Cognitive Computation
https://doi.org/10.1007/s12559-023-10149-0

Cognitively Enhanced Versions of Capuchin Search Algorithm for Feature Selection in Medical Diagnosis: a COVID-19 Case Study

Malik Braik[1] · Mohammed A. Awadallah[2,3] · Mohammed Azmi Al-Betar[3,4] · Abdelaziz I. Hammouri[1] · Omar A. Alzubi[1]

Received: 20 June 2022 / Accepted: 28 April 2023

Abstract
Feature selection (FS) is a crucial area of cognitive computation that demands further studies. It has recently received a lot of attention from researchers working in machine learning and data mining. It is broadly employed in many different applications. Many enhanced strategies have been created for FS methods in cognitive computation to boost the performance of the methods. The goal of this paper is to present three adaptive versions of the capuchin search algorithm (CSA) that each features a better search ability than the parent CSA. These versions are used to select optimal feature subset based on a binary version of each adapted one and the k-Nearest Neighbor (k-NN) classifier. These versions were matured by applying several strategies, including automated control of inertia weight, acceleration coefficients, and other computational factors, to ameliorate search potency and convergence speed of CSA. In the velocity model of CSA, some growth computational functions, known as exponential, power, and S-shaped functions, were adopted to evolve three versions of CSA, referred to as exponential CSA (ECSA), power CSA (PCSA), and S-shaped CSA (SCSA), respectively. The results of the proposed FS methods on 24 benchmark datasets with different dimensions from various repositories were compared with other k-NN based FS methods from the literature. The results revealed that the proposed methods significantly outperformed the performance of CSA and other well-established FS methods in several relevant criteria. In particular, among the 24 datasets considered, the proposed binary ECSA, which yielded the best overall results among all other proposed versions, is able to excel the others in 18 datasets in terms of classification accuracy, 13 datasets in terms of specificity, 10 datasets in terms of sensitivity, and 14 datasets in terms of fitness values. Simply put, the results on 15, 9, and 5 datasets out of the 24 datasets studied showed that the performance levels of the binary ECSA, PCSA, and SCSA are over 90% in respect of specificity, sensitivity, and accuracy measures, respectively. The thorough results via different comparisons divulge the efficiency of the proposed methods in widening the classification accuracy compared to other methods, ensuring the ability of the proposed methods in exploring the feature space and selecting the most useful features for classification studies.

Keywords Feature selection · Capuchin search algorithm · Transfer function · Optimization

✉ Malik Braik
mbraik@bau.edu.jo

Mohammed A. Awadallah
ma.awadallah@alaqsa.edu.ps

Mohammed Azmi Al-Betar
m.albetar@ajman.ac.ae

Abdelaziz I. Hammouri
aziz@bau.edu.jo

Omar A. Alzubi
o.zubi@bau.edu.jo

[1] Department of Computer Science, Al-Balqa Applied University, Salt, Jordan

[2] Department of Computer Science, Al-Aqsa University, P.O. Box 4051, Gaza, Palestine

[3] Artificial Intelligence Research Center (AIRC), College of Engineering and Information Technology, Ajman University, Ajman, United Arab Emirates

[4] Department of Information Technology, Al-Huson University College, Al-Balqa Applied University, Al-Huson, Irbid, Jordan

Published online: 05 June 2023

Introduction

The Internet is currently being adopted by people, governments, and institutions all over the world in almost all aspects of life. Thereby, a large amount of data is generated every day in a wide variety of forms, with these datasets typically serving as an extension of knowledge [1]. In this regard, society may create data from a broad range of sectors, including health, agriculture, and industry, among many others [2, 3]. These datasets may be categorized and then used for information, forecasts, and insights. Due to the favors of rapidly advancing data collection and storage technologies, organizations and governments often collect and use these vast volumes of data on a regular basis. Raw datasets are frequently fruitless and possibly unusable unless a proper automated method is used to extract useful information from them. In any case, getting usable information has proved to be a very difficult task [4]. Traditional data analysis techniques often fail when attempting to analyze huge amounts of datasets. Even when the dataset is very small, the atypical form of the data might make it challenging to employ traditional methods to effectively process and analyze such datasets. In many cases, the problems that need to be addressed cannot be solved with the existing data analysis methods, and so new methods have to be taken [5]. Based on these discussions, the key principles of data mining (DM) can be used to identify the observable patterns in unprocessed datasets [6]. DM approaches especially attempt to learn about various features within the data by removing and simulating the content of the data. In this, DM is a generic process that involves a number of transformation operations, starting from pre-processing the data and ending with post-processing the output produced by a pre-built DM technique [7]. Due to some problems inherent in the raw datasets, which may often include redundant, irrelevant, or unimportant data [8], these datasets cannot then be utilized directly for post-processing steps. In this, preprocessing steps must be used to the gathered data to clean it up and get it ready for further phases of machine learning (ML) methods [7]. Data pre-processing is perhaps the most prolonged and difficult phase of the knowledge detection process due to the variety of ways that may be used to collect data. From these angles, this work specifically focuses on feature selection (FS) to confront with the above obstacles.

Feature selection is an important area of study in ML tasks, where ML is one of the key areas of cognitive computation. In many ML tasks, high dimensionality, on the one side, augments the information of data, on the other side, leads to curse of dimensionality problem [6]. In point of fact, for a lot of real-world applications, a small subset of informative and discriminate features can act better than employing all of the features. Data dimensionality reduction [9, 10] may be thought of as a cognitive method for analyzing the inherent properties of data. Feature selection [1, 7], favorably, can tackle the problems of high dimensions by lowering the dimensionality in ML tasks. It poses one of the most important pre-steps, which aims to remove duplicate or irrelevant data from the dataset that will be analyzed [11]. On this account, the benefits of FS stretch from data dimensionality and reducing over-fitting to eliminating noisy data, ameliorating classification accuracy, speeding up the model's learning cycle, and reducing complexities within the dataset, among many other merits of FS methods [12]. Due to the above blessings, FS techniques have become active research areas and have been properly used in various fields such as facial recognition [13], image classification [14], and micro-array analysis [15].

Recently, many COVID-19 cases have been collected, and the dataset has been established [16]. Regardless of the location, cases volume, pandemic wave, or time of the collected COVID-19 data, the gathered dataset consists of fifteen features, and FS methods attempt to locate the subset of the most informative features. This subset will be used in machine learning or deep learning methods for classification purposes. Moreover, some methods have used COVID-19 data to discover chronic lung diseases. This is to estimate the severity and mortality rate among COVID-19 patients [17–19]. As of now, there more than 900 million people in China who have been infected with COVID-19, and that number is rising by the millions every day. Thus, China immediately ceased providing daily COVID figures and abandoned its zero COVID policies. An increase in COVID cases is also expected in rural China this year. It is expected that the COVID wave in China would peak in 2 to 3 months. COVID-19 and chronic obstructive pulmonary disease (COPD) have several potential adverse interrelations that may influence infection course and clinical results. There are some mechanisms that may be considered for increased COVID-19 infection susceptibility in COPD such as ineffective immunity and decreased antiviral defense [20]. COPD is linked with worse clinical results from COVID-19 [20], where COVID-19 has a large impact on the habitual care of COPD patients. There is evidence that COPD patients have worse outcomes from COVID-19 [18, 19]. A result of COVID-19 has been isolation and increased anxiety in COPD patients, with conceivably deleterious long-term repercussions [20].

Returning to the essence of this study, in general, during the implementation of the FS process, a subset of the candidate features is selected from the original feature set, and its relevance is then measured by an evaluation criterion. The process of selecting and evaluating a subset of features

is repeated until a preset stopping condition is met. Then, the best obtained subset is validated in the test dataset [21]. Two opposing goals are used in feature selection methods to optimize the search in the search space, namely, to reduce the redundancy of the selected features and to increase the relevance of the class label [22]. Various search techniques could be used to deal with these goals. These techniques can be classified as single-objective and multi-objective methods [23]. In the single-objective FS methods, the solution is enhanced by evaluating a particular objective function. Therefore, the used objective function will affect the quality of the solution obtained from the optimization algorithm. Moreover, there is no specific objective that can be suitable for all optimization problems. Thus, defining a fitness function for optimizing a single objective can lessen the performance of the optimization method. Consideration of different conflicting goals in a fitness function can overcome the above obstacles and may return a non-dominated set of solutions, which can be several subsets of feature that meet different objectives. The chief handicap of multi-objective optimization methods is the increased complexity of the search space [24]. Furthermore, the intricacy of the majority of real-world problems are further challenges of FS methods. This, in turn, requires a large amount of solution space due to the dependency and non-linear needs between the dataset characteristics [6]. Examining each subset produced during the generation of various subsets is necessary in order to determine which subsets best suit assessment techniques like maximizing classification rate or reducing error [25]. This method is computationally costly and cumbersome, especially for datasets with high-dimensions. In such a case, creating all possible subsets of high dimensional datasets becomes impractical and computationally costly. Hence, dealing with such complex problems is difficult using conventional FS methods. These and other difficulties have led researchers to investigate many different strategies to get excellent performance levels in classification purposes [26]. Thus, the application of meta-heuristics has been targeted in the search for improved solutions to FS problems with an optimal rate of performance [5]. These techniques have generally proven to be effective in tackling optimization real-world problems in reasonable amounts of time with minimal computing effort in a wide range of engineering and science fields.

From the standpoint of cognitive computation, the challenges posed by high dimensionality problems in ML tasks may be overcome by adopting highly reliable FS approaches as well as robust classification models that learn from data. In this study, a newly developed meta-heuristic, named capuchin search algorithm (CSA) [27], was adopted to solve broadly available feature selection problems in the field of medical diagnosis. Although CSA has the ability to get the optimal solution in solving diverse problems in the optimization field [28, 29], it is customarily confined to the local optima especially when it encounters complex problems with many local optimums. This may be ascribed to its narrow search ability and modest convergence property. Hence, this study made some improvements to the basic CSA to efficiently solve such convoluted FS problems. This is developed in keeping with the ideology of continuous improvements to come up with highly powerful solutions to real-world problems. This is the first and main motive for this work. The basic CSA has two essential parameters in the velocity updating model, referred to as cognitive and social parameters, which help the capuchins to reach the optimal solution. However, these parameters are constants during the iterative process of CSA, which may impair exploration and exploitation features when addressing hard optimization problems. As well-adapted cognitive and social parameters as well as an efficient inertia weight mechanism are expected to influence the performance of CSA, it is worth noting and investigating to enhance CSA from the point of view of using adaptive strategies for these parameters. Based on the underlying CSA, a promising velocity updating model with proper control parameters can be implemented to guide the local and global search stages of the capuchins in the surrounding environment. In this respect, three improved versions of CSA were developed to deal with the early convergence and low search ability of CSA. Each version has added a reasonable improvement to the parent CSA by adopting different growth functions to update the values of the cognitive and social models during the path iterations of CSA. These versions are called exponential CSA (ECSA), power CSA (PCSA), and S-shaped CSA (PCSA). Subsequently, a new inertia weight was proposed for all of these versions to further control the velocity model. This improvement is intended to empower these versions to have more exploration and exploitation abilities. These functions not only provide effective guidance for the capuchins in the search areas, but they are also useful in alleviating the stagnancy of CSA. As the optimization frameworks of the proposed and basic variants of CSA are continuous, they can only handle continuous search spaces, but they have trouble in tackling problems with binary search spaces. In light of this, binary versions of these variants were created by adjusting the key operators and parameters of these variants to align with the nature of the search space of FS problems. This is the second motive of the current work. In this work, we address the shortcomings of CSA based on cognitive models to provide efficient and reliable optimization algorithms. To strengthen the efficacy of the proposed methods on FS problems, we expand the work utilizing growth computation models. For many of the datasets considered in this study, we found that the proposed FS methods when in combination with adaptive inertia weight during the iteration process of these methods can consistently provide higher performance levels than

other FS methods. To sum up, the theoretical contributions of the proposed work can be recapped as follows:

- Three enhanced binary versions of CSA were proposed and applied together with the basic binary CSA to solve a variety of 24 datasets collected from the UCI machine learning repository.
- Three new cognitive and three new social models were embedded into the proposed methods to improve the diversity of solutions and increase the balance between exploration and exploitation features to promote the best-found solutions.
- The performance of the evolved FS algorithms were compared with other highly effective FS methods in terms of several relevant criteria.

The reset of this work is arranged as follows: a literature review of several feature selection methods is presented in the "Related Works" section. The "Basic Capuchin Search Algorithm" section provides a brief description of the parent CSA. The following "Proposed Algorithms of CSA" section presents in detail the proposed algorithms. Next, the "Proposed Algorithms for Feature Selection" section describes the binary versions of the proposed feature selection methods. In the "Experimental Results and Discussions" section, the experimental results are presented and discussed. Finally, conclusions and several future directions are provided in the "Conclusion and Future Works" section.

Related Works

More recently, meta-heuristic algorithms have been widely used by many researchers to address different kinds of FS problems of varied levels of complexity [4, 5]. Astonishingly, meta-heuristics have reported notable advantages and delivered impressive accuracy when used as wrapper-based methods for solving FS problems. Well-known classes of meta-heuristics including swarm intelligence, evolutionary algorithms, and physics-based algorithms, as well as several hybridization algorithms that combine two algorithms of the same class or different classes, have been used to solve FS problems [6]. The following is a review of some selected FS methods classified according to the class of algorithms used, which have reported promising performance in the literature.

Swarm Intelligence-Based FS

There are many prominent examples of applications of swarm intelligence (SI) algorithms as search methods for wrapper-based approaches to solve FS problems in different domains [2, 4, 5, 30]. An efficient study for solving FS problems was presented by Arora et al. [2]. In that study, Arora et al. evolved two diversified binary variants of butterfly optimization algorithm (BBOA). The S- and V-shape transfer functions were applied to generate the two binary versions of BOA. These versions were assessed on 21 datasets collected from the UCI repository. It was found in [2] that BBOA with S-shape is better than BBOA with V-shape as well as many other similar FS algorithms in all evaluation methods and all studied datasets. Xian-Fang et al. [30] evolved a three-stage FS method as follows: (1) In the first stage, irrelevant features were eliminated using *C*-relevance; (2) in the second stage, the *k*th feature cluster was used to collect analogous features in the same cluster; and (3) an improved version of particle swarm optimization (PSO) was used to determine the optimal feature set. This algorithm, referred to as HFS-C-P, was assessed on 18 datasets collected from public repositories. Xian-Fang et al. stated that the HFS-C-P algorithm achieved promising results, in respect of fitness score, number of selected features, and computational time, in all considered datasets better than those achieved by other algorithms. In a more recent and effective work on FS problems, an improved binary variant of the rat swarm optimizer (RSO) combined with the local search paradigm of PSO was proposed [4]. In this method, three crossover mechanisms, controlled by a switch probability, were embedded with RSO to improve the diversity of its solutions. This method was examined on 24 datasets collected from various repositories and assessed using several evaluation methods. While this method revealed promising levels of performance, its convergence rate is a little modest. Last but not least, another good work for solving FS problems is presented in [5] using a binary version of Horse herd Optimization Algorithm, referred to as BHOA. In this algorithm, three transfer functions, namely, S-shape, V-shape, and U-shape, were used to obtain the binary domain of the HOA, where these variants were integrated with three types of crossover mechanisms to produce fifteen different variants of the BHOA. The performance of these versions was examined on 24 real-world datasets and evaluated using a set of six metric measures. The best-formed version of the proposed versions is a BHOA with an S-shape and a one-point crossover. A comparative evaluation was conducted against 21 FS methods. The BHOA method was able to find very competitive results against these comparative methods, but the implementation of the 15 versions of BHOA demands a large computational burden.

Evolutionary Algorithms-Based FS

Evolutionary algorithms (EAs) represent another broad class of meta-heuristics inspired by the natural processes of evolution. These algorithms have been broadly adapted to mature appropriate approaches for solving FS problems with a promising degree of accuracy using low computational

burden and a small number of selected features [31, 32]. Appropriately, many variants of EAs, such as genetic algorithm (GA) [33], genetic programming (GP) [31], and binary differential evolution (DE) [32], have been used in the literature to tackle several types of FS problems. Recently, Awadallah et al. [34] developed a FS method based on a binary version of the JAYA algorithm, denoted as BJAM, with the help of an adaptive mutation operator. This operator was used to diversify the population during the iterative process, where this operator was managed using a pre-defined mutation rate. The JAYA algorithm was converted to binary utilizing an S-shape transfer function. The BJAM algorithm was assessed on 22 datasets selected from the UCI repository, where a good performance degree was realized compared to other FS methods.

Physics-Based Algorithms-Based FS

Simulated annealing (SA) [35] is one of the popular and broadly physics-based (PB) algorithms used to tackle FS problems. In [36], SA was used as a FS method for detecting various denial of service attacks. Several hybridization methods of PB algorithms with SI algorithms or EAs have been presented in the literature to solve FS problems. For example, a hybridization method of the whale optimization algorithm (WOA) with SA was used to solve FS problems for 18 datasets taken from the UCI repository [37]. In this method, SA was utilized to improve neighborhood search ability of WOA, where WOA was used to ameliorate the exploitation ability of the native algorithms. The performance of this hybrid model is promising and superior to the parent and other algorithms. Other examples of hybrid-based FS methods include a hybridization of a binary coral reefs algorithm with SA [38] and a binary spotted hyena algorithm with SA [39]. There are many other hybrid FS methods evolved in the literature such as a hybrid approach of genetic algorithms and artificial bee colony (ABC) [40] and a hybrid approach of Harris Hawks optimization algorithm with SA [41]. These hybrid models-based FS problems have achieved acceptable performance, but were subject to significant computations and complexity. There are many other researchers who have combined wrapper-based with filter-based methods to solve FS problems as discussed below.

