed an HS algorithm combined with the Optimum-Path Forest classifier for feature selection [174]. The experiments were performed to identify non-technical problems in power distribution infrastructures. Inbarani et al. presented a hybrid HS algorithm using Rough Set Quick Reduct for feature selection [175]. The number of selected features is decreased significantly during the experiments of classification of medical datasets. Moayedikia et al. introduced a method with symmetrical uncertainty and HS [176]. The method utilizes symmetrical uncertainty to select features according to their dependency on class labels. On micro-array datasets, the algorithm worked better than state-of-the-art algorithms. Diao & Shen proposed a novel HS [177]. The proposed algorithm can escape from local optima and create new solutions according to the stochastic behavior of the HS. Moreover, the authors introduced new parameter setting techniques. The proposed approach was compared with PSO and GA. Wang et al. developed a new HS for improving the performance of email classification [178]. Experiments with fuzzy SVM and Naive Bayesian classifiers were carried out on different corpora. The proposed algorithm outperformed other algorithms.

4.16. Harris' Hawk Optimization (HHO)

Bairathi and Gopalani proposed the HHO that mimics the cooperative behavior of Harris' Hawks [179]. Later, Heidari et al. [51] presented an extended version of the HHO. A set of hawks cooperatively chase prey

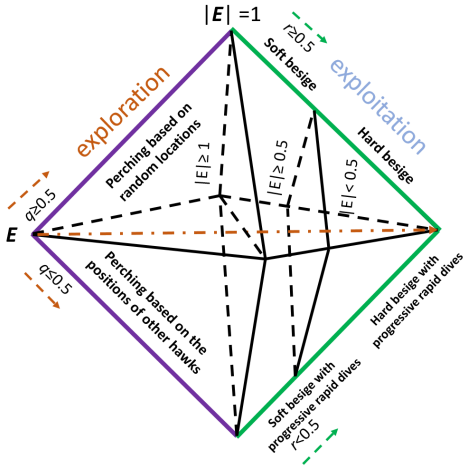

Figure 9: An overview of the exploration and exploitation phases of the HHO metaheuristic.

from diverse directions to surprise it. The HHO executes similar to the dynamic patterns and behaviors of hawks. The exploration and exploitation steps of the HHO are given in Figure 9.

The prey (rabbit) might not be detected easily while the hawks are tracking it. During the optimization process, the hawks observe the hunting site. This is similar to the exploration activity of the optimization algorithms. In HHO, the prey denoted as the available best solution, whereas the hawks are considered as candidate solutions. The hawks randomly perch and try to detect the prey using two main strategies. They perch according to the positions of other hawks and the rabbit or randomly perch as given below:

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1 \left| X_{rand}(t) - 2r_2 X(t) \right|, & q \geq 0.5 \\ (X_{rabbit}(t) - X_m(t)) - r_3(LB + r_4(UB - LB)), & q < 0.5 \end{cases} \tag{58}$$

where $X(t)$ and $X(t+1)$ are the current and next positions of the hawk, respectively. $X_{rabbit}(t)$ is the current location of the rabbit, q, r_1, r_2, r_3, and r_4 are random values between 0 and 1. LB and UB are the boundaries of the variables, $X_{rand}(t)$ is the current position of a randomly selected hawk, and X_m is the average position of the hawks in the population. HHO changes activities from exploration to exploitation according to the energy of the rabbit as it diminishes while escaping. The energy of the rabbit is given as follows:

$$E = 2E_0(1 - \frac{t}{T}) \tag{59}$$

where E is the escaping energy, T is the number of iterations, and E_0 is the initial energy state. E_0 is set randomly between -1 and 1 at each iteration of the algorithm. During exploitation, the hawks use the surprise pounce while the prey is trying to escape. When the escape chance of the prey, r, is higher than

0.5, it cannot successfully escape, and the hawks use hard or soft besiege to get the prey. They surround the prey in many directions according to the energy of the prey. When $|E| \geq 0.5$, the hawks perform soft besiege and when $|E| < 0.5$, they perform hard besiege. The hawks surround the rabbit softly and perform the surprise pounce after the prey is exhausted. This behavior is computationally modeled as follows:

$$X(t+1) = \Delta X(t) - E\,|JX_{rabbit}(t) - X(t)| \tag{60}$$

$$\Delta X(t) = X_{rabbit}(t) - X(t) \tag{61}$$

where $\Delta X(t)$ is the distance between the rabbit and the hawk at iteration t and J is the jump strength of the rabbit. The J is updated randomly to simulate the motions of the rabbit. When the prey is tired, $|E| < 0.5$, the hawks surround the prey to perform the surprise pounce. The positions are updated using the following equation:

$$X(t+1) = X_{rabbit}(t) - E\,|\Delta X(t)| \tag{62}$$

Zhang et al. presented an improved HHO algorithm combined with SalpSA to find the best solutions for feature selection [180]. The proposed HHO showed better performance for balancing the exploration and exploitation phases and faster convergence. Too et al. proposed a binary version of HHO for the feature selection task [181]. Moreover, they enhanced it by proposing a quadratic binary HHO. The experimental results verified the effectiveness of the algorithm. Abdel et al. presented a hybrid feature selection algorithm that combines the HHO algorithm and SA [182]. Similarly, Sihwail et al. proposed an enhanced HHO algorithm for feature selection by utilizing Opposition-based Learning [183]. They evaluated the proposed algorithm on various benchmark datasets and compared the results with many well-known metaheuristic algorithms. Dokeroglu et al. proposed a multiobjective HHO algorithm for the solution of the binary classification problem [184]. The authors reduced the number of features and kept the accuracy prediction as maximum as possible. Logistic Regression, Support Vector Machines, Extreme Learning Machines, and Decision Trees are used to calculate the prediction accuracy. A recent Coronavirus disease (COVID-19) dataset is also tested during the experiments.

4.17. Krill Herd (KH)

Gandomi & Alavi proposed KH in 2012 [40]. The herds of krill can construct large groups in the ocean [185]. When a sea animal attacks a herd, it can eat individuals, but this can only reduce some of the krill, not the herd. The KH has a multiobjective purpose as it increases the population of the herd while searching for food. Each krill moves according to the best solution while looking for the food. When the

attack happens, some individuals are discarded from the herd. Detailed information about KH optimization and its applications are given in a review by Bolaji et al. [186].

The fitness of each individual is calculated using the distance from the densest point of the population and the distance from the food source. The positions of the krills are affected by three actions, namely, movement of other krills (N_i), foraging (F_i), and random movement (D_i). A Lagrangian formula that utilizes these actions is used for the search operation of the optimization algorithm:

$$\frac{dX_i}{dt} = N_i + F_i + Di \tag{63}$$

Individuals move to the other members of the herd as they try to keep density high. The movement for this behavior is modeled as below:

$$N_i^{new} = N^{max}\alpha_i + \omega_n N_i^{old} \tag{64}$$

where N^{max} is the maximum speed, ω_n is the inertia weight of the motion, and N_i^{old} is the last motion. α_i is the direction of motion and calculated by the density of the swarm as follows:

$$\alpha_i = \alpha_i^{local} + \alpha_i^{target} \tag{65}$$

where α_i^{local} is the effect of adjacent krills and α_i^{target} is the effect of the target (best available krill). The neighbour individuals can have an attractive or repulsive effect on each other. The attraction is good for improving exploitation, whereas repulsive behavior provides exploration.

The foraging is controlled by current and the last known food locations. The motion for this behavior is modeled as given below:

$$F_i = V_f \beta_i + \omega_f F_i^{old} \tag{66}$$

where V_f is the foraging velocity and F_i^{old} is the last position of the food source. β_i is the direction of motion and calculated as follows:

$$\beta_i = \beta_i^{food} + \beta_i^{best} \tag{67}$$

where β_i^{food} is the attraction amount of the food and β_i^{best} is the impact amount of the best available krill. These values are estimated based on the fitness values of the individuals.

The random movement (physical diffusion) is calculated using the maximum diffusion speed (D^{max}) and a random directional vector (δ) as given in the formulation below:

$$D_i = D^{max}\delta \tag{68}$$

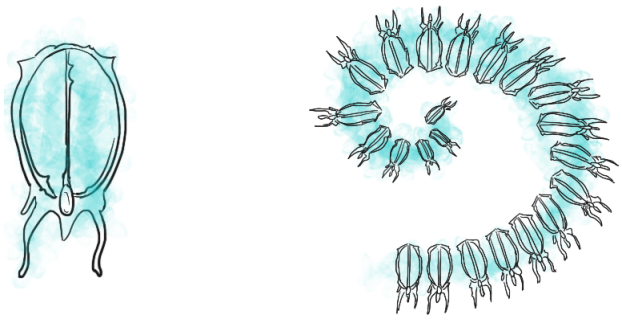

Figure 10: The Salp Swarm chain.

These movements direct the krill to the best possible position, as given in Equation 63. As a result, the position of a krill changes according to the following equation:

$$X_i(t + \Delta t) = X_i(t) + \Delta t \frac{dX_i}{dt} \tag{69}$$

Rodrigues et al. proposed a binary KH and validated its performance for feature selection on many datasets [187]. The experiment results verified that the proposed algorithm outperforms other metaheuristic approaches. Abualigah et al. developed a parallel framework to improve the performance of the KH with the swap mutation for feature selection to tackle the problem of high-dimensional data in text clustering [188]. The K-means was employed to cluster the documents. The method presented alternative methods for the text mining community. Zhang et al. proposed an algorithm with a pre-screening method based on Information Gain and KH algorithm [189]. A tangent function, a transfer factor, and a chaos memory weight factor were used to search the subsets of features. The algorithm could achieve performance improvements in the accuracy of the classification and the number of features.

4.18. Salp Swarm Algorithm (SSA)

Mirjalili et al. proposed the SSA in 2017 [48] . The main inspiration comes from the behavior of salps that navigate and forage in the oceans. In deep oceans, salps form a chain, as can be seen in Figure 10. Scientists believe that this action is realized to obtain better locomotion using fast changes and foraging [190].

The population of salps is comprised of a leader and its followers. The swarm is directed by the leader, and the rest of the population follows each other in the direction of the leader. Given the position of salps and a food source F, the leader's position in a dimension j in the problem search space is calculated as follows:

$$x_j^1 = \begin{cases} F_j + c_1((ub_j - lb_j) * c_2 + lb_j), & c_3 \geq 0 \\ F_j - c_1((ub_j - lb_j) * c_2 + lb_j), & c_3 < 0 \end{cases} \tag{70}$$

where ub_j is the upper bound, and lb_j is the lower bound. As can be seen from the equation, the leader only updates its position according to the position of the food. The coefficient c_1 balances the exploration and exploitation as given below:

$$c_1 = 2e^{-(\frac{4l}{L})^2} \tag{71}$$

where l is the current iteration, and L is the maximum iteration count. On the other hand, c_2 and c_3 indicate the step size and direction of the movement, respectively. They are set as random values between 0 and 1.

The followers' position is calculated by adjusting Newton's law of motion as given below:

$$x_j^i = \frac{1}{2}(x_j^i + x_j^{i-1}), \qquad i \geq 2 \tag{72}$$

where x_j^i is the position of the ith follower in the jth dimension.

Hegazy et al. developed a new SSA to improve the solution quality, reliability and convergence speed of the algorithm for feature selection [191]. They utilized inertia weight in their algorithm when setting the current best solution. The algorithm was compared with basic SSA and recent swarm methods. Experiment results showed that the algorithm had better results than the other optimizers in terms of prediction accuracy and selected features. Tubishat et al. developed an improved SSA for the feature selection problem [192]. The population was initialized using Opposition Based Learning. Moreover, a new local search technique was used to improve exploitation performance. The algorithm was compared with GA, PSO, ALO, and GHO. The experimental results verified the effectiveness of the algorithm. Faris et al. proposed two new SSA variants for the feature selection task [193]. In the first one, they converted the continuous version of the algorithm to the binary version using eight transfer functions. In the other one, a crossover operator was used to enhance the exploratory skills of the algorithm. The proposed approaches outperformed five selected wrapper algorithms on most of the datasets.

4.19. Sine Cosine Algorithm (SCA)

Mirjalili proposed the SCA in 2016, based on the mathematical sine and cosine functions [44]. Random and adaptive variables are applied to find a balance between exploration and exploitation. Therefore, it can effectively converge to the global optimum. In SCA, the update mechanism is designed as follows:

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 \times sin(r_2) \times |r_3 P_i^t - X_i^t|, & r_4 < 0.5 \\ X_i^t + r_1 \times cos(r_2) \times |r_3 P_i^t - X_i^t|, & otherwise \end{cases} \tag{73}$$

where X_i^{t+1} represents the ith dimension of the solution at tth iteration, P_i is the target in ith dimension, r_1 is a number that gradually diminishes from a constant value (e.g. 2) to 0 through iterations, and r_2, r_3, and r_4 are random values. $sin(\cdot)$, $cos(\cdot)$, and $|\cdot|$ denote the mathematical functions *sine*, *cosine*, and *absolute value*, in respective order. The values of r_2, r_3, and r_4 lie the intervals [0, 2π], [0,2], and [0,1], respectively.

Hafez et al. introduced an SCA model for feature selection [194]. They combined accuracy maximization and feature size minimization into a single fitness function. Tests on benchmark datasets verified the performance of their model over PSO and GA. Sindu et al. developed an SCA with an elitism approach [195]. They proposed an update technique to find the best attributes for classification accuracy. Experiment results proved the efficiency of the new algorithm.