Hybrid Filter-Wrapper Model-Based FS

Hybridization of filter and wrapper-based methods has been extensively used in the literature to strengthen the performance of FS tasks [42]. These hybrid models mainly comprise of two stages: the first stage is carried out based on a filter method to select the most important features, while the second stage is implemented based on a wrapper method in order to select a subset of features based on these selected features. For example, hybridization of mutual information (MI) as a filter method and binary cuckoo search as a wrapper method was presented in [43] to solve FS problems. Lai et al. [44] evolved a hybrid FS model using information gain (IG) as a filter method and an improved simplified swarm optimization (ISSO) as a wrapper method. In this method, IG was applied to select the most important features representing genes, while ISSO was applied to search for the optimal subset of genes. Another hybrid filter-wrapper method gene selection from microarray data for gene expression is reported in [45], where improved swarm-optimization was implemented as a wrapper-based method. The above hybrid filter-wrapper models for FS problems revealed reasonable performance. However, the filter methods may prevent some important features despite their sensible performance. Also, the wrapper methods that used some meta-heuristics may suffer from poor stability [46], where random search affected the stability of the selected features. A promising FS technique using a hybridization of binary biogeography optimization (BBO) with support vector machine recursive feature elimination (SVM-RFE), known as BBO-SVM-RFE, was developed in [6]. The SVM-RFE is embedded into the BBO to improve the quality of the obtained solutions in the mutation operator in order to reinforce the exploitation capability as well as to strike an adequate balance between exploitation and exploration of the original BBO. The BBO-SVM-RFE was assessed on 18 benchmark datasets. Comparative results showed that BBO-SVM-RFE revealed a wise degree of performance in terms of accuracy and number of selected features against other existing FS methods. However, the structure of BBO-SVM-RFE is complex and has slow convergence behavior, where the optimal solutions require high computational efforts. While the aforesaid feature selection methods realized promising levels of performance in a fair-minded time in addressing the FS problems deemed in the aimed applications and datasets, they cannot ensure that in all experimental runs, the optimal solutions will be identified. This insinuates that locating the optimal feature subset is not ensured. Moreover, they demonstrated that they can address FS problems that were taken into account in their studies by identifying a minimal number of attributes from a selected subset. However, each of these FS methods behaved properly in the problems considered and might fell short in other real-world FS problems, especially those with high-dimensional datasets. Besides, some FS methods such as the one reported in [4] suffered from large computational time and local optimums. Also, the method presented in [5] provided sensible results in some datasets but not for all considered datasets in that study. The deficiencies of the above FS systems might be attributed to the possibility that meta-heuristic algorithms may

fall into local optimum solutions, especially when tackling complex FS problems with varied degrees of complexity and high dimensionality. Hence, there is still a need for further amelioration on FS methods, particularly for datasets with a large number of features and a high level of complexity. This motivated this study to strengthen the performance of the basic CSA to deal with FS problems of high complexity. This is carried out by developing three improved variants of this basic algorithm and using binary versions of the core and proposed algorithms of CSA to explore their capacities and efficiencies in handling familiar FS problems with different numbers of features, samples, and dimensions.

Basic Capuchin Search Algorithm

CSA is a new swarm intelligence algorithm developed to imitate the foraging activity and locomotion practices of capuchins while roaming in forests. The population of CSA is divided into two bunches: leaders (i.e., alpha capuchins) and followers (i.e., the remaining capuchins). Leaders lead the followers, where they go after each other and the leaders directly or indirectly. Leaders are accountable for locating food sources for themselves and the other capuchins in the group. The other capuchins (i.e., followers) update their position by pursuing the leaders. As reported in [27], leaders employ the following strategies of movements while foraging, which can be presented as shown below:

- The leaders' positions when jumping on trees are determined as follows:

$$x_j^i = F_j + \frac{P_{bf}(v_j^i)^2 sin(2\theta)}{g} \quad i < n/2; \quad 0.1 < rand \leq 0.25 \tag{1}$$

where x_j^i denotes the current position of the leaders at dimension j, F_j is the position of the food of the capuchins found so far at dimension j, P_{bf} indicates the equilibrium probability provided by the capuchins' tails which is equal to 0.75, v_j^i is the current velocity of the ith capuchin at dimension j which is defined in Eq. 3, θ is the capuchins' leaping angle defined in Eq. 2, g is the force of gravity which is equal to 9.81, n is the number of capuchins, and *rand* is a random value generated in the interval [0, 1].

$$\theta = \frac{3}{2}r \tag{2}$$

where r is a random value produced in the range [0, 1].

$$v_j^i = \rho v_j^i + a_1\left(x_{best_j}^i - x_j^i\right)r_1 + a_2\left(F_j - x_j^i\right)r_2 \tag{3}$$

where $x_{best_j}^i$ stands for the best position of capuchin i at dimension j, a_1 and a_2 are positive values that are equal to 1.5 and 1.5, respectively, r_1 and r_2 are random values in the interval [0, 1], and ρ stands for the inertia weight of the velocity defined as given by Eq. 4.

$$\rho = w_{max} - \left(w_{max} - w_{min}\right)\frac{k}{K} \tag{4}$$

where k is the iteration index representing the current number of iterations, K is a predefined maximum number of iterations, and w_{min} and w_{max} are the minimum and maximum weight values that were set to 0.2 and 0.9, respectively.

- The leaders' positions while foraging on the banks of rivers using the jumping strategy can be determined as follows:

$$x_j^i = F_j + \frac{P_{ef}P_{bf}(v_j^i)^2 sin(2\theta)}{g} \quad i < n/2; \quad 0.25 < rand \leq 0.50 \tag{5}$$

where P_{ef} stands for the elasticity probability of capuchins' motion on the ground which is equal to 9.

- The leaders' position while foraging on the ground using normal walking can be decided as follows:

$$x_j^i = x_j^i + v_j^i \quad i < n/2; \quad 0.5 < rand \leq 0.70 \tag{6}$$

- The leaders' position while swaying on trees can be decided as follows:

$$x_j^i = F_j + \tau P_{bf} \times sin(2\theta) \quad i < n/2; \quad 0.7 < rand \leq 0.8 \tag{7}$$

where τ is a dominant parameter defined in Eq. 8.

$$\tau = 2e^{-21(\frac{k}{K})^2} \tag{8}$$

- The leaders' position while climbing trees can be decided as follows:

$$x_j^i = F_j + \tau P_{bf}(v_j^i - v_{j-1}^i) \quad i < n/2; \quad 0.80 < rand \leq 1.0 \tag{9}$$

where v_{j-1}^i is the former velocity of capuchin i at dimension j.

- The random movement of leaders while foraging can be decided as follows:

$$x_j^i = \tau \times \left[lb_j + rand \times (ub_j - lb_j)\right] \quad i < n/2; \quad rand \leq Pr \tag{10}$$

where Pr denotes the random search probability of the leaders that has a value of 0.1, and ub_j and lb_j stand for the upper and lower limits of the search space at dimension j.

The followers' positions can be updated as per Eq. 11:

$$x_j^i = \frac{1}{2}\left(\hat{x}_j^i + x_j^{i-1}\right) \quad n/2 \leq i \leq n \tag{11}$$

where x_j^{i-1} and x_j^i stand for the former and current position of the followers at dimension j, respectively, and $\hat{x}_j^i$ represents the current leaders' position at dimension j.

Each new solution for each capuchin's position is assessed using a pre-defined fitness criterion. The optimization process of CSA can be implemented through iterative steps, whereby capuchins' positions are evaluated and updated. These steps are reiterated at each iteration, through which the convergence behavior can be got when the maximum number of iterations is realized. Algorithm 1 presents a short illustration of the iterative steps of CSA.

Algorithm 1 A pseudo-code summarizing the iterative steps of CSA.

```
Initialize the key parameters of CSA
Initialize the positions and velocities of all n capuchins
while (K is not reached) do
    for i = 1 to n do
        Update the leaders' velocity using Eq. 3
    end for
    for i = 1 to n do
        if (i < n/2) then
            if (rand ≥ 0.1) then
                if (rand ≤ 0.25) then
                    Update the leaders' positions using Eq.1
                else if (0.25 < rand ≤ 0.50) then
                    Update the leaders' positions using Eq. 5
                else if (0.5 < rand ≤ 0.70) then
                    Update the leaders' positions using Eq. 6
                else if (0.70 < rand ≤ 0.8) then
                    Update the leaders' positions using Eq. 7
                else if (0.80 < rand ≤ 1.0) then
                    Update the leaders' positions using Eq. 9
                end if
            else
                Update the leaders' positions using Eq. 10
            end if
        else
            Update the followers' positions using Eq. 11
        end if
    end for
end while
```

As CSA has affirmed its reliability and efficacy in addressing a lot of broadly well-known real-world problems [27, 28], we concluded that CSA could be an appropriate alternative algorithm to avail as a feature selection method.

Proposed Algorithms of CSA

Issues of CSA

Although CSA can search for optimal solutions while solving optimization problems, its search ability is limited by its original mathematical model and the defects of the velocity update model, where these flaws often lead to local optima in addressing complex optimization problems [27, 28]. The reason is that capuchins in CSA update their velocities toward food sources by relying on constant social and cognitive models for locomotion and repetitive foraging. However, these fixed values for such key parameters cannot guarantee that CSA can escape stagnation or that it is not confined to local optima. In addition, CSA challenges another issue of feeble exploration and exploitation competencies. This is obviously faced as a result of updating the original position of the capuchins in CSA, which does not take into account the control of key parameters of the velocity model during the iterative process. Therefore, there must be some strategies that help update the velocity model of the capuchins as well as fine-tuning of the key parameters in CSA. Furthermore, the exploration aptitude of CSA is insufficient in the inception search stage, so its modest exploitation causes the difficulty of finding the global solution in the late search stage. In this case, local optima is usually received. Thus, a reasonable compromise must be made between exploration and exploitation in order to enhance the search capacity of CSA. For this purpose, a new mechanism is needed to reinforce the swarming behavior among capuchins, in which the best ones play a spirited role in leading others. In this, the best capuchins can help other capuchins to avoid premature convergence once they are bounded into local optima, and without any capability to prevent or deal with this circumstance. To deal with the aforementioned issues in CSA, an update was made to the velocity model of the basic CSA that uses adaptive social and cognitive models, with the goal of improving the performance score of CSA. The following subsections provide detailed descriptions of the proposed variants of CSA, referred to as exponential CSA (ECSA), power CSA (PCSA), and S-shaped CSA (SCSA).

Exponential Model of CSA (ECSA)

During the iterative process of CSA, the characteristics of the population distribution vary not only with iteration number but with the iterative state as well. For example, at a premature stage, the capuchins may be dispersed in different areas of the search space, and thus, the distribution of the

population is scattered. In the proposed ECSA, two steps including adaptation of the inertia weight and controlling the acceleration coefficients were carried out for the velocity model as shown below:

Adaptation of the Inertia Weight

The inertia weight ρ in CSA is used to balance global and local search potentials. For optimal performance, the value of ρ is expected to be large in the case of exploration and small in the case of exploitation. However, it is not necessarily true to reduce ρ in CSA simply with time. Thus, an iterative factor f was proposed to be used in the inertia weight ρ to take part some properties with ρ. In this, the factor f is also relatively large during the exploration case and becomes relatively small in the convergence condition. Thus, it would be useful to enable ρ to put up with the iterative state using a sigmoid mapping $\rho(f)$: $\mathfrak{R}^+ \rightarrow \mathfrak{R}^-$. Here, the inertia weight ρ shown in Eq. 12 was proposed to be utilized in the velocity model of ECSA.

$$\rho(f) = \frac{1}{1.5e^{-2.6*f}} \in [0.4, 0.9] \qquad \forall f \in [0, 1] \tag{12}$$

In this work, ρ is initialized to 0.9. Since ρ is not necessarily monotonic over time, but monotonic with the iterative factor f, ρ will, thus, adapt to the search environment illustrated by f. In the case of jumping out or exploration case, large f and ρ will benefit the global search, as noted earlier. On the contrary, when f is small, an exploitation case or convergence case is identified, and thus, ρ goes down to benefit the local search. In view of this, the movements of the capuchins in the proposed ECSA model are implemented by a new integrated velocity-updating model with adaptation of the inertia weight throughout the iterative process. To be more specific, this new inertia weight was proposed to help update the capuchin's capuchins to move adaptively toward food.

Control of the Acceleration Coefficients

In addition to the inertia weight, the acceleration coefficients a_1 and a_2 are also important parameters in CSA that control the overall velocity of the capuchins. Accordingly, an adaptive control can be devised for these coefficients on the basis of the following idea. Parameter a_1 represents the "self-cognition" that attracts the capuchins to their own historical best positions, helping to explore local niches and to preserve the diversity of capuchins. This parameter reveals how much confidence a capuchin has in itself. Parameter a_2 represents the "social effect" that drives the capuchins to converge towards the current globally best area, which aids in rapid convergence. This parameter divulges how much trust a capuchin has in its neighbors. Braik et al. [27] stated in their original work of CSA that the implemented experiments showed that both the "cognitive-only" model and the "social-only" model are essential for the success of CSA, for which a constant value of 1.50 was used for each of the acceleration coefficients. However, it is anticipated that using ad hoc values of a_1 and a_2 instead of a constant value of 1.50 for different problems could result in better performance. To do so, the exponential models introduced in [47] were used to define the cognitive and social models in order to originate an enhanced variant of CSA, referred to as exponential CSA (ECSA). The exponential growth function shown in Eq. 13 was proposed to represent the cognitive model of the capuchins in ECSA.

$$a_1(t;\beta_0, \beta_1) = \beta_0(1 - e^{-\beta_1 t}) \tag{13}$$

where $t = \frac{k}{k^2}$, the parameter β_0 denotes the initial estimate of the cognitive parameter, and the parameter β_1 represents the final estimate of the social parameter approximately achievable by the capuchins at the end of the ECSA's iterative process.

It is important to note that the exponential function a_2 is derived from the exponential function a_1 as settled in Eq. 14.

$$a_2(t;\beta_0, \beta_1) = \frac{\partial a_1(t;\beta_0, \beta_1)}{\partial t} \tag{14}$$

According to Eqs. 13 and 14, the exponential function of the social model of the capuchins in ECSA is established in Eq. 15.

$$a_2(t;\beta_0, \beta_1) = \beta_0\beta_1 e^{-\beta_1 t} \tag{15}$$

To estimate the parameters β_0 and β_1 for the functions of cognitive and social parameters, there are several conventional and intelligent methods mentioned in the literature [27]. One of the common traditional estimation methods is the least square estimation method [48]. This method has many issues with estimation accuracy and needs a large number of measurements to be able to give a good estimation of parameters. Other methods include the use of metaheuristics in estimating the parameters [27]. These methods may demand senior computational efforts.

The parameters β_0 and β_1 of the exponential models used for a_1 and a_2 were selected through the use of experimental design by examining the proposed ECSA on feature selection problems. For all of the feature selection problems solved in this work, β_0 and β_1 are equal to 2.0 and 1.0, respectively. These values presented a high level of efficiency in solving feature selection problems as presented in the results section. However, optimal values are often obtained only empirically, perhaps, not the "best" values. Thereby, these parameters can be adapted to other problems as it is demanded.

The values of a_1 and a_2 are updated exponentially at each iteration loop of ECSA. In Fig. 1, the curve of the

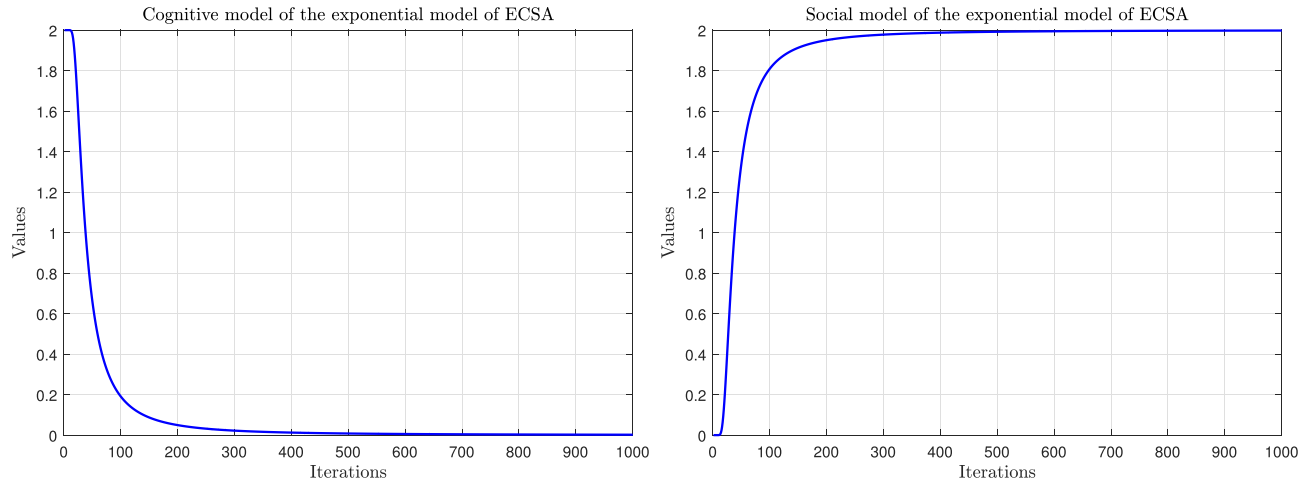

Fig. 1 Proposed exponential functions for a_1 and a_2 in the proposed ECSA, Left image a_1, Right image a_2

cognitive parameter of ECSA shows that the tendency of this parameter diminishes in an exponential manner. This has an impact on the conduct of the capuchins in ECSA which can be shifted towards more exploration and exploitation. Due to that, the capuchins can finish their foraging by locating the food at the end of their wanderings. This may also avoid local optimal solutions.

As can be observed in Fig. 1, the ECSA is proposed with time-varying acceleration coefficients, where a larger a_1 and a smaller a_2 were initially set and gradually reversed during the search process. As per this manner, it is expected that ECSA could divulge better overall performance than the basic CSA. This may be due to the time-varying of a_1 and a_2 that can balance the global and local search capabilities, which means that the adaptation of a_1 and a_2 can be encouraging in improving the performance of the basic CSA. As the iterative process of ECSA continues, the capuchins would clump together and converge into a locally or globally optimal region. Thus, the population distribution information would be various from that in the premature stage. In other words, the curve in Fig. 1 that represents the cognitive parameter expands exponentially until the capuchins realize and find the position of food sources. In this way, exploration and exploitation stages of the basic CSA are ameliorated, and the capuchins can eventually find food and not lose other comrade capuchins in the group while foraging.