4.20. Social Spider Optimization (SSO)

The SSO was proposed by Cuevas et al. in 2013 [41]. The social spiders live in colonies, and each spider may have different tasks, including web design, mating, hunting, and social interaction. The web is a communication means of the colony. The interaction between the spiders is modeled with respect to their biological nature. The search space is modeled as a communal web, and spiders represent solutions. Each spider has a weight that is proportional to its fitness value. Yu and Li proposed a different type of SSO for optimization problems that utilizes different search behaviors [196]. In this survey, we focus on the first implementation of the algorithm.

The spiders share information shared through the web. The collective coordination of the spiders is encoded as vibrations. The weight and distance are the two factors of vibrations. Formally, a vibration transmitted by spider j and received by the spider i is modeled as given below;

$$Vib_{i,j} = w_j \cdot e^{-d_{i,j}^2} \tag{74}$$

where w_j is the normalized fitness value of the spider j, and $d_{i,j}$ is the distance between the two spiders.

In SSO, the majority of the colony (65-90% of the population) is female. The gender of the spiders plays a role in the position update and mating processes. The position update of the female spider i can be modeled as given below:

$$f_i^{t+1} = \begin{cases} f_i^t + \alpha \cdot Vibc_i \cdot (s_c - f_i^t) + \beta \cdot Vibb_i \cdot (s_b - f_i^t) + \delta \cdot (rand - 0.5), & r_m < PF \\ f_i^t - \alpha \cdot Vibc_i \cdot (s_c - f_i^t) - \beta \cdot Vibb_i \cdot (s_b - f_i^t) + \delta \cdot (rand - 0.5), & otherwise \end{cases} \tag{75}$$

where α, β, δ, $rand$, and r_m are random values between [0, 1], PF is a constant threshold value, t is the iteration, $Vibc_i$ is the vibration from s_c, the nearest spider having a higher weight than the spider i, and $Vibb_i$ is the vibration from the heaviest spider in the colony, i.e., s_b.

The position update of male spiders differs with respect to dominance information. Male spiders are considered as dominant (D) if their weight is greater than the weight of the median male and non-dominant (ND) otherwise. Accordingly, the position update of a male spider i is as modeled below:

$$m_i^{t+1} = \begin{cases} m_i^t + \alpha \cdot Vibf_i \cdot (s_f - m_i^t) + \delta \cdot (rand - 0.5), & i \in D \\ m_i^t + \alpha \cdot (\tau - m_i^t), & i \in ND \end{cases} \tag{76}$$

where $Vibf_i$ is the vibration from the nearest female spider, i.e., s_f; and τ is the weighted mean of male spiders.

The mating operator can be applied between dominant males and females. If a dominant male spider has female spiders in its mating range, then they (the dominant male and all the females in its mating range) form a new spider. Each spider involved in this creation has a chance to influence the new breed that is proportional to their weight. If the new spider has a worse fitness value than the worst spider in the colony, then it is discarded. Otherwise, it replaces the worst spider.

Bas & Ulker mentioned that there is not enough research on the SSO-based feature selection [197]. They evaluated S and V-shaped transfer functions. They used KNN and SVM as classifiers. Experiments were performed on UCI datasets, and obtained results verified that the proposed algorithms had superior performance. Ibrahim et al. presented an improved SSO algorithm to deal with the local optima problem of feature selection [198]. The proposed algorithm avoids irrelevant features with the help of Opposition-based Learning. Moreover, it increases the exploration of the search space. Abd & Mohamed proposed an SSO algorithm with a fitness function that depends on the rough set theory [199]. The position of each spider was changed according to the type of spiders. The algorithm was validated on UCI clinical medical datasets. The results verified that the algorithm was superior compared to state-of-the-art swarm algorithms. Pereira et al. addressed the problem of SVM tuning parameters problem by introducing an SSO [200]. Experiments were performed with PSO and HS, and results showed that the SSO is a nice approach for SVM selection.

4.21. Teaching-Learning-Based Optimization (TLBO)

TLBO was proposed by Rao et al. in 2011 [39]. The algorithm has gained attention with respect to its rapid convergence and the lack of algorithmic parameters [201, 202]. Zou et al. provided algorithm insights and application areas in a survey paper [203]. TLBO simulates a realistic education model, where students obtain knowledge from the teacher and their classmates. At every iteration, the algorithm examines the population and selects the best individual as the trainer. All remaining individuals become learners. The trainer shares its information with the learners (exploitation). Then, the learners interact with fellow learners and revise (exploration). In the teaching phase, a new solution is generated according to the equations below:

$$DM_i = r_i(T_i - T_F M_i) \tag{77}$$

$$X_{new,i} = X_{old,i} + DM_i \tag{78}$$

where i is the iteration, and DM_i, T_i, and M_i are the difference mean, teacher, and the mean value of learners at iteration i, respectively. T_F is the teaching factor, and it affects the learning capability. Its value is either one or two. Finally, r_i is a random number between [0, 1].

After the teaching phase is complete, the learner phase begins, i.e., students interact with fellow classmates. For this purpose, each student trains with a randomly selected student. The update of two learners where $X_i \neq X_j$ is given below as:

$$X_{new,i} = \begin{cases} X_{old,i} + r_i(X_i - X_j), & X_i < X_j \\ X_{old,i} + r_i(X_j - X_i), & X_j < X_i \end{cases} \tag{79}$$

Sevinc & Dokeroglu proposed a TLBO feature selection algorithm with Extreme Learning Machines for the feature selection [204]. Tests on UCI datasets showed competitive results with state-of-the-art algorithms. Pradhan et al. developed a modified TLBO for feature selection to predict classes of an enzyme [205]. Kiziloz et al. proposed a set of multiobjective TLBO algorithms for the feature selection in 2018 [206] . The authors carried out comprehensive experiments on well-known datasets to verify the performance of the algorithms.

4.22. Whale Optimization Algorithm (WOA)

Mirjalili et al. proposed the WOA algorithm animated by the hunting behavior of humpback whales in 2016 [45]. These whales are social creatures as they hunt as a group. When they encounter a group of prey, i.e., small fish or krill groups, they blow nets of bubbles and direct their prey into this bubble-net (see Figure 11). This mathematical model of the WOA can construct a new strategy to solve challenging optimization problems. The essential functions of the algorithm include searching for prey, encircling the prey, and spiral bubble-net movements.

The WOA starts with a random solution and employs many strategies. After deciding the best search agent, the other agents update their positions accordingly. They select either the best search agent or a random whale as their target and move towards it. This action is introduced in Equations 80 and 81.

$$\overrightarrow{D} = |\overrightarrow{C}.\overrightarrow{X^*}(t) - \overrightarrow{X}(t)| \tag{80}$$

$$\overrightarrow{X}(t+1) = \overrightarrow{X^*}(t) - \overrightarrow{A} \cdot \overrightarrow{D} \tag{81}$$

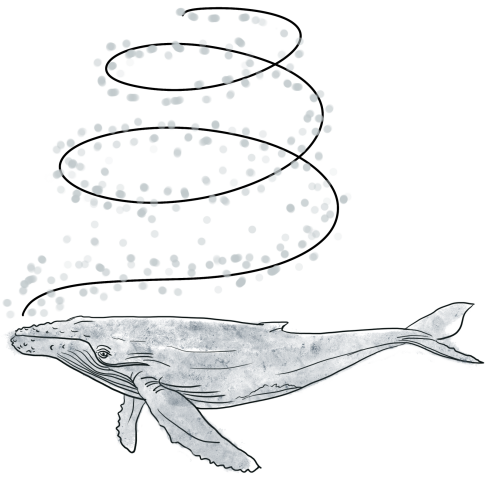

Figure 11: The WOA activities.

where $\overrightarrow{A}$ and $\overrightarrow{C}$ are vectors of coefficients, t is the iteration, X^* is the target whale, $\overrightarrow{X}$ is the position vector, and $\cdot$ is the multiplication of elements. The coefficient vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ are calculated with Equations 82 and 83, respectively.

$$\overrightarrow{A} = 2\overrightarrow{a} \cdot \overrightarrow{r} - \overrightarrow{a} \tag{82}$$

$$\overrightarrow{C} = 2 \cdot \overrightarrow{r} \tag{83}$$

where $\overrightarrow{a}$ and $\overrightarrow{r}$ are two randomly filled vectors. The $\overrightarrow{r}$ vector spans the interval [0,2), whereas the $\overrightarrow{a}$ vector ranges between (-2,2). To simulate the encircling behavior, the effect of $\overrightarrow{a}$ diminishes at each iteration in Equation 82. This avails a spiral position update for the whale, defined as below:

$$\overrightarrow{X}(t+1) = D' \cdot e^{bl} \cdot cos(2\pi l) + \overrightarrow{X^*}(t) \tag{84}$$

where (X, Y) is the position of the whale, (X^*, Y^*) is the position of the prey, $D^{'} = |\overrightarrow{X^*}(t) - \overrightarrow{X}(t)|$ is the distance between the whale and the prey, b is shaped the logarithmic spiral, and l is between [-1,1].

Target selection depends on the value of $|\overrightarrow{A}|$. If $|\overrightarrow{A}| > 1$, then a random whale is selected as the target. On the other hand, the target is set as the best whale when $|\overrightarrow{A}| < 1$.

Sharawi et al. proposed a WOA for feature selection [207]. This algorithm searches for the best and minimal subset of features. The algorithm was compared with the PSO and GA on selected UCI datasets. The results verified that the algorithm has advantages over the other optimizers. Similarly, Mafarja & Mirjalili developed two new binary WOA feature selection variants [17]. The performance of different selection mechanisms was studied in the paper. Moreover, the mutation and the crossover operators were

enhanced. The proposed algorithm was compared to PSO, GA, and ALO. In another study, Mafarja & Mirjalili proposed two hybrid WOA with SA algorithms for feature selection [208]. The SA was used in the exploitation step of the algorithm. The authors showed the efficiency of their proposed approaches with experimental results. Finally, Hussien et al. proposed a binary WOA based on a Sigmoid transfer function [209]. In their study, they used the S-shaped transfer function and applied the KNN classifier.

4.23. Other Recent Metaheuristics

In this section, we give brief information about other metaheuristic algorithms that are not included in our selected metaheuristics since they are still very new and their performances have not been observed in different domains yet.

Pourpanah et al. proposed new Brain Storm Optimization (BSO) algorithm for feature selection [210]. BSO is a metaheuristic inspired by human brainstorming. They combined a recent fuzzy model with BSO in their proposed algorithm. The statistical results indicated that the algorithm produced promising results when compared with PSO, GA, GP, and ACO. Ghaemi & Derakhshi developed a Forest Optimization Algorithm (FOA) for feature selection [211]. The experiments were held on real-world datasets and the results were compared with PSO. The algorithm could improve the accuracy of classification in selected datasets. Chen & Chen developed a wrapper method with cosine similarity based SVM for feature selection in 2015 [212]. The algorithm performs SVM parameter learning and removes redundant features. Rodrigues et al. developed a binary-constrained Flower Pollination Algorithm for feature selection [213]. The algorithm uses a boolean lattice as a search space where each solution defines whether a feature is selected or not. Experiments on datasets with FA, PSO, and HS have shown the high performance of the algorithm. Mirjalili proposed the Moth-flame Optimization Algorithm in 2015 [214]. Moths use a fixed angle considering the moon while moving at night. It is an effective way of traveling long distances. Mirjalili modeled this behavior of the insects to perform a better optimization. The results demonstrated the high performance of the algorithm. Wang et al. developed a Monarch Butterfly Optimization (MBO) for feature selection [215]. The authors demonstrated the performance of the MBO on five other metaheuristic algorithms with benchmark problems. Yan et al. developed a Coral Reefs Optimization algorithm for deciding the best features in 2019 [216]. Tournament selection was used to increase the diversity of individuals in the population. The KNN was employed to evaluate the accuracy of classification. Experiments on public medical datasets showed that the algorithm outperformed other state-of-the-art methods. Alweshah et al. developed an MBO algorithm with a wrapper feature selection method with KNN [138]. Experiments were implemented on benchmark datasets, and the results showed that MBO was superior for all datasets while reducing the selected features. Kiziloz developed a formal comparison of classifier ensemble techniques for the feature selection domain [217]. The author reported that ensemble algorithms can perform better than single classifiers; however, they spend longer optimization times.

Dolphin echolocation is a metaheuristic proposed by Kaveh and Farhoudi [218]. In addition, Colliding Bodies Optimization is another metaheuristic proposed by Kaveh and Mahdavi [219]. Shah-Hosseini proposed the Galaxy-based Search Algorithm to explore the continuous optimization problems [220]. Jain et al. proposed the Squirrel Search Algorithm in 2019 [221]. It simulates the foraging behavior of flying squirrels. The algorithm demonstrated that accurate solutions are possible with a high convergence rate when compared to other existing optimizers. There are many (hundreds of) new but less explored metaheuristics in the literature, which were excluded from this review not to disturb the readers and to prevent a chaotic environment.

4.24. A summary comparison of the selected algorithms

These 22 algorithms are selected from hundreds of metaheuristic algorithms proposed for the last two decades. When comparing the performance of algorithms, one of the most important criteria for us was the number of citations. Although the older algorithms like ABC (2005) seem to be more likely to be cited by years, the GWO (2014), GOA (2018), FA (2009), TLBO (2011), and WOA (2016), although relatively new, have a high number of citations. They can be applied in many domains, and the results make these algorithms come to the fore. Practical and simple operators in the field of exploration and exploitation and the fact that their codes are open to scientists (as a framework) also provide these algorithms with superiority.

In Table 2, we categorize the metaheuristic algorithms in terms of their nature, inspiration, and the existence/use of leaders. Of these 22 algorithms, most of them (17) are based on animal nature. The remaining ones are based on human behaviour (2), science (2), and evolutionary (1). Animal-based algorithms are inspired by the animals' varying behaviours. These behaviours are food search (8), hunting (6), mating (3), and breeding (1). Other algorithms are inspired by observational facts. Finally, we categorize the algorithms based on their search mechanism. Some of the algorithms (9) utilize the knowledge incorporated by the best solution (leader) for enhancing its exploitation capabilities. The mentioned best solution could be either the iteration best (local) or the overall best (global) solution. Accordingly, four algorithms utilize the local leader, one algorithm utilizes the global leader, and another algorithm utilizes both. Of the remaining algorithms, one of them utilizes local best and local worst solutions, whereas another algorithm utilizes their global counterparts, i.e., global best and global worst. The last algorithm, GWO, utilizes three local leaders in its search mechanism. The remaining 13 algorithms do not track the best solution. They explore the search space with random walks.

We analyzed a total of 82 studies that applies these 22 algorithms on the feature selection domain in terms of the used datasets, classifiers, evaluation metrics, and transfer functions [11, 12, 15–18, 95, 100, 111–113, 117–124, 126–128, 130–142, 147–152, 155–157, 161–165, 167–170, 173–178, 180–184, 187–189, 191–195, 197–200, 204–209].

Table 2: Categorization of the metaheuristic algorithms.

Metaheuristic	Nature	Inspiration	Leader
ABC	animal-based	food search	-
AFSA	animal-based	hunting	-
ALO	animal-based	hunting	-
BA	animal-based	hunting	global best
BBO	evolutionary	observational	-
BFO	animal-based	food search	-
BOA	animal-based	food search, mating	randomly local or global best
CS	animal-based	breeding	-
CSA	animal-based	food search	-
DA	animal-based	food search	-
FA	animal-based	mating	-
GOA	animal-based	food search	-
GSA	science-based	observational	local best and local worst
GWO	animal-based	hunting	three local best
HHO	animal-based	hunting	local best
HS	human-based	observational	-
KH	animal-based	food search	global best and global worst
SCA	science-based	observational	-
SSA	animal-based	food search	local best
SSO	animal-based	mating	-
TLBO	human-based	observational	local best
WOA	animal-based	hunting	randomly random walk or local best

In Figure 12, we present the most used datasets by these studies. The Breast Cancer dataset takes the lead in this figure as it is utilized in 61 out of 82 studies (74.39%). It is followed by the Congressional Voting Records, Ionosphere, Connectionist Bench (Sonar, Mines vs. Rocks), and Statlog (Heart) datasets. These datasets are commonly referred to as Vote, Ionosphere, Sonar, and Heart, respectively. It is also clear from the figure that researchers mostly prefer the UCI machine learning repository to retrieve datasets.

Moreover, in Figure 13, we present the most used classifiers. K-Nearest Neighbors is the most preferred classifier in these studies with its 60.98% ratio. Support Vector Machines, Naive Bayes, and Decision Tree

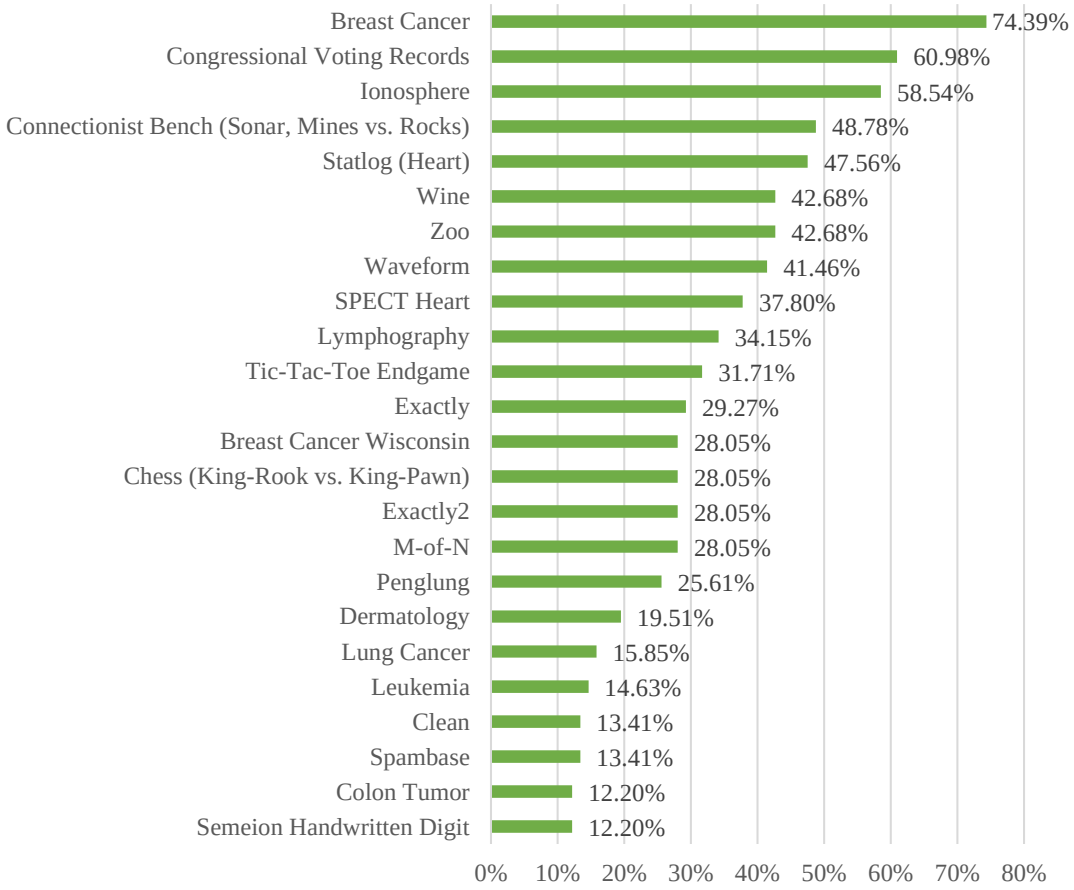

Figure 12: The most used datasets in the analyzed feature selection studies.

follow it.

Furthermore, in Figure 14, we present the most used evaluation metrics. Almost all studies (90.24%) utilize accuracy and fitness value as the performance metric. The other commonly used metrics are the number of selected features and execution time, in respective order.

Finally, details on the use of transfer function in these 82 studies are as follows. Forty-six studies (56.1%) do not use a transfer function. Of the remaining 36 studies, 25 of them (69.44%) utilize S-shaped transfer functions, and 18 of them (50%) utilize V-shaped transfer functions. Two studies (5.5%) use a simple threshold value to represent the real value in the binary domain.

It can be seen from the detailed breakdown of the datasets, classifiers, evaluation metrics, and transfer functions that most feature selection studies verify the efficiency of their new algorithms over simple and well-known scenarios. However, real-world problems are generally more complex. For example, sentiment analysis is a common natural language processing task, and it can be designed as a binary classification problem where

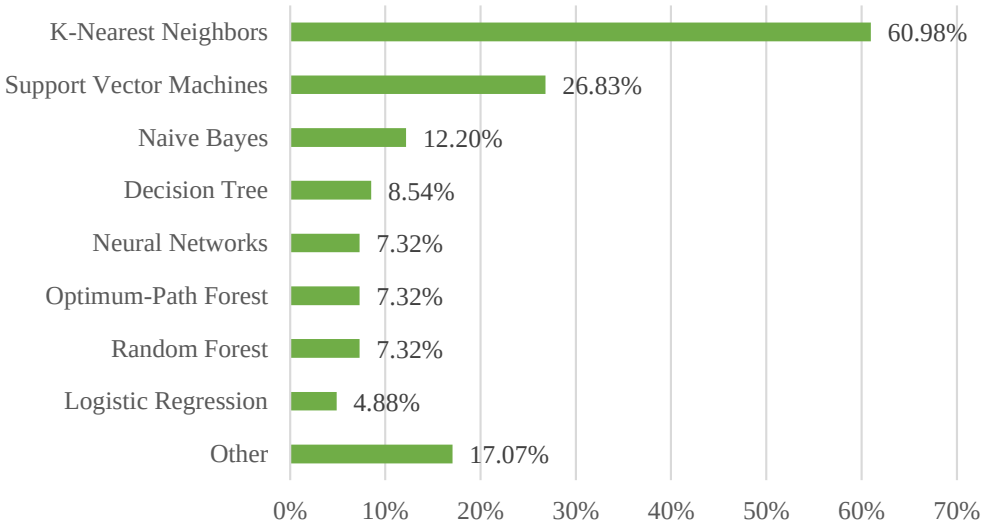

Figure 13: The most used classifiers in the analyzed feature selection studies.

the sentiment can either be positive or negative. In this scenario, the number of features may increase up to tens of thousands, where algorithms for simple scenarios may not suffice. Feature selection of gene expression data is another important topic with its high amount of features and unique requirements. All metaheuristic algorithms possess different strengths and weaknesses [29]. Therefore, by taking into consideration the differences of these algorithms, choosing the correct algorithm for a specific problem would be beneficial to improve the performance. Some of the metaheuristics are algorithmic parameterless (CS, TLBO), while some others require many parameters to be tuned (ABC, BFO, BBO). Similarly, some of them prioritize exploration (ABC, GOA), while others can perform exploitation better (GWO, SCA, WOA). For example, if the task requires analyzing a high amount of data, such as gene expression analysis, it is important to choose an algorithm that prioritizes exploration over exploitation to be able to identify the regions with high-quality solutions [71]. However, for a vehicle routing problem, the exploitation capabilities of the algorithm play an essential role in the outcome [222]. Researchers study on improving these metaheuristic algorithms in different angles for various domains and tasks [223]. There exist studies that propose an enhanced initial population for metaheuristics [224, 225] or exploration and exploitation operators [226, 227].

5. Multiobjective metaheuristic algorithms for feature selection

In this part of our survey, we summarize some of the salient articles on multiobjective feature selection version of the problem. We explain what they have achieved differently from the single-objective version

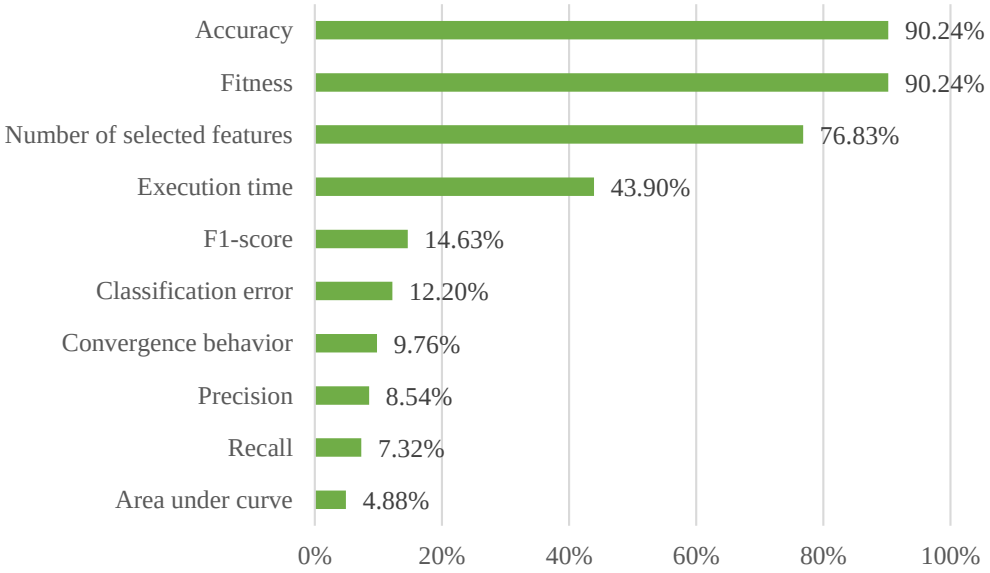

Figure 14: The most used evaluation metrics in the analyzed feature selection studies.

of the feature selection. It is clear that due to the multiobjective behavior of the problem, there is a set of solutions after optimization is applied to the problem. Therefore, we cannot expect an optimal single solution [228, 229].

Zhou et al. prepared a survey about multiobjective evolutionary algorithms between 2010 and 2018 [230]. The authors defined the multiobjective optimization problem as having several conflicting objectives and a set of solutions. Algorithmic frameworks, selection methods, reproduction operators, benchmark problems, performance indicators, and applications were analyzed in their study. Some future research issues were also discussed.

Al-Tashi et al. mentioned studies on multiobjective feature selection and proposed algorithms in their survey [86]. Their review covered related studies between 2012 and 2019. Recent challenges were explained for future research. Kiziloz et al. developed novel multiobjective TLBO methods for binary classification problems [206]. The proposed algorithms were claimed not to have any algorithm-specific parameters to be set. Experiments on UCI datasets verified that the proposed multiobjective TLBO algorithms could outperform SS, Non-dominated Sorting Genetic Algorithm II (NSGA-II), PSO, TS, and Greedy algorithms. Narendra and Fukunaga developed a branch and bound algorithm to identify the best feature subset [231]. Yang and Honavar presented a GA for the feature selection task [232]. They showed that GA improves the results on benchmark datasets.