Bounds of the Acceleration Coefficients

As discussed earlier, the aforementioned adjustments for inertia weight and acceleration coefficients should not be too troublesome. Thus, the maximum increase or decrease between two iterations is bounded by

$$\left|a_i(t+1) - a_i(t)\right| \leq \delta, \qquad i = 1, 2 \tag{16}$$

where δ is called the "acceleration rate."

Experiments revealed that a uniformly created random value for δ in the range of [0.05, 0.1] performed better in most of the feature selection problems under study. Note that 0.5 was used for δ, where it is recommended to make "slight" changes.

Power Model-Based CSA (PCSA)

The inertia weight ρ used in PCSA is the same as that used in ECSA which is given in Eq. 12. The difference between PCSA and ECSA models is the functions used to control the acceleration coefficients of the velocity of these models. Anyway, the cognitive and social coefficients of PCSA are adapted using the power model described in [49]. This is why this algorithm of CSA is referred to as power CSA (PCSA). This model is based upon the heterogeneous Poisson process model. The mathematical function formulated in Eq. 17 was utilized to carry out a_1 in the velocity model in PCSA.

$$a_1(t;\beta_0, \beta_1) = \beta_0 t^{\beta_1} \tag{17}$$

where $t = \frac{k}{K}$. It is important to note that the power model of a_2 is derived from the power model of a_1 as defined in Eq. 14. According to Eqs. 17 and 14, the power function of the social model of the capuchins in PCSA can be defined as shown in Eq. 18.

$$a_2(t;\beta_0, \beta_1) = \beta_0 \beta_1 t^{\beta_1 - 1} \tag{18}$$

Equations 17 and 18 were employed to update a_1 and a_2 during the iterative process of PCSA. It is important to know that Eq. 18 was used to find a_2 in this model. A

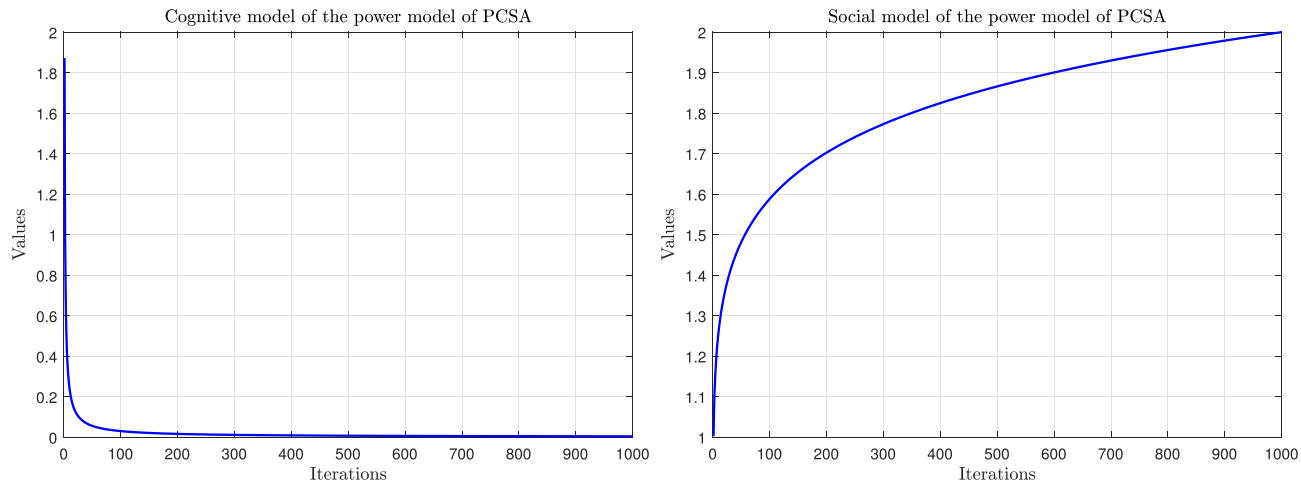

Fig. 2 Proposed power functions for a_1 and a_2 in the proposed PCSA, Left image a_1, Right image a_2

pool of values in the range from 0 to 5 were applied to each of β_0 and β_1. For all of the feature selection problems tackled in this work, β_0 and β_1 are equal to 2.0 and 0.1, respectively. These parameters were selected by experimental testing of a large subset of test datasets of varied complexities, where this value reported the best accuracy of the proposed PCSA. However, these parameters can be adapted for other problems as it is needed. The values of a_1 and a_2 were updated in PCSA, in non-linear form, as displayed in Fig. 2.

It is evident from Fig. 2 that the proclivities of a_1 and a_2 of PCSA are decreasing and increasing non-linearly, respectively. This has an impact on the search conduct of PCSA which can be moved towards more exploration or exploitation in a faster base when compared to CSA that utilizes constant values for the parameters a_1 and a_2.

It is clear from Eqs. 17 and 18 that when t is small, a_1 has an extreme value and swiftly lessens to a minimum value. On the other hand, the value of a_2 is small when t is small, and it progressively augments towards its maximum value. In this context, the capuchins in PCSA can find a food source at the end of their foraging activity. In detail, a_1 starts from a large value and declines little by little to a small value to denote that the capuchins find a source of food. Conversely, a_2 starts with a small value and gradually expands to a maximum value to indicate that the capuchins ultimately became aware of the location of the food source. This scenario of using power functions for a_1 and a_2 can ameliorate exploration and exploitation as presented in the evaluation results. The maximum increase or decrease between two successive iterations of the acceleration coefficients in PCSA is bounded using Eq. 16.

Delayed S-Shaped Model-Based CSA (SCSA)

The inertia weight ρ used in SCSA is the same as that used in ECSA and PCSA which is defined in Eq. 12. The difference between the former proposed algorithms of CSA and SCSA is the models used to control the acceleration coefficients of the velocity model of SCSA. In any case, the acceleration coefficients of SCSA are adapted using the S-shaped model introduced in [50]. This is why this algorithm of CSA is called S-shaped CSA (SCSA). The S-shaped mathematical growth model used to define the parameter a_1 is presented as given in Eq. 19.

$$a_1(t;\beta_0,\beta_1) = \beta_0\left(1-\left(1+\beta_1 t\right)e^{-\beta_1 t}\right) \tag{19}$$

where $t = \frac{K}{k}$.

Equation 14 was used to derive a_2 from a_1 in terms of iterations as defined in 19, where a_2 was got as presented in Eq. 20.

$$a_2(t;\beta_0,\beta_1) = \beta_0\beta_1^2 te^{-\beta_1 t} \tag{20}$$

For all of the feature selection problems addressed in this work, $\beta_0 and \beta_1$ are equal to 2.0 and 1.0, respectively. These parameters were determined by empirical testing of a large number of feature selection problems, where these parameters were changed several times until a trustworthy solution was acquired by the proposed SCSA. The growth functions representing the parameters a_1 and a_2 are updated in a non-linear shape as displayed in Fig. 3.

In Fig. 3, the curve of a_1 in SCSA shows that the trend of this parameter is gradually decreasing in a non-linear manner. Briefly, the parameters β_0 and β_1 in Eqs. 13, 15, 17, 18, 19, and 20 can be fine-tuned for other problems as

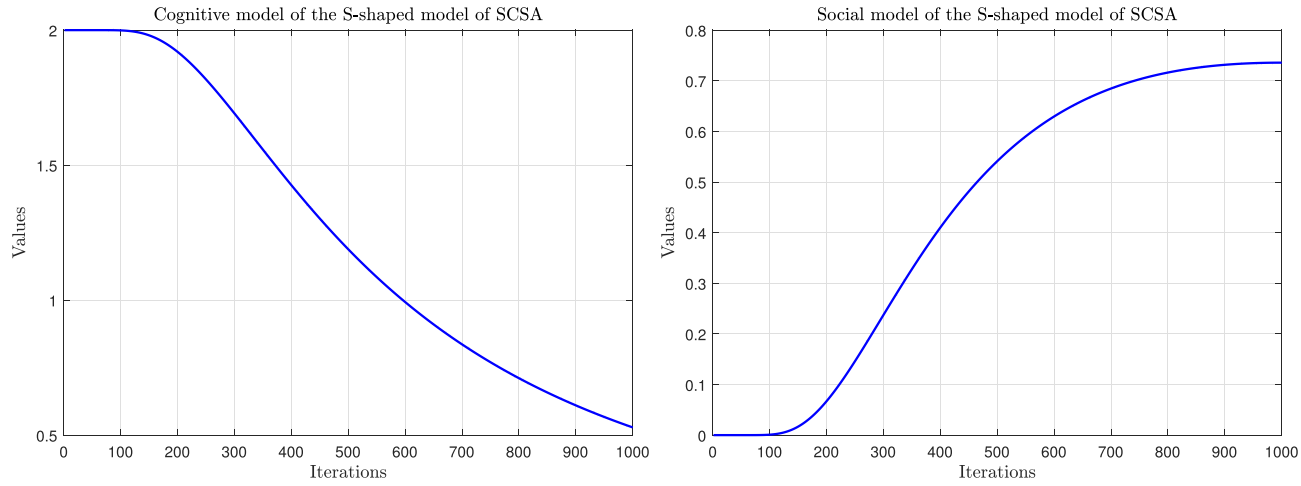

Fig. 3 Proposed S-shaped functions for a_1 and a_2 in the proposed SCSA, Left image a_1, Right image a_2

demanded. The three new algorithms of the basic CSA were proposed to lay out a suitable setting for a_1 and a_2 in order to enhance the exploration and exploitation features of CSA. These proposed algorithms are anticipated to fulfill efficient convergence and ameliorate the performance of the parent CSA in solving feature selection problems under study. Besides, the proposed algorithms of CSA could deliver outstanding potential to reliably evade stagnation in local optima regions and help them to find the global optima.

Briefly, in the original mechanism of updating capuchins' velocity as presented in Eq. 3, the values of a_1 and a_2 are constants during the iterative process of CSA. This points out that exploration and exploitation processes in CSA are based on a mathematical model that relies on stationary parameters. This has a big impact on global and local search abilities that can only give sensible exploration and exploitation without strict structure. In the proposed ECSA, PCSA, and SCSA, the parameters a_1 and a_2 were applied as interactive operators to foster exploration and exploitation capabilities of these proposed versions of CSA. With a set of different values for a_1 and a_2, these proposed algorithms can switch between global and local searches to promote the convergence performance of CSA to realize optimality.

Supportive Positioning Update Process

For further exploration and exploitation, the positions of leaders and followers in ECSA, PCSA, and SCSA are then updated as per the mechanism proposed in Eq. 21.

$$x_{k+1}^{i} = \begin{cases} x_k^i + v_k^i & rand \geq 0.5 \\ F_j + \lambda\left[lb_j + r\left(ub_j - lb_j\right)\right] & rand < 0.5 \end{cases} \tag{21}$$

where x_{k+1}^{i} and x_k^i denote the next and present positions of the ith capuchin at the next and current iterations, respectively, v_k^i stands for the present velocity of the ith capuchin at iteration k, r and $rand$ are uniformly random values generated in the interval from 0 to 1, and λ is defined as a function of iterations as drafted in Eq. 22.

$$\tau = \mu_0 e^{-(\mu_1 k/K)^{\mu_2}} \tag{22}$$

the parameters μ_0, μ_1, and μ_1 are constant values used to automatically update the parameter λ at each iteration. These parameters are useful for strengthening exploration and exploitation conducts. For all of the test problems subsequently addressed in this work, μ_0, μ_1, and μ_2 are set to 2, 4, and 2, respectively. These constants were captured by pilot testing of a bunch of test problems. However, they can be refined to suit other problems that those problems require.

The parameter λ is defined as a function of iterations k to control the random movement of capuchins iteratively and thus decreases with the number of iterations. Specifically, this parameter was proposed for dynamic system optimization to secure convergence by diminishing the search speed as well as to enhance exploration and exploitation of the proposed algorithms. This parameter can enable capuchins to scout more search space and exploit each region while looking for food or other capuchins' food. This is to arrive at an efficient convergence process, which can further improve the performance of ECSA, PCSA, and SCSA in solving feature selection problems. In this light, it is anticipated that the combination of the parameter λ with ECSA, PCSA, and SCSA will intensify the exploration capacity of these versions and bring them closer to the optimal solution.

The first case of Eq. 21 (i.e., when $rand \geq 0.5$) was suggested to allow capuchins to advance toward food sources. The second case of Eq. 21 (i.e., when $rand < 0.5$) was suggested to empower capuchins to scout several random positions in the search domain to improve local and

global search capabilities and to strike a sufficient balance between exploration and exploitation. This gives capuchins in ECSA, PCSA, and SCSA a great deal of ability to explore every potential position in the search space. In sum, Eq. 21 was combined with the mathematical models of ECSA, PCSA, and SCSA during their search iterations to address a number of FS problems from various domains, where the optimum feature subset characterizing the dataset is chosen. This is performed to get better classification efficiency by these versions than can be got with the basic CSA. In addition, the length of the feature subset is anticipated to be reduced with these versions over that realizable by the basic binary CSA.

To summarize, the proposed exponential, power, and S-shaped models for the cognitive and social models of the proposed ECSA, PCSA, and SCSA were utilized in these algorithms to improve the mobility of capuchins. In this work, the proposed binary ECSA, PCSA, and SCSA were applied to identify the most significant features from medium, small, and high-dimensional datasets in binary search spaces, on top of reducing the redundant and irrelevant features.

Complexity Analysis of ECSA, PCSA, and SCSA

Time complexity of an optimization algorithm can be drafted using a function that relates the algorithm's running time to the size of the optimization problem. Given this, Big-O notation can be used. The time complexity analysis of these optimization methods basically relies on the following steps: problem definition, initialization procedure, population update, fitness evaluation, and selection method. The computational time of the fitness assessment is highly dependent on the particular optimization problems. In doing so, the general computational complexity of each of ECSA, PCSA, and SCSA is the same and can be computed as follows:

$$\begin{aligned}\mathcal{O}(ECSA) =& \mathcal{O}(problem\ def.) + \mathcal{O}(init.)\\ &+ \mathcal{O}(K(population\ update))\\ &+ \mathcal{O}(K(fitness\ eval.))\\ &+ \mathcal{O}(K(selection\ procedure))\end{aligned} \tag{23}$$

As Eq. 23 implies, the time complexity of ECSA, PCSA, and SCSA mainly depends on the total number of iterations (K), the population size (n), the dimension of the problem under study (d), and the cost of the fitness function (c). Also, the S-shaped transfer function used to get binary versions of these proposed algorithms is intended to update the solutions. In specific form, the general computational complexity of ECSA, PCSA, and SCSA can be formulated in the worst case as follows:

$$\begin{aligned}\mathcal{O}(ECSA) =& \mathcal{O}(1) + \mathcal{O}(nd) + \mathcal{O}(VKnd)\\ &+ \mathcal{O}(VKnc) + \mathcal{O}\left(VKn^2d\right)\end{aligned} \tag{24}$$

where V stands for the number of assessment experiments.

The number of iterations (K) is often greater than the population size (n), the problem dimension (d), and the cost of the fitness function (c). In this regard, the main parameters K and n are essential in evaluating the complexity issue of optimization algorithms. Also, as $nd \ll VKnd$ and $nd \ll VKcn$, so the components 1 and nd can be ruled out from the time complexity given in Eq. 24. Consequently, the general time complexity of ECSA, PCSA, and SCSA can be viewed as follows:

$$\mathcal{O}(ECSA) \cong (VKnd + VKnc + VKn^2d) \tag{25}$$

Proposed Algorithms for Feature Selection

The proposed ECSA, PCSA, and SCSA algorithms aspire to extend the exploration and exploitation characteristics of CSA beyond enhancing CSA to deal with complex search spaces for challenging feature selection problems. The proposed binary FS algorithms aim to find the optimal subset of attributes for classification tasks by locating the most relevant attributes and abolishing the unimportant attributes. In this work, a wrapper-based FS approach was intended using four different binary algorithms to address a pool of benchmark feature selection datasets that vary in complexity, number of attributes, and characteristics. However, the CSA was originally introduced to handle continuous optimization problems, whereby each search agent in CSA updates its position based on its current position, the position of the best individual found so far, and the position of all other remaining search agents [27]. Thus, the proposed algorithms of CSA can only deal with continuous search spaces. Anyway, as these algorithms are tailored to solve FS problems, where the search spaces of such problems can be delineated by binary values due to their familiar nature. Hence, there is a need to use binary versions of the algorithm, CSA, ECSA, PCSA, and SCSA, to evolve a search strategy for FS problems of binary search spaces. In light of this, some operators of these methods demand to be altered to create binary versions of them, which can provide output in binary forms as either "*0" or "*1." This allows the search agents of these algorithms (i.e., capuchins) to have binary solutions while solving FS problems. As per this, the capuchins can change their position in the search space during the iterative procedural loops, where the movements of the capuchins are associated with values of "*1" and "*0." Considering this, an S-shape transfer function (TF) was used to transform the continuous CSA, ECSA, PCSA, and SCSA into binary

ones, referred to as BCSA, BECSA, BPCSA, and BSCSA, as described in the "S-Shape Transfer Function" section, with the objective function of these algorithms in the "Fitness Function" section.

S-Shape Transfer Function

As per the assertions conducted in [51], one of the most practical ways of transforming an optimization method that deals with continuous problems to a method that can address binary problems is to create a binary version of the purposed optimization method. To do this without modifying the basis of the algorithm, a transfer function is the most widely norm used in this area. This strategy aims to establish the probability that an element in the feature subset, x^i, will be constrained in a binary form that can be either "*0" or "*1." An element value of "*0" presents that the corresponding feature is not chosen, while a value of "*1" points out that the feature was chosen. In this study, an S-shape TF was utilized to conduct the transformation task. The S-shape TF that was applied to transform the position of the capuchins (i.e., search agents) in the proposed algorithms of CSA from continuous to binary is presented in Eq. 26:

$$S\left(x_{m,j}^{k}\right) = \frac{1}{1 + exp^{-x_{m,j}^{k}}} \tag{26}$$

where S refers to the transfer vector, $S\left(x_{m,j}^{k}\right)$ which represents the probability value of the S-shape value, $x_{m,j}^{k}$ stands for the jth element in the solution x (i.e., search agents) at the mth dimension, and k denotes the current iteration.