Deniz et al. proposed a multiobjective GA for the feature selection of binary classification [10]. The

algorithm was compared with Greedy, PSO, and TS. Hancer et al. proposed a new multiobjective ABC algorithm for feature selection [13]. The algorithm used non-dominated sorting and GA operators. Zhang et al. studied a new multiobjective binary DE with a self-learning feature selection approach [233]. Wang et al. studied a multiobjective feature selection algorithm with a sample reduction strategy [234]. Moreover, Hu et al. proposed a fuzzy multiobjective feature selection method with PSO [235]. Li et al. proposed a multiobjective feature selection approach for key quality characteristics of unbalanced data [236]. The hybrid algorithm combines GA with a direct multi-search strategy. Wang et al. studied a multiobjective feature selection evolutionary algorithm [234]. K-means clustering with differential selection was proposed. An improved ABC was combined with PSO. The proposed strategy was shown to be more suitable. Ghanem & Aman proposed a multiobjective ABC for intrusion-detection systems [237]. A feed-forward neural network was used as the classifier. Castro & Fernando developed a multiobjective Bayesian AIS for feature selection [238]. The algorithm performs a multimodal search to maintain the diversity of the population. The experiments verified the efficiency of the approach to feature selection. Li & Yin proposed a multiobjective BBO algorithm that uses the non-dominated sorting method and the crowding distance method with SVM [239]. The experiments performed on benchmarks demonstrated that the proposed method is competitive against PSO with SVM. Rodrigues et al. proposed multi and many-objective variants of Artificial Butterfly Optimization [240]. The results showed that the binary single-objective version of the algorithm performed better than the other state-of-the-art techniques.

6. Hybrid metaheuristic and hyperheuristic algorithms

In this section, we give brief information about hybrid and hyperheuristic algorithms developed for the feature selection problem. The hybrid algorithms are developed by combining the current metaheuristics or classical algorithms. The main purpose of hybrid algorithms is to combine the skills of diverse algorithms to obtain better results. Therefore, hybrid metaheuristic algorithms have significant improvements compared to single metaheuristic algorithms. Hence, more efficient and flexible algorithms can be developed using these hybrid methods [241, 242]. Hyperheuristics use a set of methods to automate the design of heuristics to optimize NP-hard computational search problems [243].

Zorarpacı et al. proposed a new hybrid method that combines ABC with DE metaheuristic [244]. Du et al. proposed a hybrid HHO to address air pollution concentrations [245]. They aimed to warn the public about hazardous air pollutants. Abdel et al. presented a hybrid version of the HHO algorithm with SA to solve the feature selection problem [182]. New wrapper approaches were proposed with WOA by Mafarja and Mirjalili [17, 208]. Ibrahim et al. presented a hybrid PSO combined with SSA [246]. The exploration and the exploitation steps were enhanced. The experiments were carried out on UCI datasets.

Neggaz et al. developed a new SSA using SCA and Disrupt Operator [247]. The algorithm improved the

exploration, and a better stagnation-aware method was developed to balance exploration and exploitation. Experiments showed a good performance in terms of accuracy, sensitivity, specificity, and the number of selected features. Arora et al. proposed a hybrid GWO with CSA to generate promising candidate solutions to obtain global optima [248]. The algorithm was compared to other algorithms, and the statistical results verified that the algorithm could outperform other algorithms, including the different versions of GWO.

Zhang et al. developed a novel HHO by embedding the SSA into the original HHO to improve the ability of the optimizer to search [180]. Lee & Dae-Won presented a memetic feature selection algorithm for multi-label classification, and the authors claimed that they prevented fast convergence and improved the performance [249]. The method employs memetic procedures to obtain better feature subsets through genetic operators. Empirical studies showed that the method outperforms conventional multi-label feature selection methods. Chen et al. proposed an HHO that combines the DE, chaos, and topological multi-population strategies [250]. The algorithm was compared with recent studies, and the method was verified to be a suitable tool for complex optimization problems. Mafarja & Mirjalili combined hill-climbing and binary ALO [251]. A set of ants was generated and combined by embedding the best features with filter feature selection models. The algorithm was superior on UCI datasets when compared with the recent approaches. Sarhani et al. proposed a binary hybrid PSO and GSA algorithm for feature selection [252]. A mutation operator was developed to provide diversity in the population. The proposed method could outperform other metaheuristic and well-known feature selection algorithms. Pandey et al. introduced a CS method based on principal component analysis and fast independent component analysis to deal with the stable selection of features [147]. The algorithm was compared with plain CS, BA, GSA, WOA with SA, and binary GWO.

Hafez et al. developed a hybrid Monkey Algorithm with KH that adaptively balances the exploration and exploitation phases of the optimization [253]. The algorithm uses the movements of the chickens. The algorithm showed advances over PSO and GA on UCI datasets. Wang et al. proposed a hybrid TLBO and DE algorithm to solve chaotic time series prediction [254]. The DE was used to update the best positions of individuals to improve the TLBO in avoiding stagnation. The hybrid algorithm improved the convergence and the performance of the optimization. Deb et al. proposed an improved version of the Chicken Swarm Optimization algorithm with TLBO using an update method of roosters and a novel constraint-handling technique to get rid of stagnation [255] . The proposed algorithm was competitive with existing optimization algorithms in the literature. Oliva et al. proposed SCA and ALO with classical thresholding criteria to perform multilevel thresholding (segmentation on images) over the energy curve [256]. Experiments verified the efficiency of ALO for multilevel thresholding. Kihel & Chouraqui proposed a new clonal selection theory inspired AIS for feature subset selection [257]. The authors used the FA and clonal selection algorithms to select the most relevant features in a dataset. Two new hybrid algorithms based on Immune Firefly Algorithm were developed. The experimental results on UCI datasets showed that the methods significantly

outperform most of the used feature selection algorithms. Ghetas et al. combined MBO with HS to enhance the exploitation and exploration ability and speed up the convergence rate of MBO [258]. The experimental results demonstrated that the algorithm performs better than the classical MBO and other algorithms. Kora et al. combined BFO and PSO for the feature selection of Electrocardiogram signals [259]. The method executes local search with the chemotactic movements of BFO and the global search with a PSO operator. Shreem et al. proposed a selection algorithm for microarray datasets with a symmetrical uncertainty filter and HS [260]. Experimental results showed that the algorithm performs better in all datasets in terms of the accuracy and number of features. Nekkaa et al. proposed a hybrid HS with a stochastic local search for feature selection [261]. A novel selection method was developed to select the solutions for local refinement, providing a good balance between exploration and exploitation. Experimental results were observed to be competitive. Mafarja & Mirjalili developed a hybrid WOA and SA algorithm [208]. The SA enhances the exploitation by working on the most promising regions selected by the WOA. The results verified the efficiency of the proposed approaches compared to other wrappers. Das et al. developed a method that combines an SVM with TLBO for the prediction of financial time-series data [262].

Yogesh et al. proposed a new BBO-PSO algorithm for the feature selection of speech recognition system, identifying speaker's emotion [263]. The experiments were carried out on many datasets, and the obtained results proved the effectiveness of the algorithm compared to previous metaheuristics. Al-Tashi et al. proposed a binary hybrid GWO and PSO algorithm for the feature selection problem [167]. The KNN classifier with the Euclidean separation metric was employed in the algorithm. The results on UCI datasets verified that the algorithm outperformed GWO, PSO, GA, and WOA with SA in terms of accuracy and the number of features. Kumar & Bharti developed a hybrid SCA with PSO algorithm to select the best subset of features [264]. A V-shaped transfer function was used to calculate the probability of changing the locations of the particles. The performance of the algorithm was better than WOA, moth flame algorithm, DFA, SCA, and ABC. The results demonstrated that the proposed algorithm attained better performance in most of the datasets. Anter & Ali developed a hybrid CSA with chaos theory and fuzzy c-means for feature selection of medical diagnosis datasets [265]. The CSA uses the global optimization technique to avoid the stagnation problem of local search. The proposed algorithm was compared with chaotic ALO, CSA, ALO, and BA on breast cancer, lung cancer, diabetes, heart, hepatitis, liver disorders, and arrhythmia datasets.

Hyperheuristics algorithms were first introduced in 2000 [243]. There are mainly two hyperheuristic approaches: heuristic selection and generation. Montazeri proposed a hyperheuristic for feature selection in 2016 [266]. Low-level heuristics were named exploiter and explorer. The author proposed a GA to select low-level heuristics and balance the exploitation and exploration activities. An adaptive feature selection method was proposed, which was observed to outperform recent methods. Hunt et al. proposed a GP hyperheuristic for feature selection [267]. The algorithm uses new heuristics with building blocks. The heuristics were used as new search algorithms. The classifiers were improved using a small number of features obtained

by evolved heuristics. Abdollahzadeh et al. proposed an HHO algorithm with the Fruitfly Optimization Algorithm for feature selection [268]. The results were compared with NSGA-II and ABC algorithms on standard datasets with mean, best, worst, and standard deviation. The results were reported on the Pareto front charts, showing that the performance on the dataset is promising. Dif & Elberrichi proposed a dynamic hybrid algorithm with GA, PSO, BA, and DE metaheuristics [269] . The experiments were performed on face image recognition datasets with KNN. The results showed that the proposed algorithm found the best solutions.

7. Common datasets and search engines

In this part of our research, we give information about the machine learning benchmark datasets and search engines that are used by feature selection algorithms in experiments. Experiments should be done on well-known datasets to observe the performances of the proposed feature selection algorithms. It is important that these datasets have a benchmark feature, are accessible to everyone, used by previous studies, appeal to a wider audience of readers and researchers, and provide a fair experimental ground. It is also seen in some studies that the authors have produced their datasets and made them available to everyone. The number of features and the instances (rows) are among the most important properties of datasets that gain importance during experiments. The performance of metaheuristics should be verified on large datasets with a higher number of features and instances.

One of the most well-known repositories for the datasets used in machine learning studies is the University of California, Irvine (UCI) Machine Learning Repository [270]. This website collects problem instances and data generators for the empirical tests[1]. The database was developed in 1987 by David Aha, and it has been intensively used by researchers all over the world. There exist 559 publicly available datasets. The properties of the most famous datasets from the UCI repository can be seen in Table 3. The presented datasets are the most used 10 datasets in the feature selection domain according to the analysis results of 82 studies (see Figure 12).

Kaggle has been another active website for data scientists and the community of machine learning scientists since 2010[2]. At Kaggle, users can access many new datasets of the researchers and release their own datasets. It is possible to study with other scientists and engineers and initialize competitions on data science challenges. Kaggle has numerous real-life datasets in different formats. Code of algorithms and data can be obtained easily. There are over 100 thousand public datasets as of November 2021 [271]. COVID-19 Open Research Dataset, Credit Card Fraud Detection, Novel Corona Virus 2019 Dataset, and Heart Disease UCI are among the most popular datasets of this repository. Moreover, Amazon Web Services

[1] https://archive.ics.uci.edu/ml/datasets.php
[2] https://www.kaggle.com/datasets

Table 3: Frequently used benchmark datasets of UCI machine learning repository.

Dataset	# of features	# of instances	# of classes
Breast Cancer	9	286	2
Congressional Voting Records	16	435	2
Ionosphere	34	351	2
Connectionist Bench (Sonar, Mines vs. Rocks)	60	208	2
Statlog (Heart)	13	270	2
Wine	13	178	3
Zoo	17	101	7
Waveform	40	5000	3
SPECT Heart	22	267	2
Lymphography	18	148	4

(AWS)[3] resources provide many datasets in Public Transport, Satellite Images, Ecological Resources, etc. The Cancer Genome Atlas, Common Web Crawl data of 50 billion web pages, Sentinel-2 (high-resolution optical imagery), and COVID-19 Datasets are the well-known datasets of AWS.

Google provided a search engine for datasets in 2018[4]. The engine unifies tens of thousands of different repositories for datasets. The search engine helps researchers find free data. Microsoft also has its Research Open Data repository on the cloud for global research communities. This service provides datasets used in published studies[5]. Awesome Public Datasets Collection is another repository of Agriculture, Biology, Earth Science, Education, Finance, Image Processing, Search Engines, Social Networks, and Transportation. Most of the datasets are free[6].

The well-known biomedical datasets used in classification are ColonTumor, DLBCL-Harvard, and Nervous-System (gene expression, protein profiling, and genomic sequence). These datasets include high-dimensional instances from 2,000 to 12,600[7]. Kent Ridge Bio-medical Dataset[8] is another source for biomedical data. Some of the well-known cancer datasets are also available[9]. For the treatment of cancer patients, five new datasets are introduced in a study that proposes the Gene Expression Model Selector (GEMS) system [272]. LIBSVM Dataset provides preprocessed data within the LIBSVM library [273]. Scikit-feature is an open-

[3]https://registry.opendata.aws/
[4]https://datasetsearch.research.google.com/
[5]https://msropendata.com/
[6]https://github.com/awesomedata/awesome-public-datasets
[7]http://datam.i2r.a-tar.edu.sg/datasets/krbd/index.html
[8]http://leo.ugr.es/elvira/DBCRepository/
[9]http://research.janelia.org/peng/proj/ and http://www.gemssystem.org

source code repository in Python[10]. It contains more than 30 feature selection algorithms.

8. Conclusion and future work

This part discusses the open issues and challenges of feature selection problems. We believe that the use of metaheuristics for feature selection problems will grow at high speed and provide state-of-the-art methods. According to the No-Free-Lunch (NFL) theorem, no heuristic is good enough to solve all optimization problems. Most metaheuristics have at least one area where they perform better than others. However, there is no guarantee to find the best subset of features from different domains using a single metaheuristic. Considering these issues, there will always be a possibility of obtaining better results with new metaheuristics on feature selection. In the literature, hundreds of new articles are being published every year that produce high-quality solutions with metaheuristics on feature selection. This intense interest attracts the attention of numerous researchers. The reported results of these algorithms are remarkable even with large datasets. In this context, we reviewed the studies of distinguished academicians who received many citations and whose papers are published in top journals and conferences.