The values of the elements in the solution x are converted to either "*1" or "*0" using Eq. 27.

$$x_{m,j}^{k} = \begin{cases} 1 & rand < S\left(x_{m,j}^{k}\right), \\ 0 & rand \geq S\left(x_{m,j}^{k}\right) \end{cases} \tag{27}$$

where *rand* stands for a function that generates a uniform random number in the range [0, 1].

Fitness Function

A wrapper-based FS method needs a learning algorithm to be involved in evaluation of the selected subset of features. Here, the k-Nearest Neighbor (k-NN) classifier [52] was used to get a sense of the classification accuracy of the obtained solutions. In addition, each optimization problem must be solved using a fitness function, which is essential to any wrapper-based FS approach. While drafting a FS method, two key considerations must be made: How to express the solution and evaluate it. A binary vector with a length equal to the number of features in the dataset is used in this study to create a feature subset. Two discrepancy objectives are imposed on the feature selection problem as a multi-objective optimization problem as follows: (1) lessen the amount of features that are chosen, and 2) problem the k-NN classifier's classification accuracy. It is broadly known that as the number of features selected in a solution diminishes, the quality of the solution improves, by which the solution with the lowest number of features associated with the largest classification value is the best solution which is required to be achieved. These two opposite goals should be deemed into one objective function in the proposed FS algorithms. By this, these goals were implemented into one objective function for each solution in the iterative process of the proposed BCSA, BECSA, BPCSA, and BSCSA that was computed using the k-NN classifier as presented below:

$$fitness = \alpha\zeta_k + \beta\frac{|R|}{|N|} \tag{28}$$

where ζ_k is the rate of classification error got by k-NN, $|R|$ and $|N|$ are the number of selected and original features, respectively, and α and β are the weights of the classification quality and selection ratio of the chosen features, respectively, where they are two counteractive parameters in the interval [0, 1], in which β is the complement of α.

The fitness function presented in Eq. 28 that considers the trade-off between the selected features in each solution vector (i.e., minimization) and the classification accuracy rate of the learning classifier (i.e., maximization) is used to assess the selected subsets of the proposed BECSA, BPCSA, and BSCSA as FS methods.

The proposed BECSA, BPCSA, and BSCSA as FS methods are evaluated using the fitness function shown in Eq. 28, which takes into account the trade-off between the classification accuracy rate of the learning classifier (i.e., maximization) and the selected features in each solution vector (i.e., minimization). The schematic diagram of the proposed work is presented in Fig. 4.

Experimental Results and Discussions

The effectiveness and robustness of the proposed algorithms of CSA and basic CSA in solving FS problems are studied and analyzed in this section. First, a brief description of the datasets used for the evaluation tasks is found in the "Dataset Description" section. Then, the parameter settings of the proposed algorithms and the characteristics of the system used to run the algorithms are provided in the "Parameter Settings" section, while the evaluation metrics are formulated in the "Performance Measures" section. The results of evaluation and analysis of the proposed algorithms are

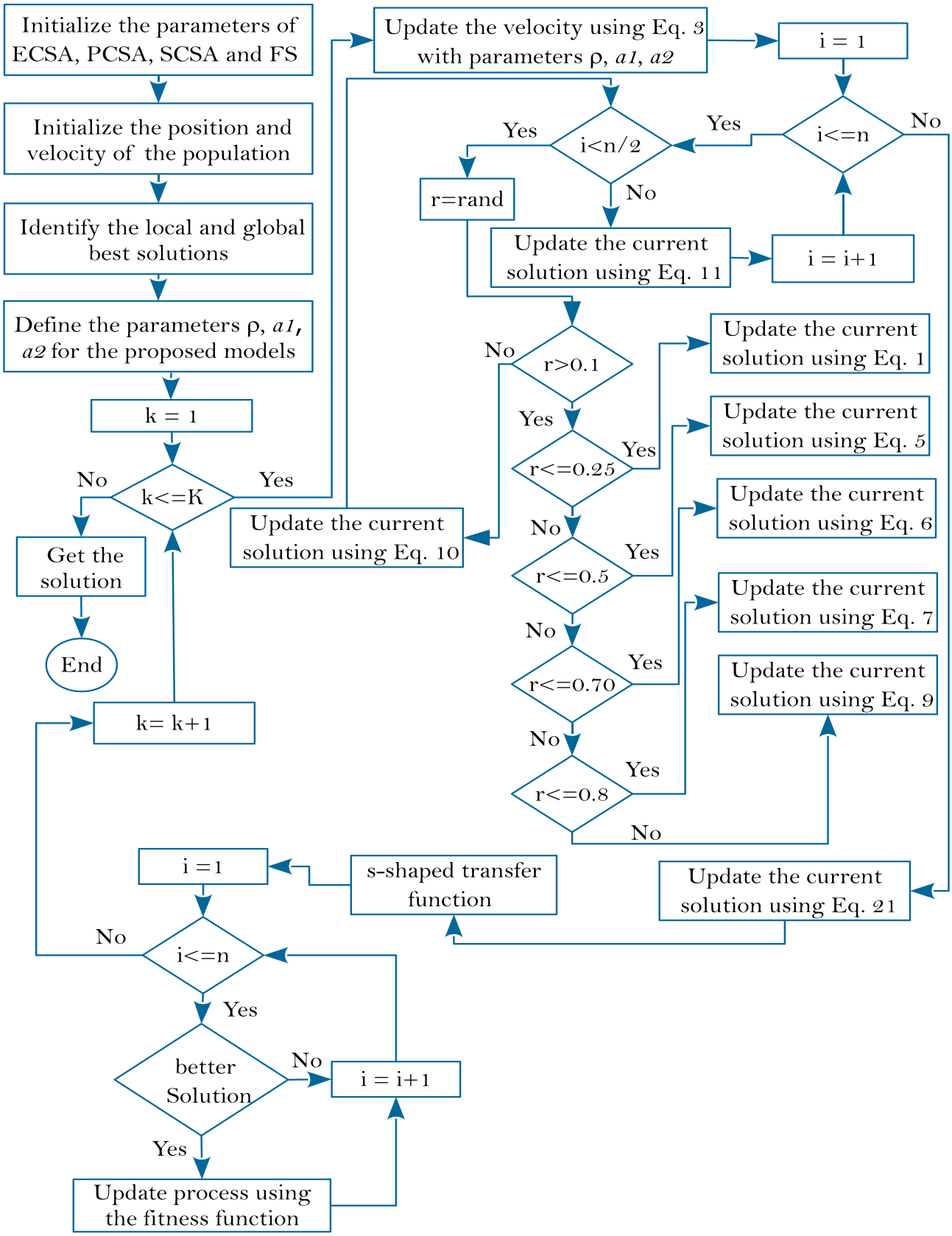

Fig. 4 Schematic diagram of the proposed algorithms of CSA for feature selections

Table 1 A brief description of the first 23 datasets

Dataset	# Features	# Samples	# Classes	Source
Diagnostic	30	569	2	UCI
Breast	9	699	2	UCI
Prognostic	33	194	2	UCI
Coimbra	9	115	2	UCI
BreastEW	30	596	2	UCI
Retinopathy	19	1151	2	UCI
Dermatology	34	366	6	UCI
ILPD	10	583	2	UCI
Lymphography	18	148	4	UCI
Parkinsons	22	194	2	UCI
ParkinsonC	753	755	2	UCI
SPECT	22	267	2	KEEL
Cleveland	13	297	5	KEEL
HeartEW	13	270	2	KEEL
Hepatitis	18	79	2	KEEL
Saheart	9	461	2	KEEL
Spectfheart	43	266	2	KEEL
Thyroid	21	7200	3	KEEL
Heart	13	302	5	Kaggle
PimaDiabetes	9	768	2	Kaggle
Leukemia	7129	72	2	Medical website
Colon	2000	62	2	Medical website
ProstateGE	5966	102	2	Medical website

summarized in the "Evaluation of the Proposed Methods" section. Finally, the outcomes of the fundamental CSA and the best developed algorithm compared to other algorithms are presented in the "Performance Comparison with Other Methods" section.

Dataset Description

The performance of the proposed algorithms of CSA is evaluated using 24 benchmark datasets of different number of features and varying complexity. These datasets were extracted from patients' medical diagnoses. It should be noted that these datasets were collected from the UCI repository, KEEL repository, Kaggle, and another well-known medical website.[1] Table 1 presents a brief description of the first 23 datasets used in the current study.

From Table 1, it can be clearly seen that only the main characteristics of each dataset are presented as follows: number of features, number of samples, number of classes, and the source of each dataset. In addition, the number of samples ranged from 62 and 1151, the number of features ranged from 9 to 5966, and the number of classes ranged from 2 to 6.

[1] https://jundongl.github.io/scikit-feature/datasets.html.

Table 2 A characterization of the COVID-19 dataset utilized

No.	Features	Description
1	ID	Patients' ID
2	Location	The locations to which patients belong
3	Country	The country to which the patients belong
4	Gender	Patients' sex
5	Age	Patients' ages
6	Sym_on	Patients' date of symptoms
7	Hosp_vis	The date of the patient's visit to the hospital
8	Vis_wuhan	Whether the patients have visited Wuhan
9	From_wuhan	Whether the patients are from Wuhan or China
10	Symptom1	Patients' symptoms
11	Symptom2	Patients' symptoms
12	Symptom3	Patients' symptoms
13	Symptom4	Patients' symptoms
14	Symptom5	Patients' symptoms
15	Symptom6	Patients' symptoms

Table 2 shows a description of the real-world coronavirus disease (i.e., SARS-CoV-2 or broadly called COVID-19) dataset, where this dataset was gathered from [16], which comprises of 15 features.

In order to prevent overfitting in solving the feature selection problems under study based on the proposed feature selection models, the data is randomly split into training and testing datasets. This is also to ensure the stability of the outcomes. In this, the total number of instances in the datasets shown in Tables 1 and 2 were split into two partitions, where 80% of the instances were used for training, and the remaining 20% of the instances in the dataset were used for testing. This is also recommended in many related works in the literature [5, 37]. For parameter tuning, the number of iterations of the training model was set to 100 based on early stopping criteria, as aforementioned, in order to help prevent overfitting. These two techniques make it possible to ameliorate the number of training instances by altering the existing ones. Parameter optimization of the proposed methods can be found in the "Sensitivity Analysis" section. There are many other methods to reduce model overfitting such as 10-fold cross validation [53], early stopping criteria [54], and image augmentation by generating new training samples from the existing training dataset [55].

Parameter Settings

The parameter settings of the proposed algorithms are recorded in Table 3. It should be pointed out that each algorithm is repeated 30 independent times to give an idea of its stability. Thereafter, the results of the proposed algorithms are collected and summarized in terms of average and standard derivation values. Furthermore,

Table 3 Parameter settings of the developed FS methods

Parameter	Setting
Number of iterations	100
Population size	30
P_{bf}, P_{ef}, Pr	0.75, 0.9, 0.1
For BCSA: a_1, a_2	1.5, 1.5
For BECSA, BPCSA, BSCSA: a_1, a_2	Adaptive
Classifier	k-NN with $k = 5$
Number of runs	30
α and β of the fitness function	0.99 and 0.01

all the experiments were conducted and carried out using a personal computer with Intel(R) Core(TM) i7-7700HQ with 2.80GHz CPU and 16.0 GB RAM. The proposed algorithms and other comparative algorithms were implemented using MATLAB 2022A.

Performance Measures

This study verifies the effectiveness and accuracy of the proposed algorithms against other comparative ones. Several performance measures are utilized for these purposes, which can be defined as follows [4, 5]:

1. Classification accuracy: This evaluation method introduces the correct classification ratio of the studied cases in the test dataset as shown in Eq. 29.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{29}$$

   where the true positive (TP) is the correctly defined classes; the true negative (TN) is the properly rejected classes; the false positive (FP) is the incorrectly identified classes; and the false negative (FN) is the incorrectly rejected classes.
2. Sensitivity: This metric is utilized to calculate the proportion of all true positive cases in the test dataset as given by Eq. 30.

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{30}$$

3. Specificity: This metric measure was introduced to compute the proportion of all true negative cases in the test dataset as given by Eq. 31.

$$\text{Specificity} = \frac{TN}{FP + TN} \tag{31}$$

4. Fitness value: This metric measure is used to determine the quality of the desired solution as given by the mathematical formula defined in Eq. 28.
5. Number of selected features: This criterion is used to present the number of features in the desired solution.

For comparison purposes, the average results and standard deviations of these metric measures were obtained, where each algorithm was executed 30 independent times as aforementioned.

Evaluation of the Proposed Methods

The evaluation of the developed algorithms of CSA together with the original CSA for feature selection is studied in this section. In this, the performance levels of the proposed algorithms (binary exponential CSA (BECSA), binary power CSA (BPCSA), and binary S-shaped CSA (BSCSA)) are compared with those of the binary version of the parent CSA (BCSA). This is necessary to specify the version of CSA reveals the best level of performance. Tables 4, 5, and 6 display the results of the different variants of CSA based upon the following evaluation measures: (i) classification accuracy rates, (2) fitness values, (3) sensitivity, (4) specificity, and (5) number of selected features. The average (AV) and standard deviation (SD) values of the results of each algorithm, which were performed 30 independent runs, for each dataset and each evaluation measure are recorded in Tables 4, 5, and 6. On top of that, the average ranking of each proposed algorithm for each evaluation measure was obtained using Friedman's test and is provided in the last line of these tables. The best results are highlighted in bold to give them more weight over the other results.

Initially, the average results and standard deviations of the different variants of CSA in terms of classification accuracy and fitness values are summarized in Table 4. The higher average accuracy results mean better algorithm performance. It can be clearly seen that the performance of the four variants of CSA was similar by obtaining the same accuracy results in 8 datasets (i.e., Breast, Prognostic, Coimbra, Saheart, PimaDiabetes, Leukemia, Colon, and ProstateGE). In addition, the performance of BECSA, BPCSA, and BSCSA was better than the basic BCSA in the ILPD and COVID-19 datasets. Besides, BECSA performed better or comparable to other versions by obtaining the best accuracy results in 18 datasets. BSCSA obtained the best results in 14 datasets, while BCSA achieved the best accuracy results in 12 datasets. Surprisingly, the performance of BPCSA was better than or similar to other proposed versions in 11 datasets. On the other hand, the results of the standard deviation (SD) reflect the robustness of the algorithm, with the lowest SD results being the best. Reading the results in Table 4 once more, it can be shown that the proposed BECSA was more robust than the other variants by getting the minimum SD values in 17 out of the 24 datasets considered in this study. Finally, but not least, it can be observed that the proposed

Table 4 A comparison of the four binary variants of the CSA algorithm based on average classification accuracy and fitness values

		Accuracy				Fitness			
Dataset	Measure	BCSA	BECSA	BPCSA	BSCSA	BCSA	BECSA	BPCSA	BSCSA
Diagnostic	AV	0.9590	**0.9655**	0.9649	0.9649	0.0423	**0.0369**	0.0375	0.0377
	SD	0.0085	0.0027	0.0016	0.0016	0.0081	0.0027	0.0015	0.0016
Breast	AV	**0.9926**	**0.9926**	**0.9926**	**0.9926**	**0.0123**	**0.0123**	**0.0123**	**0.0123**
	SD	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Prognostic	AV	**0.8947**	**0.8947**	**0.8947**	**0.8947**	**0.1070**	0.1076	0.1079	0.1078
	SD	0.0000	0.0000	0.0000	0.0000	0.0012	0.0005	0.0005	0.0005
Coimbra	AV	**0.9130**	**0.9130**	**0.9130**	**0.9130**	0.0908	**0.0905**	**0.0905**	**0.0905**
	SD	0.0000	0.0000	0.0000	0.0000	0.0005	0.0000	0.0000	0.0000
BreastEW	AV	**0.9912**	**0.9912**	0.9906	**0.9912**	0.0132	**0.0128**	0.0135	0.0129
	SD	0.0000	0.0000	0.0022	0.0000	0.0011	0.0004	0.0019	0.0005
Retinopathy	AV	0.7123	**0.7152**	0.7146	0.7148	0.2889	**0.2859**	0.2861	0.2866
	SD	0.0046	0.0032	0.0039	0.0029	0.0046	0.0024	0.0033	0.0024
Dermatology	AV	0.9920	**0.9967**	0.9930	0.9958	0.0122	**0.0080**	0.0115	0.0091
	SD	0.0071	0.0061	0.0072	0.0066	0.0062	0.0055	0.0065	0.0058
ILPD	AV	0.7557	**0.7565**	**0.7565**	**0.7565**	0.2453	**0.2440**	**0.2440**	**0.2440**
	SD	0.0027	0.0000	0.0000	0.0000	0.0031	0.0000	0.0000	0.0000
Lymphography	AV	0.9448	0.9506	0.9448	**0.9529**	0.0595	0.0539	0.0594	**0.0519**
	SD	0.0172	0.0174	0.0172	0.0169	0.0162	0.0163	0.0162	0.0158
Parkinsons	AV	0.9778	0.9803	**0.9829**	**0.9829**	0.0227	0.0212	**0.0189**	**0.0189**
	SD	0.0089	0.0110	0.0123	0.0123	0.0084	0.0100	0.0114	0.0114
ParkinsonC	AV	0.8022	**0.8150**	0.8009	0.8035	0.1982	**0.1880**	0.2020	0.1994
	SD	0.0346	0.0197	0.0209	0.0229	0.0340	0.0195	0.0207	0.0226
SPECT	AV	**0.8805**	0.8786	0.8792	0.8792	**0.1225**	0.1245	0.1240	0.1237
	SD	0.0114	0.0095	0.0094	0.0117	0.0116	0.0092	0.0089	0.0113
Cleveland	AV	0.6017	0.6096	0.6085	**0.6102**	0.3980	0.3891	0.3905	**0.3884**
	SD	0.0116	0.0031	0.0052	0.0000	0.0127	0.0038	0.0060	0.0003
HeartEW	AV	0.8951	**0.9062**	0.9031	0.9012	0.1072	**0.0966**	0.0995	0.1012
	SD	0.0156	0.0047	0.0080	0.0101	0.0151	0.0043	0.0073	0.0095
Hepatitis	AV	0.9375	**0.9479**	0.9438	0.9458	0.0631	**0.0534**	0.0577	0.0556
	SD	0.0000	0.0237	0.0191	0.0216	0.0006	0.0226	0.0182	0.0208
Saheart	AV	**0.7065**	**0.7065**	**0.7065**	**0.7065**	**0.2939**	**0.2939**	**0.2939**	**0.2939**
	SD	0.0000	0.0000	0.0000	0.0000	0.0002	0.0000	0.0000	0.0000
Spectfheart	AV	**0.9170**	0.9120	0.9145	0.9132	**0.0867**	0.0913	0.0890	0.0903
	SD	0.0161	0.0114	0.0129	0.0117	0.0165	0.0112	0.0124	0.0116
Thyroid	AV	**0.9911**	0.9908	0.9903	0.9906	**0.0111**	0.0122	0.0133	0.0127
	SD	0.0013	0.0008	0.0011	0.0007	0.0011	0.0009	0.0011	0.0009
Heart	AV	0.8778	**0.8889**	0.8867	0.8878	0.1248	**0.1141**	0.1162	0.1152
	SD	0.0160	0.0160	0.0166	0.0163	0.0152	0.0151	0.0157	0.0154
PimaDiabetes	AV	**0.7451**	**0.7451**	**0.7451**	**0.7451**	**0.2574**	**0.2574**	**0.2574**	**0.2574**
	SD	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Leukemia	AV	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**0.0012**	0.0047	0.0048	0.0048
	SD	0.0000	0.0000	0.0000	0.0000	0.0002	0.0000	0.0000	0.0000
Colon	AV	**0.9167**	**0.9167**	**0.9167**	**0.9167**	**0.0835**	0.0870	0.0870	0.0870
	SD	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000
ProstateGE	AV	**0.9000**	**0.9000**	**0.9000**	**0.9000**	**0.1004**	0.1038	0.1038	0.1038
	SD	0.0000	0.0000	0.0000	0.0000	0.0002	0.0000	0.0000	0.0000
COVID-19	AV	0.9589	**0.9593**	**0.9593**	**0.9593**	0.0430	**0.0427**	0.0434	0.0431
	SD	0.0015	0.0000	0.0000	0.0000	0.0018	0.0007	0.0011	0.0008
Avg. Rankings		2.979	2.083	2.708	2.229	2.792	1.875	2.854	2.479