Local optima and stagnation are the biggest challenges of the combinatorial optimization community. Many techniques are being developed and tested to tackle these problems. In this sense, the metaheuristics are gradient-free approaches, and they are good at obtaining global optimal values, whereas gradient-based methods are more prone to finding the local optima. Therefore, the strategy of gradient-free methods is more powerful than other approaches.

A more fair test-bed is required for new metaheuristic feature selection algorithms. This can be a good area of research for the feature selection problem. A black-box optimization benchmark can be provided, and the results of the new algorithms can be verified on this test-bed [274, 275]. One of the main drawbacks of the metaheuristic feature selection algorithms is that they have to calculate each new candidate solution using a machine learning algorithm, which consumes a significant amount of computation time and hinders an effective optimization of selecting the best subset of features. Therefore, faster machine learning techniques can be more promising alternatives while obtaining the best accuracy solutions. High-performance computation environments like GPUs, MPI, and OpenMP can be good solutions to deal with this computation burden. Since the granularity of metaheuristics is very suitable for parallel computation, each chromosome can be calculated individually and inserted into the population. Alba et al. present the advantages and opportunities that can be realized using parallel metaheuristics in their comprehensive studies [276, 277].

The introduction of a new metaheuristic facilitates many alternatives to build new hybrid wrapper algorithms. There will always be a prolific research area considering the hybrid versions of the new meta-

[10] http://featureselection.asu.edu/datasets.php

heuristics. Memetic algorithms can also be noted as another research area that combines metaheuristics and local search algorithms for better prediction accuracies.

Another exciting research area related to wrapper style feature selection algorithms can be using hyper-heuristic methods. We think that there is not enough work in this field, and it is an open area for improvement. These methods are constructed on the main principles of the NFL theorem. They select the best possible low-heuristic among many alternatives and increase the possibility of obtaining the (near)-optimal solution according to the behaviour of the problem space. The basic heuristics of the recent algorithms proposed in this review can be tested in a hyperheuristic approach. We believe that interesting results will be observed during the experiments.

It is not expected that every metaheuristic feature selection algorithm will work best in all problems. Therefore, under-performing results should also be reported. The researcher should give good reasons why the proposed algorithm works well for particular issues. Experiments should be expected to provide the same results under the same conditions. The number of experimental executions should be large enough to provide statistical analysis, and the average results of the executions should be presented. An equal number of fitness evaluations should be performed when making comparisons with other metaheuristics to be fair. The code (datasets and even the LaTeX formulas) should be publicly available.

Providing a good balance between the exploration and exploitation phases of the metaheuristics will always be a critical issue for better results. Efforts on this problem should also be highlighted as new metaheuristics are being proposed. In these days of epidemic disease, new datasets of COVID-19 are being introduced every day. Verifying the results of the new proposed wrapper feature selection algorithms in this field will gain importance shortly.

Acknowledgement

We would like to thank Ozlem Tekdemir Dokeroglu for her artistic illustrations in Figures 10 and 11.

References

[1] Z. Obermeyer, E. J. Emanuel, Predicting the future—big data, machine learning, and clinical medicine, The New England journal of medicine 375 (13) (2016) 1216.

[2] J. Qiu, Q. Wu, G. Ding, Y. Xu, S. Feng, A survey of machine learning for big data processing, EURASIP Journal on Advances in Signal Processing 2016 (1) (2016) 1–16.

[3] L. Zhou, S. Pan, J. Wang, A. V. Vasilakos, Machine learning on big data: Opportunities and challenges, Neurocomputing 237 (2017) 350–361.

[4] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Computers & Electrical Engineering 40 (1) (2014) 16–28.

[5] R. Kohavi, G. H. John, et al., Wrappers for feature subset selection, Artificial intelligence 97 (1-2) (1997) 273–324.

[6] H. Liu, L. Yu, Toward integrating feature selection algorithms for classification and clustering, IEEE Transactions on knowledge and data engineering 17 (4) (2005) 491–502.

[7] I. BoussaïD, J. Lepagnot, P. Siarry, A survey on optimization metaheuristics, Information sciences 237 (2013) 82–117.

[8] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, A. Cosar, A survey on new generation metaheuristic algorithms, Computers & Industrial Engineering 137 (2019) 106040.

[9] J. Miao, L. Niu, A survey on feature selection, Procedia Computer Science 91 (2016) 919–926.

[10] A. Deniz, H. E. Kiziloz, T. Dokeroglu, A. Cosar, Robust multiobjective evolutionary feature subset selection algorithm for binary classification using machine learning techniques, Neurocomputing 241 (2017) 128–146.

[11] E. Emary, H. M. Zawbaa, A. E. Hassanien, Binary ant lion approaches for feature selection, Neurocomputing 213 (2016) 54–65.

[12] E. Emary, H. M. Zawbaa, A. E. Hassanien, Binary grey wolf optimization approaches for feature selection, Neurocomputing 172 (2016) 371–381.

[13] E. Hancer, B. Xue, M. Zhang, D. Karaboga, B. Akay, Pareto front feature selection based on artificial bee colony optimization, Information Sciences 422 (2018) 462–479.

[14] S. Kashef, H. Nezamabadi-pour, An advanced aco algorithm for feature subset selection, Neurocomputing 147 (2015) 271–279.

[15] M. Mafarja, I. Aljarah, H. Faris, A. I. Hammouri, A.-Z. Ala'M, S. Mirjalili, Binary grasshopper optimisation algorithm approaches for feature selection problems, Expert Systems with Applications 117 (2019) 267–286.

[16] M. Mafarja, I. Aljarah, A. A. Heidari, H. Faris, P. Fournier-Viger, X. Li, S. Mirjalili, Binary dragonfly optimization for feature selection using time-varying transfer functions, Knowledge-Based Systems 161 (2018) 185–204.

[17] M. Mafarja, S. Mirjalili, Whale optimization approaches for wrapper feature selection, Applied Soft Computing 62 (2018) 441–453.

[18] Y. Zhang, X.-f. Song, D.-w. Gong, A return-cost-based binary firefly algorithm for feature selection, Information Sciences 418 (2017) 561–574.

[19] D. E. Goldberg, J. H. Holland, Genetic algorithms and machine learning, Machine Learning 3 (1988) 95–99.

[20] F. Han, W.-T. Chen, Q.-H. Ling, H. Han, Multi-objective particle swarm optimization with adaptive strategies for feature selection, Swarm and Evolutionary Computation (2021) 100847.

[21] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-international conference on neural networks, Vol. 4, IEEE, 1995, pp. 1942–1948.

[22] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, IEEE computational intelligence magazine 1 (4) (2006) 28–39.

[23] P. J. Van Laarhoven, E. H. Aarts, Simulated annealing, in: Simulated annealing: Theory and applications, Springer, 1987, pp. 7–15.

[24] W. Banzhaf, P. Nordin, R. E. Keller, F. D. Francone, Genetic programming: an introduction, Vol. 1, Morgan Kaufmann Publishers San Francisco, 1998.

[25] K. V. Price, Differential evolution, in: Handbook of optimization, Springer, 2013, pp. 187–214.

[26] F. Glover, M. Laguna, Tabu search, in: Handbook of combinatorial optimization, Springer, 1998, pp. 2093–2229.

[27] J. E. Hunt, D. E. Cooke, Learning using an artificial immune system, Journal of network and computer applications 19 (2) (1996) 189–212.

[28] Y. Lu, M. Liang, Z. Ye, L. Cao, Improved particle swarm optimization algorithm and its application in text feature selection, Applied Soft Computing 35 (2015) 629–636.

[29] D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, IEEE transactions on evolutionary computation 1 (1) (1997) 67–82.

[30] Z. W. Geem, J. H. Kim, G. V. Loganathan, A new heuristic optimization algorithm: harmony search, simulation 76 (2) (2001) 60–68.

[31] K. M. Passino, Biomimicry of bacterial foraging for distributed optimization and control, IEEE control systems magazine 22 (3) (2002) 52–67.

[32] X. Li, A new intelligent optimization-artificial fish swarm algorithm, Doctor thesis, Zhejiang University of Zhejiang, China (2003) 27.

[33] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Tech. rep., Technical report-tr06, Erciyes university, engineering faculty, computer engineering department (2005).

[34] D. Simon, Biogeography-based optimization, IEEE transactions on evolutionary computation 12 (6) (2008) 702–713.

[35] X.-S. Yang, Firefly algorithms for multimodal optimization, in: International symposium on stochastic algorithms, Springer, 2009, pp. 169–178.

[36] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, Gsa: a gravitational search algorithm, Information sciences 179 (13) (2009) 2232–2248.

[37] X.-S. Yang, S. Deb, Cuckoo search via lévy flights, in: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), IEEE, 2009, pp. 210–214.

[38] X.-S. Yang, A new metaheuristic bat-inspired algorithm, in: Nature inspired cooperative strategies for optimization (NICSO 2010), Springer, 2010, pp. 65–74.

[39] R. V. Rao, V. J. Savsani, D. Vakharia, Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, Computer-Aided Design 43 (3) (2011) 303–315.

[40] A. H. Gandomi, A. H. Alavi, Krill herd: a new bio-inspired optimization algorithm, Communications in nonlinear science and numerical simulation 17 (12) (2012) 4831–4845.

[41] E. Cuevas, M. Cienfuegos, D. ZaldíVar, M. Pérez-Cisneros, A swarm optimization algorithm inspired in the behavior of the social-spider, Expert Systems with Applications 40 (16) (2013) 6374–6384.

[42] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, Advances in engineering software 69 (2014) 46–61.

[43] S. Mirjalili, The ant lion optimizer, Advances in engineering software 83 (2015) 80–98.

[44] S. Mirjalili, Sca: a sine cosine algorithm for solving optimization problems, Knowledge-based systems 96 (2016) 120–133.

[45] S. Mirjalili, A. Lewis, The whale optimization algorithm, Advances in engineering software 95 (2016) 51–67.

[46] A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm, Computers & Structures 169 (2016) 1–12.

[47] S. Mirjalili, Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, Neural Computing and Applications 27 (4) (2016) 1053–1073.

[48] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, Advances in Engineering Software 114 (2017) 163–191.

[49] S. Z. Mirjalili, S. Mirjalili, S. Saremi, H. Faris, I. Aljarah, Grasshopper optimization algorithm for multi-objective optimization problems, Applied Intelligence 48 (4) (2018) 805–820.

[50] S. Arora, S. Singh, Butterfly optimization algorithm: a novel approach for global optimization, Soft Computing 23 (3) (2019) 715–734.

[51] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future generation computer systems 97 (2019) 849–872.

[52] K. Kira, L. A. Rendell, et al., The feature selection problem: Traditional methods and a new algorithm, in: Aaai, Vol. 2, 1992, pp. 129–134.

[53] W. Siedlecki, J. Sklansky, On automatic feature selection, in: Handbook of pattern recognition and computer vision, World Scientific, 1993, pp. 63–87.

[54] M. Dash, H. Liu, Feature selection for classification, Intelligent data analysis 1 (3) (1997) 131–156.

[55] M. J. Martin-Bautista, M.-A. Vila, A survey of genetic feature selection in mining issues, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Vol. 2, IEEE, 1999, pp. 1314–1321.

[56] P. Leray, P. Gallinari, Feature selection with neural networks, Behaviormetrika 26 (1) (1999) 145–166.

[57] L. C. Molina, L. Belanche, À. Nebot, Feature selection algorithms: A survey and experimental evaluation, in: 2002 IEEE International Conference on Data Mining, 2002. Proceedings., IEEE, 2002, pp. 306–313.

[58] Y. Saeys, I. Inza, P. Larranaga, A review of feature selection techniques in bioinformatics, bioinformatics 23 (19) (2007) 2507–2517.

[59] S. C. Yusta, Different metaheuristic strategies to solve the feature selection problem, Pattern Recognition Letters 30 (5) (2009) 525–534.

[60] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, H. Liu, Advancing feature selection research, ASU feature selection repository (2010) 1–28.

[61] H. Liu, H. Motoda, R. Setiono, Z. Zhao, Feature selection: An ever evolving frontier in data mining, in: Feature selection in data mining, PMLR, 2010, pp. 4–13.

[62] B. De La Iglesia, Evolutionary computation for feature selection in classification problems, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 3 (6) (2013) 381–407.

[63] S. Ganapathy, K. Kulothungan, S. Muthurajkumar, M. Vijayalakshmi, P. Yogesh, A. Kannan, Intelligent feature selection and classification techniques for intrusion detection in networks: a survey, EURASIP Journal on Wireless Communications and Networking 2013 (1) (2013) 1–16.

[64] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, A review of feature selection methods on synthetic data, Knowledge and information systems 34 (3) (2013) 483–519.

[65] Y. Zhai, Y.-S. Ong, I. W. Tsang, The emerging "big dimensionality", IEEE Computational Intelligence Magazine 9 (3) (2014) 14–26.

[66] V. Kumar, S. Minz, Feature selection: a literature review, SmartCR 4 (3) (2014) 211–229.

[67] J. Tang, S. Alelyani, H. Liu, Feature selection for classification: A review, Data classification: Algorithms and applications (2014) 37.

[68] J. R. Vergara, P. A. Estévez, A review of feature selection methods based on mutual information, Neural computing and applications 24 (1) (2014) 175–186.

[69] S. Khalid, T. Khalil, S. Nasreen, A survey of feature selection and feature extraction techniques in machine learning, in: 2014 science and information conference, IEEE, 2014, pp. 372–378.