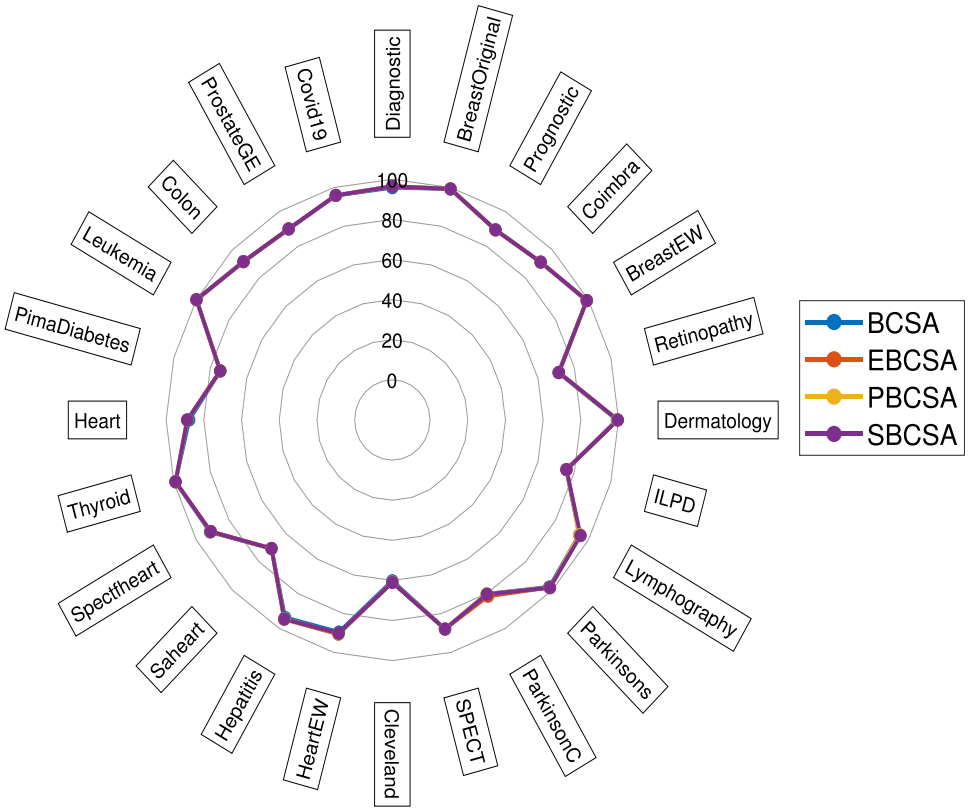

Fig. 5 Radar graph for BCSA, BECSA, BPCSA, and BSCSA based on the average classification accuracy results for all datasets

BECSA ranked first by having the minimum average ranking using Friedman's test, while BSCSA and BPCSA ranked second and third, respectively. BCSA was ranked last by having the largest average rankings. This proves the efficiency of the proposed changes to the main structure of the original CSA while favoring the proposed BECSA.

Figure 5 shows the average accuracy results of the four variants of CSA in all datasets using a radar chart. The shape with the larger area indicates that the algorithm obtains higher accuracy results and thus better performance. Obviously, the proposed BECSA, BPCSA, and BSCSA have almost the same scheme; however, there are slight differences between the BECSA, BPCSA, BSCSA, and BCSA algorithms.

The average fitness values and standard derivations of the four variants of CSA are presented in Table 4. A lower fitness value indicates better performance. From Table 4, it can be seen that the four variants got the same best results in the Breast, Saheart, and PimaDiabetes datasets. In addition, BECSA, BPCSA, and BSCSA obtained the same best fitness figures that are better than BCSA in the Coimbra and ILPD datasets. Besides, BECSA performed better or similar to the other variants by having the best results in 14 datasets. BCSA comes in second with the best results in 10 datasets. BSCSA and BPCSA came in third and fourth with the best results in 8 and 6 datasets, respectively. In the same vein, it can be observed that BECSA performed more robustly than other competitors by having the lowest SD values in 18 out of 24 datasets. Reading the results given in Table 4 one more time, one can see that the proposed BECSA was ranked first by getting the minimum average rankings using Friedman's test. As per this, BSCSA, BCSA, and PBSCA ranked second, third, and fourth, respectively.

Figure 6 shows the convergence behavior of the proposed BCSA, BECSA, BPCSA, and BSCSA according to the fitness results. In these plots, the x-axis reflects the number of iterations, while the y-axis represents the fitness values. The convergence behavior of the best solution obtained by each algorithm over 30 independent runs is illustrated in these plots. The preferred algorithm is the one that culminates the minimum fitness results with few iterations and at the same time does not get stuck in local optima. From Fig. 6, it can be noticed that there are slight differences between the behavior of the four proposed algorithms when navigating the search space of each problem. This is due to how well the algorithm balances exploitation and exploration capabilities. Upon examining the plots in Fig. 6 again, it can be viewed that the convergence conducts of the four proposed algorithms are nearly identical in the first 50 iterations in the Coimbra, Breast, ILPD, PimaDiabetes, and Saheart datasets. The convergence behavior of the four proposed companions becomes identical after half of the iterations in BreastEW, Retinopathy, Dermatology, Lymphography, Cleveland, HeartEW,

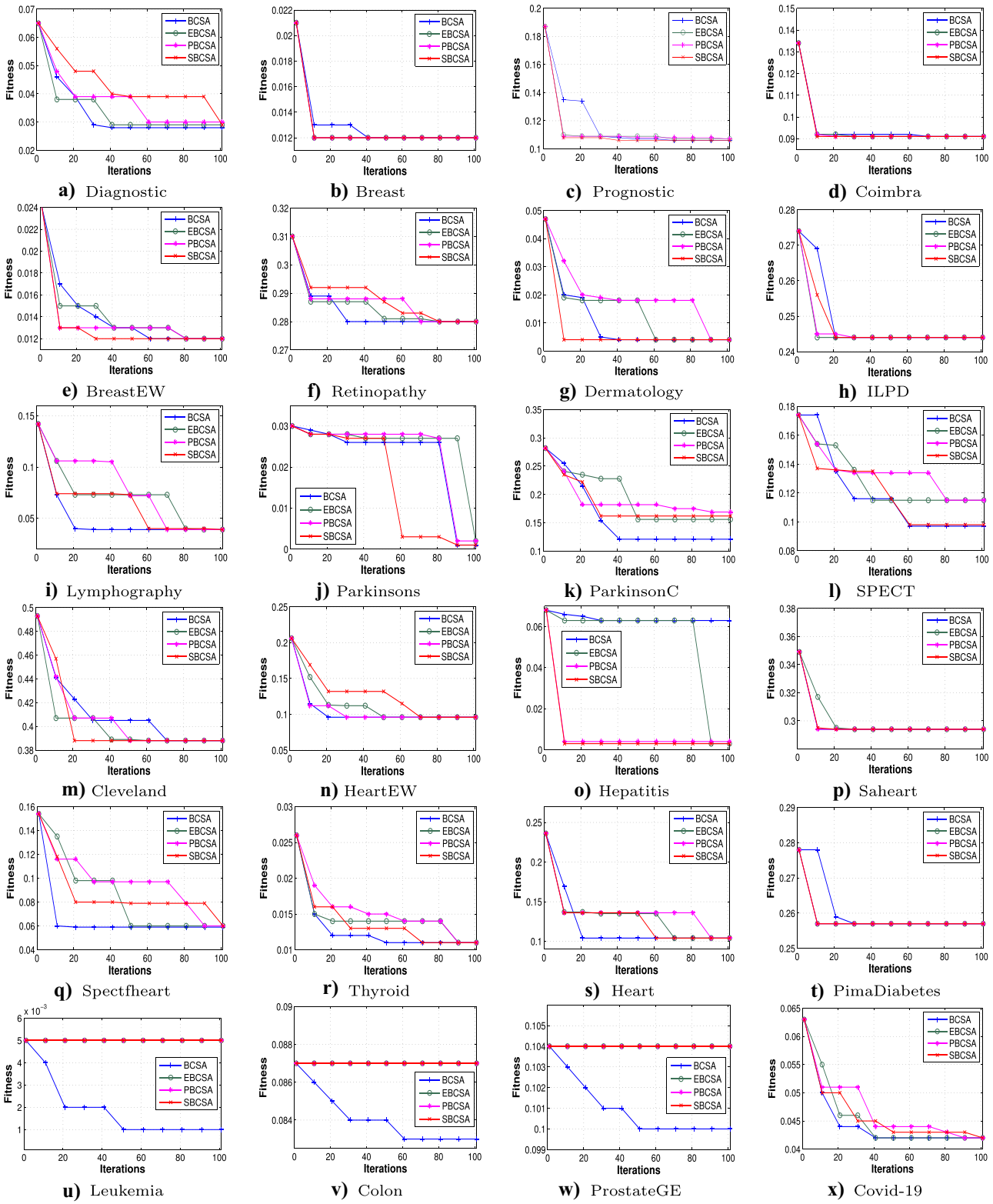

Fig. 6 Convergence characteristic curves of BCSA, BECSA, BPCSA, and BSCSA for all studied datasets

Table 5 A comparison of the four binary variants of the CSA algorithm in terms of sensitivity and specificity values

		Sensitivity				Specificity			
Dataset	Measure	BCSA	BECSA	BPCSA	BSCSA	BCSA	BECSA	BPCSA	BSCSA
Diagnostic	AV	0.8801	**0.8963**	0.8805	0.8749	0.9625	**0.9703**	0.9663	0.9649
	SD	0.0497	0.0413	0.0418	0.0461	0.0223	0.0260	0.0245	0.0194
Breast	AV	0.9877	**0.9919**	0.9903	0.9904	**0.9988**	0.9934	0.9957	0.9955
	SD	0.0035	0.0068	0.0060	0.0060	0.0045	0.0083	0.0072	0.0076
Prognostic	AV	**0.3214**	0.2714	0.3201	0.3125	0.8932	**0.9207**	0.9068	0.9000
	SD	0.1508	0.1269	0.1441	0.1836	0.0692	0.0489	0.0558	0.0592
Coimbra	AV	**0.7332**	0.7225	0.6691	0.6695	0.7923	**0.8202**	0.8199	0.8133
	SD	0.1177	0.1178	0.1103	0.1534	0.1066	0.0979	0.1190	0.1133
BreastEW	AV	**0.9579**	0.9454	0.9456	0.9511	0.9220	**0.9323**	0.9219	0.9281
	SD	0.0238	0.0234	0.0266	0.0256	0.0408	0.0457	0.0499	0.0453
Retinopathy	AV	0.6465	**0.6773**	0.6624	0.6506	0.6828	**0.6961**	0.6836	0.6955
	SD	0.0469	0.0500	0.0347	0.0484	0.0467	0.0495	0.0543	0.0455
Dermatology	AV	**0.9909**	**0.9909**	0.9908	0.9888	0.9073	**0.9399**	0.9263	0.9340
	SD	0.0188	0.0209	0.0217	0.0245	0.0482	0.0456	0.0296	0.0499
ILPD	AV	0.8400	**0.8447**	0.8356	0.8381	**0.3097**	0.2757	0.2755	0.2625
	SD	0.0508	0.0488	0.0406	0.0494	0.0991	0.0774	0.1006	0.0705
Lymphography	AV	**0.6541**	0.5263	0.6453	0.5900	0.8771	0.8663	0.8779	**0.8892**
	SD	0.4712	0.5011	0.4651	0.4906	0.0729	0.0711	0.0668	0.0609
Parkinsons	AV	0.9075	**0.9240**	0.9214	0.9154	0.5785	**0.5812**	0.5710	0.5242
	SD	0.0578	0.0482	0.0493	0.0638	0.1538	0.1931	0.1424	0.2274
ParkinsonC	AV	0.8823	**0.9007**	0.8870	0.8871	0.3262	**0.4038**	0.3642	0.3889
	SD	0.0282	0.0345	0.0304	0.0377	0.1219	0.0847	0.0676	0.1003
SPECT	AV	0.5953	0.5814	0.6100	**0.6274**	0.7408	0.7537	0.7512	**0.7946**
	SD	0.1186	0.1094	0.1246	0.1141	0.1183	0.0983	0.1390	0.1071
Cleveland	AV	**0.1779**	0.1735	0.1488	0.1652	0.6516	0.6642	0.6611	**0.6682**
	SD	0.0828	0.1428	0.0981	0.1168	0.0587	0.0715	0.0632	0.0652
HeartEW	AV	0.8788	**0.8807**	0.8470	0.8547	0.7139	**0.7481**	0.7215	0.7374
	SD	0.0652	0.0673	0.0651	0.0816	0.1234	0.0964	0.1099	0.1056
Hepatitis	AV	**0.3337**	0.2586	0.2282	0.2456	0.9583	0.9535	0.9531	**0.9640**
	SD	0.3451	0.3351	0.3433	0.3365	0.0529	0.1029	0.0567	0.0482
Saheart	AV	**0.2964**	0.2798	0.2725	0.2659	0.7623	0.7833	**0.7842**	0.7798
	SD	0.0797	0.0761	0.0879	0.0650	0.0516	0.0471	0.0591	0.0451
Spectfheart	AV	0.8723	0.8701	**0.8738**	0.8468	0.4145	**0.4380**	0.4354	0.4353
	SD	0.0499	0.0543	0.0662	0.0750	0.1639	0.1685	0.1699	0.1547
Thyroid	AV	0.6909	**0.7042**	0.6883	0.6966	**0.9876**	0.9872	0.9860	0.9873
	SD	0.0926	0.1003	0.0941	0.0726	0.0031	0.0036	0.0029	0.0037
Heart	AV	0.9280	0.9396	0.9342	**0.9462**	0.6416	0.6238	**0.6438**	0.6216
	SD	0.0478	0.0462	0.0441	0.0434	0.1307	0.1143	0.1251	0.0951
PimaDiabetes	AV	0.4973	0.4791	**0.5202**	0.4885	0.8008	**0.8203**	0.7859	0.8058
	SD	0.0630	0.0550	0.0530	0.0651	0.0318	0.0314	0.0413	0.0396
Leukemia	AV	0.7610	0.6932	**0.7980**	0.7534	0.9735	**0.9782**	0.9474	0.9720
	SD	0.1685	0.2449	0.1708	0.1970	0.0449	0.0452	0.0650	0.0476
Colon	AV	0.5633	0.5972	0.5340	**0.6252**	0.8640	0.8328	**0.8987**	0.8154
	SD	0.2570	0.2216	0.2931	0.2288	0.1201	0.1669	0.1364	0.1495
ProstateGE	AV	0.8216	0.8305	**0.8646**	0.8313	0.8198	**0.8487**	0.8373	0.8455
	SD	0.1309	0.1214	0.0884	0.1553	0.1264	0.1178	0.1197	0.0968
COVID-19	AV	0.5997	**0.6161**	0.5984	0.6010	**0.9794**	0.9788	0.9790	0.9777
	SD	0.0928	0.1144	0.1027	0.0937	0.0108	0.0109	0.0151	0.0123
Avg. Rankings		2.333	2.333	2.708	2.625	2.958	1.833	2.667	2.542

Table 6 A comparison of the four binary variants of the CSA algorithm in terms of the number of selected features

Dataset	Measure	BCSA	BECSA	BPCSA	BSCSA
Diagnostic	AV	**5.30**	8.47	8.37	9.03
	SD	3.42	1.74	1.54	1.16
Breast	AV	**5.00**	**5.00**	**5.00**	**5.00**
	SD	0.00	0.00	0.00	0.00
Prognostic	AV	**9.47**	11.67	12.67	12.37
	SD	4.01	1.65	1.63	1.65
Coimbra	AV	4.27	**4.00**	**4.00**	**4.00**
	SD	0.45	0.00	0.00	0.00
BreastEW	AV	13.43	**12.20**	12.60	12.43
	SD	3.35	1.30	1.65	1.57
Retinopathy	AV	7.80	7.50	**6.87**	8.03
	SD	2.44	2.22	2.05	1.92
Dermatology	AV	**14.50**	16.07	15.30	16.67
	SD	4.51	3.19	2.87	3.22
ILPD	AV	3.37	**3.00**	**3.00**	**3.00**
	SD	0.67	0.00	0.00	0.00
Lymphography	AV	8.73	8.93	**8.60**	9.40
	SD	1.86	1.96	1.75	2.04
Parkinsons	AV	**1.43**	3.90	4.33	4.33
	SD	1.04	2.23	1.94	2.04
ParkinsonC	AV	**177.47**	367.17	367.07	366.43
	SD	64.22	15.01	14.41	11.13
SPECT	AV	9.30	9.47	9.83	**9.10**
	SD	2.51	1.76	2.09	1.58
Cleveland	AV	4.73	3.33	3.70	**3.27**
	SD	2.30	0.96	1.34	0.45
HeartEW	AV	**4.30**	4.87	4.57	4.47
	SD	1.12	0.51	0.82	0.86
Hepatitis	AV	**2.30**	3.50	3.77	3.70
	SD	1.06	1.68	1.45	1.29
Saheart	AV	3.03	**3.00**	**3.00**	**3.00**
	SD	0.18	0.00	0.00	0.00
Spectfheart	AV	19.80	**18.27**	19.17	19.20
	SD	5.54	2.92	3.90	3.18
Thyroid	AV	**4.87**	6.47	7.73	7.17
	SD	0.57	1.11	1.34	1.49
Heart	AV	**4.90**	5.33	5.23	5.27
	SD	1.09	0.96	0.97	0.98
PimaDiabetes	AV	**4.00**	**4.00**	**4.00**	**4.00**
	SD	0.00	0.00	0.00	0.00
Leukemia	AV	**878.53**	3347.43	3365.70	3365.20
	SD	170.66	13.62	13.58	9.80
Colon	AV	**206.10**	902.07	906.43	907.50
	SD	27.16	6.50	8.33	7.51
ProstateGE	AV	**841.50**	2842.37	2843.30	2843.70
	SD	106.92	10.29	13.04	13.67
COVID-19	AV	**3.30**	3.43	4.40	3.93
	SD	0.79	0.94	1.59	1.17
Avg. Rankings		2.042	2.375	2.771	2.812

Thyroid, and Heart. The convergence behavior of the four algorithms is not identical in most iterations and becomes indistinguishable in the last few iterations as shown in the plots of Spectfheart and COVID-19 datasets. However, the convergence behavior of the proposed BECSA is superior to others in the Parkinsons, Prognostic, and Hepatitis datasets. Finally, the convergence conduct of BCSA is better than the other variants in the Leukemia, Colon, ProstateGE, ParkinsonC, and SPECT datasets. It should be distinguished that the performance of the proposed BECSA is better than BCSA in terms of average fitness values in these datasets as displayed in Table 4.

The average sensitivity and specificity results as well as the standard derivations of the four proposed binary variants of CSA are given in Table 5. Higher sensitivity and specificity indicate better performance. According to the sensitivity results, it can be clearly seen that BCSA got the best sensitivity results in 9 datasets, while BECSA acquired the best sensitivity results in 8 datasets. BPCSA achieved the best sensitivity results in 4 datasets, while BSCSA ranked last by having the best scores in 3 datasets. Therefore, both BCSA and ECSA were ranked first by having the same minimum average ranking using Friedman's test, while the remaining two variants came in the last two rankings. When reading the sensitivity results again, it can be noticed that the performance of BCSA, BECSA, and BPCSA is almost the same as per the SD results, while the performance of these three algorithms was better than that of the basic BSCSA.

The specificity outcomes of the developed binary versions of CSA are recorded in Table 5. Apparently, BECSA performed better than or similar to the other variants by having the best specificity scores in 13 out of 24 datasets. BCSA and BSCSA came in second place with each having the best specificity results in 4 datasets. BPCSA finally came out with the best results in 3 datasets. In the same vein, BECSA ranked first by obtaining the minimum average ranking using Friedman's test, while BSCSA, BPCSA, and BCSA ranked second, third, and fourth, respectively.

Finally, the average number of selected features and standard derivations of the four variants of CSA are presented in Table 6. It is worthy to note that the minimum average of the selected features points out a better performance score. From Table 6, it can be seen that BCSA obtained the lowest average of the number of selected features in 15 out of 24 datasets. At the same time, each algorithm of the remaining variants got the lowest average of the selected features in 7 datasets. Additionally, BCSA is placed first by achieving the minimum average ranking using Friedman's test, while BECSA is ranked second. BPCSA was placed third using Friedman's test, and finally, BSCSA ranked last.

Discussion of the Results

As evidenced by a second reading of the findings in Table 4, BECSA exclusively outperformed its competitors in respect of the accuracy it obtained in 7 out of the 24 problems and had the highest accuracy rate in an overall of 18 problems. Similar to this, BPCSA, BSCSA, and BCSA were exclusively best in 0, 2, and 3 problems, respectively, and had the greatest accuracies in 11, 14, and 12 problems, respectively. It is evident that BECSA, BPCSA, BSCSA, and BCSA achieved the optimum accuracy in one dataset, namely, Leukemia, with a value of 100%. In addition, BECSA achieved accuracy rates near to 100% in other datasets, including Diagnostic, Breast, BreastEW, Dermatology, and Thyroid. In these datasets, BECSA performed quite well, consistently locating a solution close to the near-optimal solution over 30 independent runs. In the Dermatology dataset, BPCSA achieved an accuracy value of 99.30%, whereas BCSA, BECSA, and BSCSA earned accuracy values of 99.20%, 99.67%, and 99.58%, respectively. For the Diagnostic dataset, BPCSA and BSCSA obtained an accuracy of 96.49%, while BECSA has obtained an accuracy of 96.55%. BSCSA was exclusive to obtain an accuracy rate of 95.29% in the Lymphography dataset, and BCSA arrived at an accuracy rate of 88.05% exclusively in the SPECT dataset. BCSA has the greatest accuracy in the Spectfheart, coming up at 91.70%, while BECSA, BPCSA, and BSCSA have accuracy values of 91.20%, 91.45%, and 91.32%, respectively. In the Thyroid dataset, BCSA has the greatest accuracy, coming in at 99.11%, while BECSA, BPCSA, and BSCSA have accuracy rates of 99.08%, 99.03%, and 99.06%, respectively. For COVID-19, the proposed BECSA, BSCSA, and BPCSA achieved a classification rate of 95.93%, where BCSA has realized a classification rate 95.89%. Furthermore, BCSA has an SD of 0.0015, while the others have an SD value of 0.0000. Retinopathy, ILPD, Cleveland, Saheart, and PimaDiabetess were among the datasets for which it was challenging to obtain accuracy levels near to 80%. This can be attributed to the difficult nature of these problems. Having said that, as we shall show in a moment, the proposed BECSA algorithm attained accuracy levels that were on par with or better than those reported in the literature. In terms of best outcomes, BECSA is able to obtain good solutions for all datasets under consideration, with the exception of the Cleveland dataset, where the performance was subpar which is 60.96%. This insinuates that BECSA is stuck in the local optimum solution, which, although not the best solution to this problem, is nonetheless not far from that best solution. In the Breast, Prognostic, Coimbra, ILPD, Saheart, Leukemia, PimaDiabetes, ProstateGE, COVID-19, and Colon datasets, the proposed BECSA, BSCSA, and BPCSA have the lowest standard deviations with a value of 0.0000. These findings support the effectiveness of BECSA, BSCSA, and BPCSA in achieving appropriate exploration and exploitation in addition to a proper balance between these two features.

The developed binary methods of CSA for FS problems have the goal of lowering the fitness scores produced by the criterion defined in Eq. 28. This is shown as the average fitness values in Table 4, which also includes the averages and standard deviations of the standard and developed algorithms of CSA over 30 independent runs for each of the 24 FS problems. Overall, Table 4 shows that BECSA is capable of finding the global optimal solution constantly and exclusively in 9 datasets and achieving the best fitness outcomes in an overall of 13 datasets. Evidently, it came in the first place by getting these outcomes. BCSA came in second place by getting the lowest average fitness values in seven problems and the lowest fitness scores over a number of ten datasets. The third place was kept for BSCSA, with best exclusive fitness values in 2 datasets and best fitness values in an overall of 8 problems. BPCSA was placed last, with no unique best fitness results in any dataset and best fitness outcomes in an overall of 6 datasets. For the Retinopathy, ILPD, Saheart, and PimaDiabetes datasets, although best solutions were not constantly found, the outcomes got are not far from the global optimum solution which can be substantiated by the very humble standard deviations obtained. BECSA had the best obtained fitness score of 0.0212 for the Parkinsons data set. This value is not too far from the values obtained by BPCSA and BSCSA where the values got by these two algorithms were 0.0189. For the Hepatitis dataset, BECSA revealed the best fitness of 0.0534, while BCSA, BSCSA, and BPCSA showed comparable small fitness values of 0.0631, 0.0556, and 0.0577, respectively.

It is important to note that higher sensitivity values correspond to higher levels of performance. Table 5 reveals that BECSA ranked first by having the top exclusive sensitivity findings across 9 of the 24 datasets. The BCSA algorithm came second by providing the highest exclusive sensitivity findings in 7 datasets (Prognostic, Coimbra, BreastEW, Lymphography, Cleveland, Hepatitis, and Saheart) and had the highest sensitivity outcomes in an overall of 8 problems, the eighth of which being the Dermatology dataset. BPCSA appeared in third place with best exclusive sensitivity outcomes in 4 problems, that is, Spectfheart, PimaDiabetes, Leukemia, and ProstateGE, whereas BSCSA appeared in the last place with best exclusive sensitivity outcomes only in 3 datasets, that is, SPECT, Heart, and Colon. Additionally, a sensitivity value of 99.09% was reported by both of BCSA and BECSA, compared to 99.08% and 98.88% reported by BPCSA and BSCSA for the Dermatology dataset, respectively. BCSA obtained an exclusive sensitivity result of 73.32% for the Coimbra dataset, while BECSA, BSCSA, and BPCSA recorded sensitivity values of 72.25%, 66.91%, and 66.95%, respectively. Returning to Table 5, one

can emphasize that there is a relatively large discrepancy between the findings produced by BCSA and those recorded by its developed partners.

Table 5 shows the specificity findings of the fundamental binary and proposed binary versions of CSA in relation to their average and standard deviation outcomes. It is obvious that BECSA emerged in the first place where it got the best exclusive specificity outcomes in 13 datasets. This binary variant of CSA is successively followed by BCSA, BSCSA, and BPCSA, with these versions had the best exclusive specificity results in 4, 4, and 3 datasets, respectively. There is only a slight difference between the results of these versions arrived at for the Breast, Lymphography, Cleveland, Hepatitis, Thyroid, Leukemia, and COVID-19 datasets. For the Coimbra dataset, there is an extremely little difference between the specificity result got by BECSA and that got by BPCSA, with the former having a specificity value of 82.02% and the latter having a specificity score of 81.99%. For the Diagnostic dataset, the proposed BECSA and BPCSA algorithms reported specificity values of 97.03% and 96.63%, respectively, while BCSA and BSCSA reported specificity values of 96.25% and 96.49%, respectively. There are either zero or almost zero values for the SD results in Table 5, which indicates the robustness of the proposed FS methods.

The average number of features got in the classification of 24 datasets for each of the binary algorithms proposed for CSA is presented in Table 6. Comparison of the four proposed FS methods in Table 6 reveals a significant difference in the results. BCSA was the most beneficial FS method in reducing the features need for classification, where, compared to the other algorithms, the average number of attributes that were selected was the fewest. BCSA has the exclusive fewest number of features in 13 out of 24 datasets, with an overall of lowest average number of selected features is in 15 datasets. BECSA has the exclusive minimal number of features in the BreastEW and Spectfheart datasets, shared the minimal number of chosen features for the Breast and PimaDiabetes datasets with BCSA, BPCSA, and BSCSA, and shared the minimum number of selected features for Coimbra, ILPD, Saheart with BPCSA and BSCSA. The proposed BPCSA exclusively captured the minimum number of features in the Retinopathy and Lymphography datasets, and BSCSA exclusively captured the minimum number of features in the SPECT and Cleveland datasets.

Finally, it is preferable to determine the relative performance of one or all of the developed FS methods when their performance levels are compared to those of other FS methods. This is because the absolute performance of the proposed FS methods compared to each other is not fair to grade these methods. In this, it is clear from a review of the results in Tables 4, 5, and 6 that BECSA beat all competing algorithms in all evaluation criteria. Because of this, the outcomes of both the basic BCSA and the proposed BECSA are compared with those of widely used algorithms stated in the pertinent literature, as presented later in a subsection below.

Limitations of the Proposed Methods

Although the proposed FS algorithms of CSA have shown outstanding performance levels in addressing low, medium, and high-dimensional FS problems in binary search spaces, they do not guarantee overall optimality. Furthermore, for several datasets including Retinopathy, ILPD, Cleveland, Saheart, and PimaDiabetes, the classification accuracy, sensitivity, and specificity rates were essentially modest. The corresponding fitness values for these datasets on the basis of the proposed FS methods are not small to the extent required. According to this, the proposed FS algorithms have some limitations, such as falling into local optimal values or departing from the global optimal solution, which impacts how well they perform when tackling complex FS problems with a large number of features and dimensions. Besides, the number of features selected by the proposed FS in some datasets methods for the classification tasks is reasonable and is not better than that of the basic binary version of CSA. Despite having good performance in the majority of the test FS problems, the proposed FS methods may experience poor convergence rates and may become stuck in the local optimum in certain datasets, such as in the case of the Cleveland dataset. The sensitivity and specificity results for some datasets such as Retinopathy, Lymphography, SPECT, and Cleveland are not as large as needed. In order to cope with these limitations, more works on how to make the proposed FS methods perform better may be considered. By enhancing these proposed algorithms' exploration and exploitation capabilities within the binary search spaces of the relevant datasets, these issues might be avoided.

Sensitivity Analysis

A thorough sensitivity analysis based on the Design of Experiment (DoE) technique was carried out in order to pinpoint the best parameter settings of the proposed FS methods by which the experimental results can be greatly affected. The proposed FS methods that employ k-NN as a classifier used DoE to examine the sensitivity of the key control parameters (β_0, and β_1). The ranges of the key control parameters were first established, and the values of these parameters were defined to assess if the best values fell within the range or whether more experiments were required. The experiments using FS methods then employed a parameter with one input of the generated DoE values in the designated range, leaving the other parameters at their starting values. These experiments were performed in a systematic manner

in order to study the influence of the input parameters on the accuracy level of the proposed binary algorithms and to arrive at a sensible solution. In this situation, a comprehensive analysis was conducted using the previously indicated parameters, and the average classification accuracy was obtained for each experiment for each proposed FS method for all datasets. The values of each parameter used in this experimental analysis are as follows: (1) the control parameters for BECSA are identified to be $\beta_0 = 0.5, 1.0, 1.5, 2.0$ and $\beta_1 = 0.5, 1.0, 1.5, 2.0$; (2) the control parameters for BPCSA are identified to be $\beta_0 = 0.5, 1.0, 1.5, 2.0$ and $\beta_1 = 0.1, 0.3, 0.5, 1.0$; (3) the control parameters of BSCSA are identified to be $\beta_0 = 0.5, 1.0, 1.5, 2.0$ and $\beta_1 = 0.5, 1.0, 1.5, 2.0$. The number of capuchins and the maximum iterations in this study were set to 100 and 30, respectively, correlated with 30 separate runs.

1. The control parameters β_0 and β_1 of BECSA: the proposed BECSA was tested for various values of the parameters β_0 and β_1. The average classification accuracy of BECSA with different values for β_0 and β_1 when employed to solve the feature selection problems under investigation, with the other parameters left intact, is shown in Table 7. The computational results in Table 7 show that BECSA presented the best classification rates when the parameters β_0 and β_1 of BECSA are equal to 2.0 and 1.0, respectively. This demonstrates the significance of employing a sensible range of these crucial parameters to augment the robustness of BECSA.
2. The control parameters β_0 and β_1 of BPCSA: the proposed BPCSA was tested for a variety of range values of β_0 and β_1. Table 8 displays the average classification accuracy rate of BPCSA when applied to the feature selection problems under study, while the other parameters were left unchanged. The optimal classification accuracy for BPCSA is obvious from Table 8 when the values of the parameters β_0 and β_1 are equal to 2.0 and 0.1, respectively. This highlights the need to study the BPCSA's sensitivity to various values of this parameter.
3. The control parameters β_0 and β_1 of BECSA: the proposed BSCSA was tested for a variety values of the parameters β_0 and β_1. The average classification accuracy of BECSA when applied to the 24 feature selection problems under study, with varying the values of β_0 and β_1 and with all other parameter values left unaltered, is shown in Table 9. Table 9 clearly shows that the classification accuracy results on the basis of the values of β_0 and β_1 change considerably, demonstrating how sensitive the BSCSA is to these parameters. Additionally, it should be noted that the values of β_0 and β_1 were adjusted to 2.0 and 1.0, respectively, for BSCSA to produce the optimal classification accuracy.

The most sensitive values out of all those provided in Tables 7, 8, and 9 were chosen after sensitivity analysis. As mentioned, while addressing feature selection problems, the classification accuracy of the proposed BECSA, BPCSA, and BSCSA can be impacted by different values of the parameters β_0 and β_1. For datasets with different degrees of dimensionality, the standard deviation values of the classification accuracy of BECSA, BPCSA, and BSCSA are low, indicating that these proposed methods are almost stable to changes in the relevant control parameters for each corresponding method. In sum, the findings in Tables 7, 8, and 9 demonstrate that BECSA, BPCSA, and BSCSA are usually quite sensitive to the parameter settings of $\beta 0$ and $\beta 1$.