[70] B. Xue, M. Zhang, W. N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, IEEE Transactions on Evolutionary Computation 20 (4) (2015) 606–626.

[71] J. C. Ang, A. Mirzal, H. Haron, H. N. A. Hamed, Supervised, unsupervised, and semi-supervised feature selection: a review on gene selection, IEEE/ACM transactions on computational biology and bioinformatics 13 (5) (2015) 971–989.

[72] A. Jović, K. Brkić, N. Bogunović, A review of feature selection methods with applications, in: 2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO), Ieee, 2015, pp. 1200–1205.

[73] D. A. A. Gnana, S. A. A. Balamurugan, E. J. Leavline, Literature review on feature selection methods for high-dimensional data, International Journal of Computer Applications 975 (2016) 8887.

[74] L. Wang, Y. Wang, Q. Chang, Feature selection methods for big data bioinformatics: A survey from the search perspective, Methods 111 (2016) 21–31.

[75] Y. Li, T. Li, H. Liu, Recent advances in feature selection and its applications, Knowledge and Information Systems 53 (3) (2017) 551–577.

[76] R. Sheikhpour, M. A. Sarram, S. Gharaghani, M. A. Z. Chahooki, A survey on semi-supervised feature selection methods, Pattern Recognition 64 (2017) 141–158.

[77] J. Li, H. Liu, Challenges of feature selection for big data analytics, IEEE Intelligent Systems 32 (2) (2017) 9–15.

[78] J. Cai, J. Luo, S. Wang, S. Yang, Feature selection in machine learning: A new perspective, Neurocomputing 300 (2018) 70–79.

[79] R. J. Urbanowicz, M. Meeker, W. La Cava, R. S. Olson, J. H. Moore, Relief-based feature selection: Introduction and review, Journal of biomedical informatics 85 (2018) 189–203.

[80] L. Brezočnik, I. Fister, V. Podgorelec, Swarm intelligence algorithms for feature selection: a review, Applied Sciences 8 (9) (2018) 1521.

[81] X. Deng, Y. Li, J. Weng, J. Zhang, Feature selection for text classification: A review, Multimedia Tools and Applications 78 (3) (2019) 3797–3816.

[82] B. Venkatesh, J. Anuradha, A review of feature selection and its methods, Cybernetics and Information Technologies 19 (1) (2019) 3–26.

[83] J. F. M.-T. Saúl Solorio-Fernández, J. Ariel Carrasco-Ochoa, A review of unsupervised feature selection methods, Artificial Intelligence Review 53 (2019) 907–948.

[84] V. Bolón-Canedo, A. Alonso-Betanzos, Ensembles for feature selection: A review and future trends, Information Fusion 52 (2019) 1–12.

[85] W. Liu, J. Wang, A brief survey on nature-inspired metaheuristics for feature selection in classification in this decade, in: 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC), IEEE, 2019, pp. 424–429.

[86] Q. Al-Tashi, S. J. Abdulkadir, H. M. Rais, S. Mirjalili, H. Alhussian, Approaches to multi-objective feature selection: A systematic literature review, IEEE Access 8 (2020) 125076–125096.

[87] P. Agrawal, H. F. Abutarboush, T. Ganesh, A. W. Mohamed, Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019), IEEE Access 9 (2021) 26766–26791.

[88] A. Unler, A. Murat, A discrete particle swarm optimization method for feature selection in binary classification problems, European Journal of Operational Research 206 (3) (2010) 528–539.

[89] Y. Zhang, Y.-h. Wang, D.-w. Gong, X.-y. Sun, Clustering-guided particle swarm feature selection algorithm for high-dimensional imbalanced data with missing values, IEEE Transactions on Evolutionary Computation.

[90] X.-F. Song, Y. Zhang, D.-W. Gong, X.-Z. Gao, A fast hybrid feature selection based on correlation-guided clustering and particle swarm optimization for high-dimensional data, IEEE Transactions on Cybernetics.

[91] X.-f. Song, Y. Zhang, D.-w. Gong, X.-y. Sun, Feature selection using bare-bones particle swarm optimization with mutual information, Pattern Recognition 112 (2021) 107804.

[92] X.-F. Song, Y. Zhang, Y.-N. Guo, X.-Y. Sun, Y.-L. Wang, Variable-size cooperative coevolutionary particle swarm optimization for feature selection on high-dimensional data, IEEE Transactions on Evolutionary Computation 24 (5) (2020) 882–895.

[93] Y. Zhang, D.-w. Gong, J. Cheng, Multi-objective particle swarm optimization approach for cost-based feature selection in classification, IEEE/ACM transactions on computational biology and bioinformatics 14 (1) (2015) 64–75.

[94] B. Xue, M. Zhang, W. N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, IEEE transactions on cybernetics 43 (6) (2012) 1656–1671.

[95] M. Taradeh, M. Mafarja, A. A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, H. Fujita, An evolutionary gravitational search-based feature selection, Information Sciences 497 (2019) 219–239.

[96] E. Hancer, B. Xue, M. Zhang, Differential evolution for filter feature selection based on information theory and feature ranking, Knowledge-Based Systems 140 (2018) 103–119.

[97] J. Xu, J. Zhang, Exploration-exploitation tradeoffs in metaheuristics: Survey and analysis, in: Proceedings of the 33rd

Chinese control conference, IEEE, 2014, pp. 8633–8638.

[98] Y. Borenstein, R. Poli, Information landscapes, in: Proceedings of the 7th annual conference on Genetic and evolutionary computation, 2005, pp. 1515–1522.

[99] K. Hussain, M. N. M. Salleh, S. Cheng, Y. Shi, On the exploration and exploitation in popular swarm-based metaheuristic algorithms, Neural Computing and Applications 31 (11) (2019) 7665–7683.

[100] J. Too, S. Mirjalili, A hyper learning binary dragonfly algorithm for feature selection: A covid-19 case study, Knowledge-Based Systems (2020) 106553.

[101] J. Kennedy, R. C. Eberhart, A discrete binary version of the particle swarm algorithm, in: 1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation, Vol. 5, IEEE, 1997, pp. 4104–4108.

[102] Á. E. Eiben, R. Hinterding, Z. Michalewicz, Parameter control in evolutionary algorithms, IEEE Transactions on evolutionary computation 3 (2) (1999) 124–141.

[103] M. Črepinšek, S.-H. Liu, L. Mernik, A note on teaching–learning-based optimization algorithm, Information Sciences 212 (2012) 79–93.

[104] T. Dokeroglu, E. Sevinc, Evolutionary parallel extreme learning machines for the data classification problem, Computers & Industrial Engineering 130 (2019) 237–249.

[105] E. Sevinc, A novel evolutionary algorithm for data classification problem with extreme learning machines, IEEE Access 7 (2019) 122419–122427.

[106] Y. Chen, Z. Lin, X. Zhao, G. Wang, Y. Gu, Deep learning-based classification of hyperspectral data, IEEE Journal of Selected topics in applied earth observations and remote sensing 7 (6) (2014) 2094–2107.

[107] A. J. Viera, J. M. Garrett, et al., Understanding interobserver agreement: the kappa statistic, Fam med 37 (5) (2005) 360–363.

[108] R. Taylor, Interpretation of the correlation coefficient: a basic review, Journal of diagnostic medical sonography 6 (1) (1990) 35–39.

[109] C. J. Willmott, K. Matsuura, Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance, Climate research 30 (1) (2005) 79–82.

[110] J. Carrasco, S. García, M. Rueda, S. Das, F. Herrera, Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review, Swarm and Evolutionary Computation 54 (2020) 100665.

[111] M. Wang, C. Wu, L. Wang, D. Xiang, X. Huang, A feature selection approach for hyperspectral image based on modified ant lion optimizer, Knowledge-Based Systems 168 (2019) 39–48.

[112] H. M. Zawbaa, E. Emary, C. Grosan, Feature selection via chaotic antlion optimization, PloS one 11 (3) (2016) e0150652.

[113] M. Mafarja, D. Eleyan, S. Abdullah, S. Mirjalili, S-shaped vs. v-shaped transfer functions for ant lion optimization algorithm in feature selection problem, in: Proceedings of the international conference on future networks and distributed systems, 2017, pp. 1–7.

[114] T. Dokeroglu, S. Pehlivan, B. Avenoglu, Robust parallel hybrid artificial bee colony algorithms for the multi-dimensional numerical optimization, The Journal of Supercomputing (2020) 1–21.

[115] T. Dokeroglu, E. Sevinc, A. Cosar, Artificial bee colony optimization for the quadratic assignment problem, Applied soft computing 76 (2019) 595–606.

[116] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm, Journal of global optimization 39 (3) (2007) 459–471.

[117] M. Schiezaro, H. Pedrini, Data feature selection based on artificial bee colony algorithm, EURASIP Journal on Image and Video Processing 2013 (1) (2013) 1–8.

[118] Y. Zhang, S. Cheng, Y. Shi, D.-w. Gong, X. Zhao, Cost-sensitive feature selection using two-archive multi-objective artificial bee colony algorithm, Expert Systems with Applications 137 (2019) 46–58.

[119] H. Rao, X. Shi, A. K. Rodrigue, J. Feng, Y. Xia, M. Elhoseny, X. Yuan, L. Gu, Feature selection based on artificial bee colony and gradient boosting decision tree, Applied Soft Computing 74 (2019) 634–642.

[120] F. G. Mohammadi, M. S. Abadeh, Image steganalysis using a bee colony based feature selection algorithm, Engineering Applications of Artificial Intelligence 31 (2014) 35–43.

[121] Y. Wang, L. Feng, J. Zhu, Novel artificial bee colony based feature selection method for filtering redundant information, Applied Intelligence 48 (4) (2018) 868–885.

[122] R. Manikandan, A. Kalpana, Feature selection using fish swarm optimization in big data, Cluster Computing 22 (5) (2019) 10825–10837.

[123] M. S. R. Nalluri, T. SaiSujana, K. H. Reddy, V. Swaminathan, An efficient feature selection using artificial fish swarm optimization and svm classifier, in: 2017 international conference on networks & advances in computational technologies (NetACT), IEEE, 2017, pp. 407–411.

[124] M. Zhang, C. Shao, F. Li, Y. Gan, J. Sun, Evolving neural network classifiers and feature subset using artificial fish swarm, in: 2006 international conference on mechatronics and automation, IEEE, 2006, pp. 1598–1602.

[125] K. M. Passino, Bacterial foraging optimization, International Journal of Swarm Intelligence Research (IJSIR) 1 (1) (2010) 1–16.

[126] H. Wang, B. Niu, A novel bacterial algorithm with randomness control for feature selection in classification, Neurocomputing 228 (2017) 176–186.

[127] M. Pal, S. Bhattacharyya, S. Roy, A. Konar, D. Tibarewala, R. Janarthanan, A bacterial foraging optimization and learning automata based feature selection for motor imagery eeg classification, in: 2014 International Conference on Signal Processing and Communications (SPCOM), IEEE, 2014, pp. 1–5.

[128] B. Niu, W. Yi, L. Tan, S. Geng, H. Wang, A multi-objective feature selection method based on bacterial foraging optimization, Natural Computing (2019) 1–14.

[129] X.-S. Yang, X. He, Bat algorithm: literature review and applications, International Journal of Bio-inspired computation 5 (3) (2013) 141–149.

[130] D. Rodrigues, L. A. Pereira, R. Y. Nakamura, K. A. Costa, X.-S. Yang, A. N. Souza, J. P. Papa, A wrapper approach for feature selection based on bat algorithm and optimum-path forest, Expert Systems with Applications 41 (5) (2014) 2250–2258.

[131] S. Jeyasingh, M. Veluchamy, Modified bat algorithm for feature selection with the wisconsin diagnosis breast cancer (wdbc) dataset, Asian Pacific journal of cancer prevention: APJCP 18 (5) (2017) 1257.

[132] A. M. Taha, A. Mustapha, S.-D. Chen, Naive bayes-guided bat algorithm for feature selection, The Scientific World Journal 2013.

[133] R. Y. M. Nakamura, L. A. M. Pereira, D. Rodrigues, K. A. P. Costa, J. P. Papa, X.-S. Yang, Binary bat algorithm for feature selection, in: Swarm Intelligence and Bio-Inspired Computation, Elsevier, 2013, pp. 225–237.

[134] D. Albashish, A. I. Hammouri, M. Braik, J. Atwan, S. Sahran, Binary biogeography-based optimization based svm-rfe for feature selection, Applied Soft Computing 101 (2021) 107026.

[135] B. Liu, M. Tian, C. Zhang, X. Li, Discrete biogeography based optimization for feature selection in molecular signatures, Molecular informatics 34 (4) (2015) 197–215.

[136] S. Arora, P. Anand, Binary butterfly optimization approaches for feature selection, Expert Systems with Applications 116 (2019) 147–160.

[137] Z. Sadeghian, E. Akbari, H. Nematzadeh, A hybrid feature selection method based on information theory and binary butterfly optimization algorithm, Engineering Applications of Artificial Intelligence 97 (2021) 104079.

[138] M. Alweshah, S. Al Khalaileh, B. B. Gupta, A. Almomani, A. I. Hammouri, M. A. Al-Betar, The monarch butterfly optimization algorithm for solving feature selection problems, Neural Computing and Applications (2020) 1–15.

[139] S. Ouadfel, M. Abd Elaziz, Enhanced crow search algorithm for feature selection, Expert Systems with Applications 159 (2020) 113572.

[140] G. I. Sayed, A. E. Hassanien, A. T. Azar, Feature selection via a novel chaotic crow search algorithm, Neural computing and applications 31 (1) (2019) 171–188.