Performance Comparison with Other Methods

To study the performance of the proposed BECSA in depth, its findings on FS problems were compared with those of standard BCSA and other FS methods, namely, Binary Biogeography-Based Optimization (BBBO) [56], Binary Moth-Flame Optimization (BMFO) algorithm [57], Binary Teaching-Learning-Based Optimization (BTLBO) [58], Binary Success-History based Adaptive Differential Evolution with Linear population size reduction (BLSHADE) [59], Binary Particle Swarm Optimization (BPSO) [60], Binary Ali Baba and the Forty Thieves (BAFT) algorithm [1], and Binary Honey Badger Algorithm (BHBA) [61]. Not only were the details of the experimental results provided in this comparison, but also an in-depth and careful comparison with those comparative optimization methods was made. The same experimental conditions, including maximum number of iterations and population size, were applied to all of the examined FS methods in order to make an equitable comparison of the proposed BECSA with the comparative optimization methods. Table 10 contains the parameter settings for each of the competing algorithms.

The experimental runs were performed 30 times in total, independently, to get statistically meaningful findings. Based on the overall capabilities and findings reached throughout these runs, the statistical results were then obtained. The number of attributes in each dataset is the same as the dimension of each associated dataset. Classification accuracy, sensitivity, specificity, fitness values, and number of selected features were used to evaluate the performance of the proposed BECSA, and this performance was compared to the performance of other methods in terms of all the above criteria. The best outcomes are highlighted in bold in all comparison tables to give them more weight than the other findings. In Table 11, the average classification accuracy results for the basic BCSA, the proposed BECSA, and other competing methods are tabulated along with their respective standard derivation results.

Table 7 Classification accuracy of BECSA using different values of β_0 and β_1 with the k-NN classifier

Dataset	Measure	$\beta_0 = 0.5$	$\beta_0 = 1.0$	$\beta_0 = 1.5$	$\beta_0 = 2.0$	$\beta_1 = 0.5$	$\beta_1 = 1.0$	$\beta_1 = 1.5$	$\beta_1 = 2.0$
Diagnostic	AV	0.7787	0.8149	0.8401	0.9645	0.7305	0.9559	0.8476	0.6759
	SD	0.0032	0.0032	0.0027	0.0033	0.0032	0.0032	0.0027	0.0031
Breast	AV	0.8006	0.8377	0.8637	0.9916	0.7510	0.9828	0.8714	0.6948
	SD	0.0036	0.0033	0.0030	0.0011	0.0041	0.0017	0.0031	0.0023
Prognostic	AV	0.7216	0.7551	0.7785	0.8938	0.6769	0.8858	0.7854	0.6263
	SD	0.0039	0.0033	0.0036	0.0032	0.0027	0.0019	0.0024	0.0027
Coimbra	AV	0.7364	0.7706	0.7944	0.9121	0.6907	0.9040	0.8015	0.6391
	SD	0.0037	0.0034	0.0035	0.0022	0.0043	0.0024	0.0029	0.0045
BreastEW	AV	0.7995	0.8366	0.8624	0.9902	0.7499	0.9814	0.8702	0.6938
	SD	0.0038	0.0035	0.0033	0.0029	0.0011	0.0120	0.0018	0.0100
Retinopathy	AV	0.5769	0.6036	0.6223	0.7145	0.5411	0.7081	0.6279	0.5006
	SD	0.0038	0.0038	0.0032	0.0039	0.0038	0.0038	0.0032	0.0037
Dermatology	AV	0.8039	0.8412	0.8672	0.9957	0.7541	0.9868	0.8750	0.6977
	SD	0.0073	0.0073	0.0061	0.0074	0.0072	0.0072	0.0061	0.0071
ILPD	AV	0.6102	0.6385	0.6582	0.7557	0.5723	0.7490	0.6641	0.5296
	SD	0.0121	0.0114	0.0114	0.0100	0.0219	0.0123	0.0201	0.0223
Lymphography	AV	0.7667	0.8023	0.8271	0.9496	0.7192	0.9412	0.8345	0.6654
	SD	0.0209	0.0209	0.0174	0.0211	0.0207	0.0205	0.0174	0.0203
Parkinsons	AV	0.7907	0.8274	0.8529	0.9793	0.7417	0.9706	0.8606	0.6862
	SD	0.0132	0.0132	0.0110	0.0133	0.0131	0.0130	0.0110	0.0128
ParkinsonC	AV	0.6574	0.6878	0.7091	0.8142	0.6166	0.8069	0.7155	0.5705
	SD	0.0236	0.0236	0.0197	0.0239	0.0234	0.0232	0.0197	0.0230
SPECT	AV	0.7087	0.7415	0.7645	0.8777	0.6647	0.8699	0.7713	0.6150
	SD	0.0114	0.0114	0.0095	0.0115	0.0113	0.0112	0.0095	0.0111
Cleveland	AV	0.4917	0.5145	0.5304	0.6090	0.4612	0.6036	0.5352	0.4267
	SD	0.0037	0.0037	0.0031	0.0038	0.0037	0.0037	0.0031	0.0036
HeartEW	AV	0.7309	0.7648	0.7885	0.9053	0.6856	0.8972	0.7955	0.6343
	SD	0.0056	0.0056	0.0047	0.0057	0.0056	0.0055	0.0047	0.0055
Hepatitis	AV	0.7645	0.8000	0.8248	0.9469	0.7172	0.9385	0.8321	0.6635
	SD	0.0284	0.0284	0.0238	0.0287	0.0281	0.0279	0.0237	0.0276
Saheart	AV	0.5698	0.5963	0.6147	0.7058	0.5345	0.6995	0.6202	0.4946
	SD	0.0241	0.0238	0.0228	0.0198	0.0281	0.0198	0.0187	0.0309
Spectfheart	AV	0.7356	0.7697	0.7935	0.9111	0.6900	0.9030	0.8006	0.6384
	SD	0.0137	0.0137	0.0114	0.0138	0.0135	0.0134	0.0114	0.0133
Thyroid	AV	0.7991	0.8362	0.8621	0.9898	0.7496	0.9810	0.8698	0.6936
	SD	0.0010	0.0010	0.0008	0.0010	0.0009	0.0009	0.0008	0.0009
Heart	AV	0.7170	0.7502	0.7734	0.8880	0.6725	0.8801	0.7803	0.6222
	SD	0.0192	0.0192	0.0160	0.0194	0.0190	0.0188	0.0160	0.0186
PimaDiabetes	AV	0.6010	0.6288	0.6483	0.7443	0.5637	0.7377	0.6541	0.5216
	SD	0.0238	0.0229	0.218	0.0187	0.0252	0.0189	0.0210	0.0278
Leukemia	AV	0.8066	0.8440	0.8701	0.9990	0.7566	0.9901	0.8779	0.7000
	SD	0.0045	0.0038	0.0030	0.0035	0.0009	0.0023	0.0013	0.0018
Colon	AV	0.7394	0.7737	0.7976	0.9158	0.6935	0.9076	0.8048	0.6417
	SD	0.0043	0.0040	0.0038	0.0012	0.0048	0.0015	0.0033	0.0056
ProstateGE	AV	0.7259	0.7596	0.7831	0.8991	0.6809	0.8911	0.7901	0.6300
	SD	0.0046	0.0033	0.0043	0.0033	0.0053	0.0033	0.0034	0.0058
COVID-19	AV	0.7737	0.8096	0.8347	0.9583	0.7258	0.9498	0.8421	0.6715
	SD	0.0032	0.0035	0.0033	0.0010	0.0036	0.0009	0.0028	0.0033

Table 8 Classification accuracy of BPCSA using different values of β_0 and β_1 with the k-NN classifier

Dataset	Measure	$\beta_0 = 0.5$	$\beta_0 = 1.0$	$\beta_0 = 1.5$	$\beta_0 = 2.0$	$\beta_1 = 0.1$	$\beta_1 = 0.3$	$\beta_1 = 0.5$	$\beta_1 = 1.0$
Diagnostic	AV	0.7399	0.7794	0.8642	0.9560	0.9543	0.6732	0.6541	0.5789
	SD	0.0019	0.0019	0.0016	0.0020	0.0019	0.0019	0.0016	0.0019
Breast	AV	0.7611	0.8018	0.8890	0.9834	0.9817	0.6925	0.6729	0.5956
	SD	0.0040	0.0037	0.0030	0.0014	0.0012	0.0055	0.0060	0.0086
Prognostic	AV	0.6860	0.7227	0.8013	0.8864	0.8848	0.6242	0.6065	0.5368
	SD	0.0054	0.0047	0.0042	0.0028	0.0029	0.0056	0.0063	0.0068
Coimbra	AV	0.7001	0.7375	0.8177	0.9046	0.9029	0.6370	0.6189	0.5478
	SD	0.0056	0.0053	0.0045	0.0023	0.0025	0.0060	0.0064	0.0069
BreastEW	AV	0.7596	0.8002	0.8872	0.9815	0.9797	0.6911	0.6715	0.5944
	SD	0.0027	0.0027	0.0023	0.0027	0.0026	0.0026	0.0022	0.0026
Retinopathy	AV	0.5479	0.5772	0.6400	0.7080	0.7067	0.4986	0.4844	0.4288
	SD	0.0047	0.0047	0.0040	0.0048	0.0047	0.0046	0.0039	0.0046
Dermatology	AV	0.7614	0.8021	0.8894	0.9838	0.9821	0.6928	0.6731	0.5958
	SD	0.0087	0.0087	0.0074	0.0088	0.0086	0.0086	0.0072	0.0085
ILPD	AV	0.5801	0.6111	0.6776	0.7495	0.7482	0.5278	0.5128	0.4539
	SD	0.0123	0.0119	0.0107	0.0101	0.0100	0.0197	0.0209	0.0245
Lymphography	AV	0.7244	0.7632	0.8462	0.9361	0.9344	0.6592	0.6405	0.5669
	SD	0.0208	0.0209	0.0176	0.0210	0.0206	0.0204	0.0173	0.0202
Parkinsons	AV	0.7537	0.7940	0.8803	0.9738	0.9721	0.6857	0.6663	0.5897
	SD	0.0149	0.0149	0.0126	0.0150	0.0147	0.0146	0.0124	0.0145
ParkinsonC	AV	0.6141	0.6469	0.7173	0.7935	0.7921	0.5588	0.5429	0.4805
	SD	0.0253	0.0253	0.0214	0.0255	0.0251	0.0248	0.0210	0.0246
SPECT	AV	0.6741	0.7102	0.7875	0.8711	0.8695	0.6134	0.5960	0.5275
	SD	0.0114	0.0114	0.0096	0.0115	0.0113	0.0112	0.0094	0.0111
Cleveland	AV	0.4666	0.4915	0.5450	0.6029	0.6018	0.4245	0.4125	0.3651
	SD	0.0063	0.0063	0.0053	0.0064	0.0062	0.0062	0.0052	0.0061
HeartEW	AV	0.6925	0.7295	0.8089	0.8948	0.8932	0.6301	0.6122	0.5419
	SD	0.0097	0.0097	0.0082	0.0098	0.0096	0.0095	0.0080	0.0094
Hepatitis	AV	0.7237	0.7624	0.8453	0.9351	0.9334	0.6585	0.6398	0.5663
	SD	0.0232	0.0232	0.0195	0.0233	0.0229	0.0227	0.0192	0.0225
Saheart	AV	0.5417	0.5707	0.6328	0.7000	0.6987	0.4929	0.4789	0.4239
	SD	0.0087	0.0033	0.0080	0.0076	0.0065	0.0064	0.0086	0.0095
Spectfheart	AV	0.7012	0.7387	0.8191	0.9061	0.9044	0.6380	0.6199	0.5487
	SD	0.0156	0.0156	0.0132	0.0158	0.0155	0.0153	0.0130	0.0152
Thyroid	AV	0.7593	0.7999	0.8870	0.9812	0.9794	0.6909	0.6713	0.5942
	SD	0.0013	0.0013	0.0011	0.0013	0.0013	0.0013	0.0011	0.0013
Heart	AV	0.6799	0.7162	0.7942	0.8785	0.8769	0.6186	0.6011	0.5320
	SD	0.0201	0.0201	0.0170	0.0203	0.0199	0.0197	0.0167	0.0195
PimaDiabetes	AV	0.5713	0.6019	0.6673	0.7382	0.7369	0.5198	0.5051	0.4471
	SD	0.0067	0.0064	0.0058	0.0044	0.0045	0.0076	0.0079	0.0088
Leukemia	AV	0.7668	0.8078	0.8956	0.9908	0.9890	0.6977	0.6779	0.6000
	SD	0.0047	0.0043	0.0031	0.0008	0.0011	0.0050	0.0054	0.0069
Colon	AV	0.7029	0.7405	0.8210	0.9082	0.9066	0.6396	0.6214	0.5500
	SD	0.0043	0.0040	0.0024	0.0013	0.0014	0.0046	0.0047	0.0056
ProstateGE	AV	0.6901	0.7270	0.8061	0.8917	0.8901	0.6279	0.6101	0.5400
	SD	0.0045	0.0043	0.0034	0.0019	0.0020	0.0056	0.0057	0.0063
COVID-19	AV	0.7356	0.7749	0.8592	0.9505	0.9487	0.6693	0.6503	0.5756
	SD	0.0041	0.0037	0.0020	0.0009	0.0010	0.0049	0.0051	0.0057

Table 9 Classification accuracy of BSCSA using different values of β_0 and β_1 with the k-NN classifier

Dataset	Measure	$\beta_0 = 0.5$	$\beta_0 = 1.0$	$\beta_0 = 1.5$	$\beta_0 = 2.0$	$\beta_1 = 0.5$	$\beta_1 = 1.0$	$\beta_1 = 1.5$	$\beta_1 = 2.0$
Diagnostic	AV	0.7836	0.8256	0.8383	0.9553	0.7600	0.9551	0.8148	0.7549
	SD	0.0019	0.0019	0.0016	0.0019	0.0019	0.0019	0.0016	0.0019
Breast	AV	0.8061	0.8493	0.8624	0.9828	0.7818	0.9826	0.8382	0.7765
	SD	0.0043	0.0040	0.0039	0.0013	0.0034	0.0014	0.0035	0.0036
Prognostic	AV	0.7266	0.7656	0.7773	0.8858	0.7047	0.8857	0.7555	0.6999
	SD	0.0045	0.0043	0.0041	0.0022	0.0047	0.0022	0.0042	0.0049
Coimbra	AV	0.7415	0.7812	0.7932	0.9040	0.7191	0.9038	0.7710	0.7143
	SD	0.0047	0.0044	0.0043	0.0013	0.0052	0.0013	0.0043	0.0053
BreastEW	AV	0.8050	0.8481	0.8611	0.9814	0.7807	0.9812	0.8370	0.7754
	SD	0.0037	0.0032	0.0024	0.0009	0.0042	0.0010	0.00383	0.0036
Retinopathy	AV	0.5805	0.6116	0.6210	0.7077	0.5630	0.7076	0.6036	0.5592
	SD	0.0035	0.0035	0.0030	0.0035	0.0034	0.0034	0.0030	0.0034
Dermatology	AV	0.8087	0.8521	0.8651	0.9859	0.7843	0.9857	0.8409	0.7790
	SD	0.0080	0.0080	0.0067	0.0080	0.0078	0.0078	0.0068	0.0078
ILPD	AV	0.6144	0.6473	0.6572	0.7490	0.5959	0.7489	0.6388	0.5918
	SD	0.0066	0.0063	0.0060	0.0051	0.0071	0.0047	0.0061	0.0073
Lymphography	AV	0.7739	0.8154	0.8279	0.9435	0.7506	0.9433	0.8047	0.7455
	SD	0.0204	0.0204	0.0173	0.0205	0.0199	0.0201	0.0173	0.0201
Parkinsons	AV	0.7982	0.8410	0.8539	0.9732	0.7742	0.9730	0.8300	0.7689
	SD	0.0149	0.0149	0.0126	0.0149	0.0145	0.0146	0.0126	0.0146
ParkinsonC	AV	0.6525	0.6875	0.6981	0.7955	0.6329	0.7954	0.6785	0.6286
	SD	0.0277	0.0277	0.0234	0.0278	0.0270	0.0272	0.0234	0.0272
SPECT	AV	0.7140	0.7523	0.7638	0.8705	0.6925	0.8703	0.7424	0.6878
	SD	0.0141	0.0141	0.0119	0.0142	0.0138	0.0139	0.0120	0.0139
Cleveland	AV	0.4956	0.5221	0.5301	0.6042	0.4806	0.6040	0.5153	0.4774
	SD	0.0096	0.0083	0.0080	0.0071	0.0091	0.0077	0.0081	0.0093
HeartEW	AV	0.7319	0.7711	0.7830	0.8923	0.7098	0.8921	0.7610	0.7050
	SD	0.0122	0.0122	0.0103	0.0122	0.0119	0.0120	0.0103	0.0120
Hepatitis	AV	0.7681	0.8093	0.8217	0.9364	0.7450	0.9362	0.7987	0.7399
	SD	0.0261	0.0261	0.0221	0.0262	0.0255	0.0257	0.0221	0.0257
Saheart	AV	0.5738	0.6045	0.6138	0.6995	0.5565	0.6994	0.5966	0.5527
	SD	0.0086	0.0083	0.0080	0.0081	0.0091	0.0077	0.0076	0.0089
Spectfheart	AV	0.7416	0.7814	0.7934	0.9042	0.7193	0.9040	0.7712	0.7144
	SD	0.0141	0.0141	0.0119	0.0142	0.0138	0.0139	0.0120	0.0139
Thyroid	AV	0.8045	0.8476	0.8606	0.9808	0.7803	0.9806	0.8365	0.7750
	SD	0.0008	0.0008	0.0007	0.0008	0.0008	0.0008	0.0007	0.0008
Heart	AV	0.7210	0.7597	0.7713	0.8790	0.6993	0.8788	0.7497	0.6945
	SD	0.0197	0.0197	0.0166	0.0198	0.0192	0.0194	0.0167	0.0194
PimaDiabetes	AV	0.6051	0.6376	0.6473	0.7377	0.5869	0.7376	0.6292	0.5829
	SD	0.0056	0.0053	0.0050	0.0045	0.0061	0.0057	0.0051	0.0063
Leukemia	AV	0.8121	0.8557	0.8688	0.9901	0.7877	0.9899	0.8445	0.7823
	SD	0.0027	0.0023	0.0020	0.0007	0.0031	0.0013	0.0019	0.0032
Colon	AV	0.7445	0.7844	0.7964	0.9076	0.7220	0.9074	0.7741	0.7172
	SD	0.0042	0.0039	0.0037	0.0015	0.0046	0.0015	0.0038	0.0048
ProstateGE	AV	0.7309	0.7701	0.7819	0.8911	0.7089	0.8909	0.7600	0.7041
	SD	0.0046	0.0040	0.0039	0.0023	0.0045	0.0027	0.0037	0.0046
COVID-19	AV	0.7791	0.8208	0.8334	0.9498	0.7556	0.9496	0.8101	0.7505
	SD	0.0047	0.0042	0.0040	0.0014	0.0048	0.0013	0.0035	0.003s3