[141] D. Gupta, J. J. Rodrigues, S. Sundaram, A. Khanna, V. Korotaev, V. H. C. de Albuquerque, Usability feature extraction using modified crow search algorithm: a novel approach, Neural Computing and Applications (2018) 1–11.

[142] R. C. T. De Souza, L. dos Santos Coelho, C. A. De Macedo, J. Pierezan, A v-shaped binary crow search algorithm for feature selection, in: 2018 IEEE congress on evolutionary computation (CEC), IEEE, 2018, pp. 1–8.

[143] A. H. Gandomi, X.-S. Yang, A. H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, Engineering with computers 29 (1) (2013) 17–35.

[144] X.-S. Yang, Cuckoo search and firefly algorithm: theory and applications, Vol. 516, Springer, 2013.

[145] X.-S. Yang, S. Deb, Engineering optimisation by cuckoo search, International Journal of Mathematical Modelling and Numerical Optimisation 1 (4) (2010) 330–343.

[146] M. Shehab, A. T. Khader, M. A. Al-Betar, A survey on applications and variants of the cuckoo search algorithm, Applied Soft Computing 61 (2017) 1041–1059.

[147] A. C. Pandey, D. S. Rajpoot, M. Saraswat, Feature selection method based on hybrid data transformation and binary binomial cuckoo search, Journal of Ambient Intelligence and Humanized Computing 11 (2) (2020) 719–738.

[148] M. Abd El Aziz, A. E. Hassanien, Modified cuckoo search algorithm with rough sets for feature selection, Neural Computing and Applications 29 (4) (2018) 925–934.

[149] D. Rodrigues, L. A. Pereira, T. Almeida, J. P. Papa, A. Souza, C. C. Ramos, X.-S. Yang, Bcs: A binary cuckoo search algorithm for feature selection, in: 2013 IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, 2013, pp. 465–468.

[150] M. M. Mafarja, D. Eleyan, I. Jaber, A. Hammouri, S. Mirjalili, Binary dragonfly algorithm for feature selection, in: 2017 international conference on new trends in computing sciences (ICTCS), IEEE, 2017, pp. 12–17.

[151] A. I. Hammouri, M. Mafarja, M. A. Al-Betar, M. A. Awadallah, I. Abu-Doush, An improved dragonfly algorithm for feature selection, Knowledge-Based Systems 203 (2020) 106131.

[152] G. I. Sayed, A. Tharwat, A. E. Hassanien, Chaotic dragonfly algorithm: an improved metaheuristic algorithm for feature selection, Applied Intelligence 49 (1) (2019) 188–205.

[153] X.-S. Yang, Firefly algorithm, stochastic test functions and design optimisation, International journal of bio-inspired computation 2 (2) (2010) 78–84.

[154] I. Fister, I. Fister Jr, X.-S. Yang, J. Brest, A comprehensive review of firefly algorithms, Swarm and Evolutionary Computation 13 (2013) 34–46.

[155] E. Emary, H. M. Zawbaa, K. K. A. Ghany, A. E. Hassanien, B. Parv, Firefly optimization algorithm for feature selection, in: Proceedings of the 7th Balkan Conference on Informatics Conference, 2015, pp. 1–7.

[156] B. Selvakumar, K. Muneeswaran, Firefly algorithm based feature selection for network intrusion detection, Computers & Security 81 (2019) 148–155.

[157] L. Zhang, K. Mistry, C. P. Lim, S. C. Neoh, Feature selection using firefly optimization for classification and regression models, Decision Support Systems 106 (2018) 64–85.

[158] A. H. Gandomi, X.-S. Yang, S. Talatahari, A. H. Alavi, Firefly algorithm with chaos, Communications in Nonlinear Science and Numerical Simulation 18 (1) (2013) 89–98.

[159] Y. Meraihi, A. B. Gabis, S. Mirjalili, A. Ramdane-Cherif, Grasshopper optimization algorithm: Theory, variants, and

applications, IEEE Access 9 (2021) 50001–50024.

[160] S. Saremi, S. Mirjalili, A. Lewis, Grasshopper optimisation algorithm: theory and application, Advances in Engineering Software 105 (2017) 30–47.

[161] M. Mafarja, I. Aljarah, A. A. Heidari, A. I. Hammouri, H. Faris, A.-Z. Ala'M, S. Mirjalili, Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems, Knowledge-Based Systems 145 (2018) 25–45.

[162] A. Zakeri, A. Hokmabadi, Efficient feature selection method using real-valued grasshopper optimization algorithm, Expert Systems with Applications 119 (2019) 61–72.

[163] J. P. Papa, A. Pagnin, S. A. Schellini, A. Spadotto, R. C. Guido, M. Ponti, G. Chiachia, A. X. Falcão, Feature selection through gravitational search algorithm, in: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2011, pp. 2052–2055.

[164] J. Xiang, X. Han, F. Duan, Y. Qiang, X. Xiong, Y. Lan, H. Chai, A novel hybrid system for feature selection based on an improved gravitational search algorithm and k-nn method, Applied Soft Computing 31 (2015) 293–307.

[165] S. Nagpal, S. Arora, S. Dey, et al., Feature selection using gravitational search algorithm for biomedical data, Procedia Computer Science 115 (2017) 258–265.

[166] S. Mirjalili, S. Saremi, S. M. Mirjalili, L. d. S. Coelho, Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization, Expert Systems with Applications 47 (2016) 106–119.

[167] Q. Al-Tashi, S. J. A. Kadir, H. M. Rais, S. Mirjalili, H. Alhussian, Binary optimization using hybrid grey wolf optimization for feature selection, IEEE Access 7 (2019) 39496–39508.

[168] Q. Tu, X. Chen, X. Liu, Multi-strategy ensemble grey wolf optimizer and its application to feature selection, Applied Soft Computing 76 (2019) 16–30.

[169] P. Hu, J.-S. Pan, S.-C. Chu, Improved binary grey wolf optimizer and its application for feature selection, Knowledge-Based Systems 195 (2020) 105746.

[170] H. Chantar, M. Mafarja, H. Alsawalqah, A. A. Heidari, I. Aljarah, H. Faris, Feature selection using binary grey wolf optimizer with elite-based crossover for arabic text classification, Neural Computing and Applications 32 (16) (2020) 12201–12220.

[171] D. Manjarres, I. Landa-Torres, S. Gil-Lopez, J. Del Ser, M. N. Bilbao, S. Salcedo-Sanz, Z. W. Geem, A survey on applications of the harmony search algorithm, Engineering Applications of Artificial Intelligence 26 (8) (2013) 1818–1831.

[172] X.-S. Yang, Harmony search as a metaheuristic algorithm, in: Music-inspired harmony search algorithm, Springer, 2009, pp. 1–14.

[173] J. Gholami, F. Pourpanah, X. Wang, Feature selection based on improved binary global harmony search for data classification, Applied Soft Computing (2020) 106402.

[174] C. C. Ramos, A. N. Souza, G. Chiachia, A. X. Falcão, J. P. Papa, A novel algorithm for feature selection using harmony search and its application for non-technical losses detection, Computers & Electrical Engineering 37 (6) (2011) 886–894.

[175] H. H. Inbarani, M. Bagyamathi, A. T. Azar, A novel hybrid feature selection method based on rough set and improved harmony search, Neural Computing and Applications 26 (8) (2015) 1859–1880.

[176] A. Moayedikia, K.-L. Ong, Y. L. Boo, W. G. Yeoh, R. Jensen, Feature selection for high dimensional imbalanced class data using harmony search, Engineering Applications of Artificial Intelligence 57 (2017) 38–49.

[177] R. Diao, Q. Shen, Feature selection with harmony search, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (6) (2012) 1509–1523.

[178] Y. Wang, Y. Liu, L. Feng, X. Zhu, Novel feature selection method based on harmony search for email classification, Knowledge-Based Systems 73 (2015) 311–323.

[179] D. Bairathi, D. Gopalani, A novel swarm intelligence based optimization method: Harris' hawk optimization, in: International Conference on Intelligent Systems Design and Applications, Springer, 2018, pp. 832–842.

[180] Y. Zhang, R. Liu, X. Wang, H. Chen, C. Li, Boosted binary harris hawks optimizer and feature selection, structure 25 (2020) 26.

[181] J. Too, A. R. Abdullah, N. Mohd Saad, A new quadratic binary harris hawk optimization for feature selection, Electronics 8 (10) (2019) 1130.

[182] M. Abdel-Basset, W. Ding, D. El-Shahat, A hybrid harris hawks optimization algorithm with simulated annealing for feature selection, Artificial Intelligence Review (2020) 1–45.

[183] R. Sihwail, K. Omar, K. A. Z. Ariffin, M. Tubishat, Improved harris hawks optimization using elite opposition-based learning and novel search mechanism for feature selection, IEEE Access 8 (2020) 121127–121145.

[184] T. Dokeroglu, A. Deniz, H. E. Kiziloz, A robust multiobjective harris' hawks optimization algorithm for the binary classification problem, Knowledge-Based Systems (2021) 107219.

[185] A. C. Hardy, The plankton of the south georgia whaling grounds and adjacent waters, 1926-1932, Discovery Rep. 11 (1935) 1–456.

[186] A. L. Bolaji, M. A. Al-Betar, M. A. Awadallah, A. T. Khader, L. M. Abualigah, A comprehensive review: Krill herd algorithm (kh) and its applications, Applied Soft Computing 49 (2016) 437–446.

[187] D. Rodrigues, L. A. Pereira, J. P. Papa, S. A. Weber, A binary krill herd approach for feature selection, in: 2014 22nd International Conference on Pattern Recognition, IEEE, 2014, pp. 1407–1412.

[188] L. Abualigah, B. Alsalibi, M. Shehab, M. Alshinwan, A. M. Khasawneh, H. Alabool, A parallel hybrid krill herd algorithm for feature selection, International Journal of Machine Learning and Cybernetics (2020) 1–24.

[189] G. Zhang, J. Hou, J. Wang, C. Yan, J. Luo, Feature selection for microarray data classification using hybrid information gain and a modified binary krill herd algorithm, Interdisciplinary Sciences: Computational Life Sciences 12 (2020) 288–301.

[190] P. Anderson, Q. Bone, Communication between individuals in salp chains. ii. physiology, Proceedings of the Royal Society of London. Series B. Biological Sciences 210 (1181) (1980) 559–574.

[191] A. E. Hegazy, M. Makhlouf, G. S. El-Tawel, Improved salp swarm algorithm for feature selection, Journal of King Saud University-Computer and Information Sciences 32 (3) (2020) 335–344.

[192] M. Tubishat, N. Idris, L. Shuib, M. A. Abushariah, S. Mirjalili, Improved salp swarm algorithm based on opposition based learning and novel local search algorithm for feature selection, Expert Systems with Applications 145 (2020) 113122.

[193] H. Faris, M. M. Mafarja, A. A. Heidari, I. Aljarah, A.-Z. Ala'M, S. Mirjalili, H. Fujita, An efficient binary salp swarm algorithm with crossover scheme for feature selection problems, Knowledge-Based Systems 154 (2018) 43–67.

[194] A. I. Hafez, H. M. Zawbaa, E. Emary, A. E. Hassanien, Sine cosine optimization algorithm for feature selection, in: 2016 international symposium on innovations in intelligent systems and applications (INISTA), IEEE, 2016, pp. 1–5.

[195] R. Sindhu, R. Ngadiran, Y. M. Yacob, N. A. H. Zahri, M. Hariharan, Sine–cosine algorithm for feature selection with elitism strategy and new updating mechanism, Neural Computing and Applications 28 (10) (2017) 2947–2958.

[196] J. James, V. O. Li, A social spider algorithm for global optimization, Applied Soft Computing 30 (2015) 614–627.

[197] B. Emine, E. Ülker, An efficient binary social spider algorithm for feature selection problem, Expert Systems with Applications 146 (2020) 113185.

[198] R. A. Ibrahim, M. Abd Elaziz, D. Oliva, E. Cuevas, S. Lu, An opposition-based social spider optimization for feature selection, Soft Computing 23 (24) (2019) 13547–13567.

[199] M. Abd El Aziz, A. E. Hassanien, An improved social spider optimization algorithm based on rough sets for solving minimum number attribute reduction problem, Neural Computing and Applications 30 (8) (2018) 2441–2452.

[200] D. R. Pereira, M. A. Pazoti, L. A. Pereira, D. Rodrigues, C. O. Ramos, A. N. Souza, J. P. Papa, Social-spider optimization-

based support vector machines applied for energy theft detection, Computers & Electrical Engineering 49 (2016) 25–38.

[201] R. Rao, V. Patel, An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems, International Journal of Industrial Engineering Computations 3 (4) (2012) 535–560.

[202] R. V. Rao, V. J. Savsani, D. Vakharia, Teaching–learning-based optimization: an optimization method for continuous non-linear large scale problems, Information sciences 183 (1) (2012) 1–15.

[203] F. Zou, D. Chen, Q. Xu, A survey of teaching–learning-based optimization, Neurocomputing 335 (2019) 366–383.

[204] E. Sevinc, T. DÖKEROĞLU, A novel hybrid teaching-learning-based optimization algorithm for the classification of data by using extreme learning machines, Turkish Journal of Electrical Engineering & Computer Sciences 27 (2) (2019) 1523–1533.

[205] D. Pradhan, B. Sahoo, B. B. Misra, S. Padhy, A multiclass svm classifier with teaching learning based feature subset selection for enzyme subclass classification, Applied Soft Computing 96 (2020) 106664.

[206] H. E. Kiziloz, A. Deniz, T. Dokeroglu, A. Cosar, Novel multiobjective tlbo algorithms for the feature subset selection problem, Neurocomputing 306 (2018) 94–107.

[207] M. Sharawi, H. M. Zawbaa, E. Emary, Feature selection approach based on whale optimization algorithm, in: 2017 Ninth international conference on advanced computational intelligence (ICACI), IEEE, 2017, pp. 163–168.

[208] M. M. Mafarja, S. Mirjalili, Hybrid whale optimization algorithm with simulated annealing for feature selection, Neurocomputing 260 (2017) 302–312.

[209] A. G. Hussien, A. E. Hassanien, E. H. Houssein, S. Bhattacharyya, M. Amin, S-shaped binary whale optimization algorithm for feature selection, in: Recent trends in signal and image processing, Springer, 2019, pp. 79–87.

[210] F. Pourpanah, Y. Shi, C. P. Lim, Q. Hao, C. J. Tan, Feature selection based on brain storm optimization for data classification, Applied Soft Computing 80 (2019) 761–775.

[211] M. Ghaemi, M.-R. Feizi-Derakhshi, Feature selection using forest optimization algorithm, Pattern Recognition 60 (2016) 121–129.

[212] G. Chen, J. Chen, A novel wrapper method for feature selection and its applications, Neurocomputing 159 (2015) 219–226.

[213] D. Rodrigues, X.-S. Yang, A. N. De Souza, J. P. Papa, Binary flower pollination algorithm and its application to feature selection, in: Recent advances in swarm intelligence and evolutionary computation, Springer, 2015, pp. 85–100.

[214] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowledge-based systems 89 (2015) 228–249.

[215] G.-G. Wang, S. Deb, Z. Cui, Monarch butterfly optimization, Neural computing and applications 31 (7) (2019) 1995–2014.

[216] C. Yan, J. Ma, H. Luo, A. Patel, Hybrid binary coral reefs optimization algorithm with simulated annealing for feature selection in high-dimensional biomedical datasets, Chemometrics and Intelligent Laboratory Systems 184 (2019) 102–111.

[217] H. E. Kiziloz, Classifier ensemble methods in feature selection, Neurocomputing 419 (2021) 97–107.

[218] A. Kaveh, N. Farhoudi, A new optimization method: Dolphin echolocation, Advances in Engineering Software 59 (2013) 53–70.

[219] A. Kaveh, V. R. Mahdavi, Colliding bodies optimization: a novel meta-heuristic method, Computers & Structures 139 (2014) 18–27.

[220] H. Shah-Hosseini, Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimisation, International Journal of Computational Science and Engineering 6 (1-2) (2011) 132–140.

[221] M. Jain, V. Singh, A. Rani, A novel nature-inspired algorithm for optimization: Squirrel search algorithm, Swarm and evolutionary computation 44 (2019) 148–175.

[222] Y.-q. Han, J.-q. Li, Z. Liu, C. Liu, J. Tian, Metaheuristic algorithm for solving the multi-objective vehicle routing problem with time window and drones, International Journal of Advanced Robotic Systems 17 (2) (2020) 1729881420920031.

[223] G.-G. Wang, Y. Tan, Improving metaheuristic algorithms with information feedback models, IEEE transactions on

cybernetics 49 (2) (2017) 542–555.

[224] H. Xu, B. Xue, M. Zhang, Segmented initialization and offspring modification in evolutionary algorithms for bi-objective feature selection, in: Proceedings of the 2020 Genetic and Evolutionary Computation Conference, 2020, pp. 444–452.

[225] A. Deniz, H. E. Kiziloz, On initial population generation in feature subset selection, Expert Systems with Applications 137 (2019) 11–21.

[226] R. Salgotra, U. Singh, S. Saha, New cuckoo search algorithms with enhanced exploration and exploitation properties, Expert Systems with Applications 95 (2018) 384–420.

[227] E. Cuevas, A. Echavarría, M. A. Ramírez-Ortegón, An optimization algorithm inspired by the states of matter that improves the balance between exploration and exploitation, Applied intelligence 40 (2) (2014) 256–272.

[228] R. T. Marler, J. S. Arora, Survey of multi-objective optimization methods for engineering, Structural and multidisciplinary optimization 26 (6) (2004) 369–395.

[229] Y. Sawaragi, H. NAKAYAMA, T. TANINO, Theory of multiobjective optimization, Elsevier, 1985.

[230] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: A survey of the state of the art, Swarm and Evolutionary Computation 1 (1) (2011) 32–49.

[231] P. M. Narendra, K. Fukunaga, A branch and bound algorithm for feature subset selection, IEEE Transactions on computers C-26 (9) (1977) 917–922.

[232] J. Yang, V. Honavar, Feature subset selection using a genetic algorithm, in: Feature extraction, construction and selection, Springer, 1998, pp. 117–136.

[233] Y. Zhang, D.-w. Gong, X.-z. Gao, T. Tian, X.-y. Sun, Binary differential evolution with self-learning for multi-objective feature selection, Information Sciences 507 (2020) 67–85.

[234] X.-h. Wang, Y. Zhang, X.-y. Sun, Y.-l. Wang, C.-h. Du, Multi-objective feature selection based on artificial bee colony: An acceleration approach with variable sample size, Applied Soft Computing 88 (2020) 106041.

[235] Y. Hu, Y. Zhang, D. Gong, Multiobjective particle swarm optimization for feature selection with fuzzy cost, IEEE Transactions on Cybernetics.

[236] A.-D. Li, B. Xue, M. Zhang, Multi-objective feature selection using hybridization of a genetic algorithm and direct multisearch for key quality characteristic selection, Information Sciences.

[237] W. Ghanem, A. Jantan, Novel multi-objective artificial bee colony optimization for wrapper based feature selection in intrusion detection, International journal of advance soft computing applications 8 (1).

[238] P. A. Castro, F. J. Von Zuben, Multi-objective feature selection using a bayesian artificial immune system, International Journal of Intelligent Computing and Cybernetics.

[239] X. Li, M. Yin, Multiobjective binary biogeography based optimization for feature selection using gene expression data, IEEE Transactions on NanoBioscience 12 (4) (2013) 343–353.

[240] D. Rodrigues, V. H. C. de Albuquerque, J. P. Papa, A multi-objective artificial butterfly optimization approach for feature selection, Applied Soft Computing 94 (2020) 106442.

[241] C. Blum, J. Puchinger, G. R. Raidl, A. Roli, Hybrid metaheuristics in combinatorial optimization: A survey, Applied soft computing 11 (6) (2011) 4135–4151.

[242] C. Blum, A. Roli, Hybrid metaheuristics: an introduction, in: Hybrid Metaheuristics, Springer, 2008, pp. 1–30.

[243] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, R. Qu, Hyper-heuristics: A survey of the state of the art, Journal of the Operational Research Society 64 (12) (2013) 1695–1724.

[244] E. Zorarpacı, S. A. Özel, A hybrid approach of differential evolution and artificial bee colony for feature selection, Expert Systems with Applications 62 (2016) 91–103.

[245] P. Du, J. Wang, Y. Hao, T. Niu, W. Yang, A novel hybrid model based on multi-objective harris hawks optimization algorithm for daily pm2. 5 and pm10 forecasting, Applied Soft Computing 96 (2020) 106620.

[246] R. A. Ibrahim, A. A. Ewees, D. Oliva, M. Abd Elaziz, S. Lu, Improved salp swarm algorithm based on particle swarm optimization for feature selection, Journal of Ambient Intelligence and Humanized Computing 10 (8) (2019) 3155–3169.

[247] N. Neggaz, A. A. Ewees, M. Abd Elaziz, M. Mafarja, Boosting salp swarm algorithm by sine cosine algorithm and disrupt operator for feature selection, Expert Systems with Applications 145 (2020) 113103.

[248] S. Arora, H. Singh, M. Sharma, S. Sharma, P. Anand, A new hybrid algorithm based on grey wolf optimization and crow search algorithm for unconstrained function optimization and feature selection, Ieee Access 7 (2019) 26343–26361.

[249] J. Lee, D.-W. Kim, Memetic feature selection algorithm for multi-label classification, Information Sciences 293 (2015) 80–96.

[250] H. Chen, A. A. Heidari, H. Chen, M. Wang, Z. Pan, A. H. Gandomi, Multi-population differential evolution-assisted harris hawks optimization: Framework and case studies, Future Generation Computer Systems.

[251] M. M. Mafarja, S. Mirjalili, Hybrid binary ant lion optimizer with rough set and approximate entropy reducts for feature selection, Soft Computing 23 (15) (2019) 6249–6265.

[252] M. Sarhani, A. El Afia, R. Faizi, Facing the feature selection problem with a binary pso-gsa approach, in: Recent developments in metaheuristics, Springer, 2018, pp. 447–462.

[253] A. I. Hafez, A. E. Hassanien, H. M. Zawbaa, E. Emary, Hybrid monkey algorithm with krill herd algorithm optimization for feature selection, in: 2015 11th international computer engineering conference (ICENCO), IEEE, 2015, pp. 273–277.

[254] L. Wang, F. Zou, X. Hei, D. Yang, D. Chen, Q. Jiang, Z. Cao, A hybridization of teaching–learning-based optimization and differential evolution for chaotic time series prediction, Neural computing and applications 25 (6) (2014) 1407–1422.

[255] S. Deb, X.-Z. Gao, K. Tammi, K. Kalita, P. Mahanta, A new teaching–learning-based chicken swarm optimization algorithm, Soft Computing 24 (7) (2020) 5313–5331.

[256] D. Oliva, S. Hinojosa, M. Abd Elaziz, N. Ortega-Sánchez, Context based image segmentation using antlion optimization and sine cosine algorithm, Multimedia Tools and Applications 77 (19) (2018) 25761–25797.

[257] B. K. Kihel, S. Chouraqui, Firefly optimization using artificial immune system for feature subset selection, Int. J. Intell. Eng. Syst 12 (4) (2019) 337–347.

[258] M. Ghetas, C. H. Yong, P. Sumari, Harmony-based monarch butterfly optimization algorithm, in: 2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE), IEEE, 2015, pp. 156–161.

[259] P. Kora, S. R. Kalva, Hybrid bacterial foraging and particle swarm optimization for detecting bundle branch block, SpringerPlus 4 (1) (2015) 1–19.

[260] S. S. Shreem, S. Abdullah, M. Z. A. Nazri, Hybrid feature selection algorithm using symmetrical uncertainty and a harmony search algorithm, International Journal of Systems Science 47 (6) (2016) 1312–1329.

[261] M. Nekkaa, D. Boughaci, Hybrid harmony search combined with stochastic local search for feature selection, Neural Processing Letters 44 (1) (2016) 199–220.

[262] S. P. Das, S. Padhy, A novel hybrid model using teaching–learning-based optimization and a support vector machine for commodity futures index forecasting, International Journal of Machine Learning and Cybernetics 9 (1) (2018) 97–111.

[263] C. Yogesh, M. Hariharan, R. Ngadiran, A. H. Adom, S. Yaacob, C. Berkai, K. Polat, A new hybrid pso assisted biogeography-based optimization for emotion and stress recognition from speech signal, Expert Systems with Applications 69 (2017) 149–158.

[264] L. Kumar, K. K. Bharti, A novel hybrid bpso–sca approach for feature selection, Natural Computing (2019) 1–23.

[265] A. M. Anter, M. Ali, Feature selection strategy based on hybrid crow search optimization algorithm integrated with chaos theory and fuzzy c-means algorithm for medical diagnosis problems, Soft Computing 24 (3) (2020) 1565–1584.

[266] M. Montazeri, Hhfs: Hyper-heuristic feature selection, Intelligent Data Analysis 20 (4) (2016) 953–974.

[267] R. Hunt, K. Neshatian, M. Zhang, A genetic programming approach to hyper-heuristic feature selection, in: Asia-Pacific Conference on Simulated Evolution and Learning, Springer, 2012, pp. 320–330.

[268] B. Abdollahzadeh, F. S. Gharehchopogh, A multi-objective optimization algorithm for feature selection problems, Engineering with Computers (2021) 1–19.

[269] N. Dif, Z. Elberrichi, A novel dynamic hybridization method for best feature selection, International Journal of Applied Metaheuristic Computing (IJAMC) 12 (2) (2021) 85–99.

[270] D. Dua, C. Graff, UCI machine learning repository (2017).
URL http://archive.ics.uci.edu/ml

[271] S. B. Taieb, R. J. Hyndman, A gradient boosting approach to the kaggle load forecasting competition, International journal of forecasting 30 (2) (2014) 382–394.

[272] A. Statnikov, I. Tsamardinos, Y. Dosbayev, C. F. Aliferis, Gems: a system for automated cancer diagnosis and biomarker discovery from microarray gene expression data, International journal of medical informatics 74 (7-8) (2005) 491–503.

[273] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM transactions on intelligent systems and technology (TIST) 2 (3) (2011) 1–27.

[274] N. Hansen, A. Auger, R. Ros, S. Finck, P. Pošík, Comparing results of 31 algorithms from the black-box optimization benchmarking bbob-2009, in: Proceedings of the 12th annual conference companion on Genetic and evolutionary computation, 2010, pp. 1689–1696.

[275] X.-S. Yang, Nature-inspired optimization algorithms: Challenges and open problems, Journal of Computational Science 46 (2020) 101104.

[276] E. Alba, Parallel metaheuristics: a new class of algorithms, Vol. 47, John Wiley & Sons, 2005.

[277] E. Alba, G. Luque, S. Nesmachnow, Parallel metaheuristics: recent advances and new trends, International Transactions in Operational Research 20 (1) (2013) 1–48.