Table 10 Parameter settings of the comparative binary algorithms

Algorithm	Parameter	Value
All algorithms	Population size, number of iterations	30, 100
BBBO	Habitat modification probability	1
	Immigration probability bounds for each gene	[0, 1]
	Mutation probability	0
	Maximum migration and immigration rates for each island	1
	Step size for numerical integration of probabilities	1
BMFO	Convergence constant	[−1, −2]
	Logarithmic spiral	0.75
BPSO	Inertia weight (ω)	[0.9, 0.4]
	Cognitive factor (c_1), Social factor (c_2)	1.8, 2.0
BTLBO	No especial parameters	Population size is 5 as this method has two stages
BAFT	Td_0, s, L	2.0, 2.0, 100
	α_0,α_1,β_0,β_1	1.0, 2.0, 0.1, 2.0
BLSHADE	Crossover rateM_{CR}	0.5
	Scaling factorM_F	0.5
	Archive rate, memory size	1.4, 5
	Pbest	0.11
BHBA	The ability of the honey badger to get food (β), C	6.0, 2

Again, a higher value of the accuracy results means better robustness, while a lower value of SD reflects the stability of the algorithm. It can be seen from Table 11 that the proposed BECSA ranked first exclusively by having the best accuracy results in 5 datasets and achieved the highest accuracy in a total of 13 datasets. BCSA ranked second by exclusively acquiring the best accuracy results in 3 datasets and realized the highest accuracy in a total of 10 datasets. BHBA came in third by exclusively achieving the best results in 5 datasets and achieved the highest accuracy in a total of 11 datasets. BBBO ranked fourth by having the best results exclusively in 2 datasets, namely, the Cleveland and Hepatitis. BLSHADE came in fifth place by achieving the best results in 3 datasets, while BAFT came in sixth by fulfilling the best result only in ProstateGE dataset. BPSO, BMFO, and BTLBO did not achieve the best results in any of the datasets. However, both BCSA and BECSA obtained 100% accuracy in the Leukemia dataset. When reading the SD results recorded in Table 11, it can be noticed that the proposed BECSA was more robust than other rivals by having the minimum SD values in 13 out of the 24 datasets.

The average fitness value and standard deviation of all rivals in each dataset are presented in Table 12.

It should be perceived that the minimum fitness values reveal better performance for the optimization algorithms. From Table 12, it can be observed that BECSA, BCSA, and BHBA ranked first, second, and third, respectively, where each exclusively got the minimum fitness values in an overall of 7 datasets. BTLBO ranked fourth by exclusively achieving the minimum fitness values in 3 datasets, namely, Parkinsons, Cleveland, and Hepatitis. Finally, BLSHADE, BBBO, BAFT, BMFO, and BPSO did not report any lowest fitness results in any of the datasets examined in this work, where they ranked fifth, sixth, seventh, eighth, and ninth, respectively. When reading the standard derivation results tabulated in Table 12 one more time, it can be shown that the performance of BECSA is more powerful than the other competitors by acquisition the lowest SD results in 11 out of 24 datasets.

The sensitivity results of the proposed ECSA compared to BCSA and the other comparative methods are given in Table 13.

Then, we move on to compare the proposed BECSA with other competing algorithms in respect of the sensitivity results that the FS algorithms intend to increase. It should be noted that higher sensitivity results mean preferable performance level. The results of these competing algorithms are summarized in terms of the average sensitivity results associated with their standard derivation values in Table 13. Regarding the results presented in this table, without a doubt, BLSHADE has the largest sensitivity figures compared to all other rivals. Specifically, BLSHADE exclusively achieved the highest sensitivity values in 12 out of 24 datasets. Evidently, BCSA ranked second by obtaining the best sensitivity results in 3 out of 24 datasets. BHBA ranked third even though it did not achieve any optimal sensitivity result than all other competing algorithms, but its results are rather high, while the

Table 11 Comparison results between the proposed BECSA and other methods based on classification accuracy

Dataset	Measure	BCSA	BECSA	BBBO	BMFO	BPSO	BTLBO	BAFT	BLSHADE	BHBA
Diagnostic	AV	0.9590	0.9655	0.9634	0.9431	0.8575	0.8044	0.9540	0.9699	**0.9823**
	SD	0.0085	0.0027	0.0086	0.0731	0.1112	0.1044	0.0074	0.0101	**0.0000**
Breast	AV	**0.9926**	**0.9926**	0.9730	0.9191	0.8502	0.7331	0.9824	0.9868	**0.9926**
	SD	**0.0000**	**0.0000**	0.0035	0.1213	0.1558	0.1661	0.0066	0.0033	**0.0000**
Prognostic	AV	**0.8947**	**0.8947**	0.8184	0.7754	0.7211	0.7018	0.8895	**0.8947**	**0.8947**
	SD	**0.0000**	**0.0000**	0.0144	0.0669	0.0506	0.0348	0.0118	**0.0000**	**0.0000**
Coimbra	AV	**0.9130**	**0.9130**	0.8783	0.7971	0.6522	0.6014	0.8174	0.8261	0.8261
	SD	**0.0000**	**0.0000**	0.0289	0.1258	0.2157	0.1827	0.0194	**0.0000**	**0.0000**
BreastEW	AV	**0.9912**	**0.9912**	0.9693	0.9510	0.9322	0.9348	0.9327	0.9398	0.9487
	SD	**0.0000**	**0.0000**	0.0056	0.0181	0.0165	0.0205	0.0048	0.0040	0.0040
Retinopathy	AV	0.7123	0.7152	0.7077	0.7236	0.6306	0.6225	0.6939	0.7139	**0.7417**
	SD	0.0046	0.0032	0.0108	0.0359	0.0511	0.0236	0.0196	0.0200	**0.0024**
Dermatology	AV	0.9920	**0.9967**	0.9901	0.9455	0.8737	0.8286	0.9859	0.9887	0.9944
	SD	0.0071	0.0061	0.0066	0.0654	0.0847	0.0868	**0.0000**	0.0063	0.0077
ILPD	AV	0.7557	0.7565	0.7371	0.7130	0.6899	0.6896	0.7374	0.7461	**0.7739**
	SD	0.0027	**0.0000**	0.0118	0.0549	0.0266	0.0284	0.0167	0.0113	**0.0000**
Lymphography	AV	0.9448	**0.9506**	0.8575	0.7908	0.7333	0.7310	0.5389	0.5594	0.6402
	SD	0.0172	0.0174	**0.0150**	0.0646	0.0701	0.0668	0.0415	0.0437	0.0208
Parkinsons	AV	0.9778	0.9803	**0.9974**	0.9718	0.9231	0.9094	0.9744	0.9744	0.9744
	SD	0.0089	0.0110	0.0103	0.0483	0.0476	0.0385	**0.0000**	**0.0000**	**0.0000**
ParkinsonC	AV	0.8022	**0.8150**	0.6956	0.6916	0.6826	0.6770	0.7483	0.7550	0.7550
	SD	0.0346	0.0197	0.0021	0.0118	0.0087	0.0057	0.0094	**0.0000**	**0.0000**
SPECT	AV	**0.8805**	0.8786	0.7377	0.7006	0.6252	0.6088	0.7585	0.7849	0.8377
	SD	0.0114	**0.0095**	0.0181	0.0445	0.0573	0.0615	0.0207	0.0215	0.0169
Cleveland	AV	0.6017	0.6096	**0.6808**	0.6661	0.6113	0.5910	0.5729	0.5763	0.5763
	SD	0.0116	0.0031	0.0100	0.0500	0.0477	0.0419	0.0076	**0.0000**	**0.0000**
HeartEW	AV	0.8951	0.9062	0.8395	0.8463	0.7414	0.6796	0.8667	0.8741	**0.9630**
	SD	0.0156	0.0047	0.0301	0.0567	0.1028	0.0672	0.0275	0.0304	**0.0000**
Hepatitis	AV	0.9375	0.9479	**1.0000**	0.9708	0.9208	0.8792	0.8750	0.8750	0.9250
	SD	**0.0000**	0.0237	**0.0000**	0.0426	0.0769	0.0675	**0.0000**	**0.0000**	0.0280
Saheart	AV	0.7065	0.7065	0.7246	0.6953	0.6054	0.6022	0.7109	0.7304	**0.7609**
	SD	**0.0000**	**0.0000**	0.0096	0.0491	0.0476	0.0531	0.0182	0.0179	**0.0000**
Spectfheart	AV	**0.9170**	0.9120	0.8516	0.8063	0.7597	0.7497	0.8679	0.8717	0.9094
	SD	0.0161	0.0114	0.0155	0.0470	0.0547	0.0364	0.0267	0.0207	**0.0084**
Thyroid	AV	**0.9911**	0.9908	0.9751	0.9737	0.9542	0.9401	0.9800	0.9836	0.9875
	SD	0.0013	**0.0008**	0.0036	0.0171	0.0192	0.0105	0.0023	**0.0008**	0.0010
Heart	AV	0.8778	**0.8889**	0.8233	0.8222	0.7161	0.6633	0.7867	0.8067	0.8500
	SD	0.0160	0.0160	0.0230	0.0552	0.0864	0.0683	0.0361	0.0190	**0.0000**
PimaDiabetes	AV	0.7451	0.7451	0.7712	0.7495	0.7004	0.7192	0.7765	**0.7974**	**0.7974**
	SD	**0.0000**	**0.0000**	**0.0000**	0.0300	0.0679	0.0490	0.0286	**0.0000**	**0.0000**
Leukemia	AV	**1.0000**	**1.0000**	0.9952	0.9524	0.9238	0.9262	0.9286	0.9429	0.9857
	SD	**0.0000**	**0.0000**	0.0181	0.0433	0.0261	0.0130	**0.0000**	0.0319	0.0319
Colon	AV	**0.9167**	**0.9167**	0.7222	0.7222	0.6139	0.5500	0.8333	0.8667	**0.9167**
	SD	**0.0000**	**0.0000**	0.0505	0.0911	0.0742	0.0603	**0.0000**	0.0456	**0.0000**
ProstateGE	AV	**0.9000**	**0.9000**	0.8517	0.8117	0.7700	0.7517	**0.9000**	**0.9000**	**0.9000**
	SD	**0.0000**	**0.0000**	0.0091	0.0468	0.0466	0.0404	**0.0000**	**0.0000**	**0.0000**
COVID-19	AV	0.9589	**0.9593**	0.9543	0.9465	0.9188	0.9097	0.9523	0.9581	**0.9593**
	SD	0.0015	**0.0000**	0.0025	0.0234	0.0219	0.0242	0.0026	0.0026	**0.0000**

Table 12 Comparison results between the proposed BECSA and other methods based on fitness values

Dataset	Measure	BCSA	BECSA	BBBO	BMFO	BPSO	BTLBO	BAFT	BLSHADE	BHBA
Diagnostic	AV	0.0423	0.0369	0.0413	0.0599	0.1446	0.0346	0.0512	0.0354	**0.0217**
	SD	0.0081	0.0027	0.0088	0.0725	0.1102	0.0049	0.0080	0.0103	**0.0004**
Breast	AV	0.0123	0.0123	0.0308	0.0832	0.1524	0.0293	0.0217	0.0179	**0.0107**
	SD	**0.0000**	**0.0000**	0.0033	0.1204	0.1542	0.0034	0.0073	0.0039	0.0005
Prognostic	AV	**0.1070**	0.1076	0.1848	0.2266	0.2802	0.1775	0.1148	0.1089	0.1080
	SD	0.0012	0.0005	0.0141	0.0661	0.0500	0.0143	0.0118	**0.0004**	0.0006
Coimbra	AV	0.0908	**0.0905**	0.1253	0.2049	0.3492	0.1234	0.1879	0.1797	0.1762
	SD	0.0005	**0.0000**	0.0282	0.1242	0.2137	0.0289	0.0190	0.0009	0.0015
BreastEW	AV	0.0132	**0.0128**	0.0366	0.0540	0.0723	0.0398	0.0728	0.0650	0.0563
	SD	0.0011	**0.0004**	0.0053	0.0180	0.0160	0.0055	0.0042	0.0042	0.0037
Retinopathy	AV	0.2889	0.2859	0.2951	0.2777	0.3700	0.2813	0.3083	0.2884	**0.2606**
	SD	0.0046	**0.0024**	0.0110	0.0359	0.0503	0.0219	0.0194	0.0198	0.0027
Dermatology	AV	0.0122	**0.0080**	0.0156	0.0584	0.1298	0.0172	0.0211	0.0172	0.0106
	SD	0.0062	0.0055	0.0061	0.0646	0.0840	0.0059	**0.0007**	0.0068	0.0070
ILPD	AV	0.2453	0.2440	0.2651	0.2870	0.3115	0.2614	0.2640	0.2556	**0.2268**
	SD	0.0031	**0.0000**	0.0123	0.0542	0.0263	0.0156	0.0174	0.0113	**0.0000**
Lymphography	AV	0.0595	**0.0539**	0.1465	0.2108	0.2680	0.1439	0.4621	0.4409	0.3613
	SD	0.0162	0.0163	0.0144	0.0639	0.0694	**0.0083**	0.0418	0.0432	0.0204
Parkinsons	AV	0.0227	0.0212	0.0078	0.0313	0.0798	**0.0056**	0.0305	0.0296	0.0280
	SD	0.0084	0.0100	0.0098	0.0475	0.0467	0.0048	**0.0005**	0.0010	0.0006
ParkinsonC	AV	0.1982	**0.1880**	0.3068	0.3108	0.3191	0.3049	0.2554	0.2484	0.2476
	SD	0.0340	0.0195	0.0021	0.0116	0.0086	0.0027	0.0091	0.0009	**0.0005**
SPECT	AV	**0.1225**	0.1245	0.2656	0.3012	0.3760	0.2558	0.2453	0.2187	0.1662
	SD	0.0116	**0.0092**	0.0179	0.0438	0.0566	0.0151	0.0201	0.0203	0.0173
Cleveland	AV	0.3980	0.3891	0.3211	0.3342	0.3894	**0.3059**	0.4287	0.4243	0.4226
	SD	0.0127	0.0038	0.0100	0.0493	0.0476	0.0238	0.0078	0.0015	**0.0008**
HeartEW	AV	0.1072	0.0966	0.1631	0.1562	0.2601	0.1457	0.1366	0.1288	**0.0411**
	SD	0.0151	0.0043	0.0295	0.0563	0.1018	0.0273	0.0279	0.0306	**0.0006**
Hepatitis	AV	0.0631	0.0534	0.0036	0.0313	0.0818	**0.0033**	0.1289	0.1270	0.0778
	SD	0.0006	0.0226	0.0007	0.0420	0.0757	0.0006	**0.0004**	0.0007	0.0268
Saheart	AV	0.2939	0.2939	0.2792	0.3068	0.3958	0.2812	0.2911	0.2709	**0.2401**
	SD	0.0002	**0.0000**	0.0093	0.0481	0.0470	0.0130	0.0177	0.0181	**0.0000**
Spectfheart	AV	**0.0867**	0.0913	0.1526	0.1969	0.2425	0.1439	0.1373	0.1321	0.0959
	SD	0.0165	0.0112	0.0150	0.0466	0.0542	0.0166	0.0263	0.0195	**0.0078**
Thyroid	AV	**0.0111**	0.0122	0.0297	0.0299	0.0493	0.0252	0.0152	0.0144	0.0144
	SD	0.0011	0.0009	0.0039	0.0168	0.0187	0.0038	0.0030	0.0015	**0.0008**
Heart	AV	0.1248	**0.1141**	0.1793	0.1791	0.2855	0.1656	0.2174	0.1974	0.1531
	SD	0.0152	0.0151	0.0229	0.0543	0.0857	0.0217	0.0362	0.0182	**0.0000**
PimaDiabetes	AV	0.2574	0.2574	0.2320	0.2526	0.3011	0.2320	0.2283	0.2058	**0.2056**
	SD	**0.0000**	**0.0000**	0.0007	0.0289	0.0665	0.0006	0.0294	0.0006	**0.0000**
Leukemia	AV	**0.0012**	0.0047	0.0101	0.0527	0.0803	0.0073	0.0771	0.0615	0.0182
	SD	0.0002	**0.0000**	0.0178	0.0429	0.0258	0.0129	0.0001	0.0316	0.0308
Colon	AV	**0.0835**	0.0870	0.2803	0.2806	0.3871	0.2388	0.1712	0.1369	0.0878
	SD	0.0001	**0.0000**	0.0498	0.0902	0.0734	0.0380	0.0001	0.0451	0.0005
ProstateGE	AV	**0.1004**	0.1038	0.1522	0.1919	0.2326	0.1501	0.1052	0.1039	0.1015
	SD	0.0002	**0.0000**	0.0091	0.0463	0.0461	0.0125	**0.0000**	**0.0000**	**0.0000**
COVID-19	AV	0.0430	**0.0427**	0.0507	0.0567	0.0849	0.0502	0.0531	0.0467	0.0443
	SD	0.0018	0.0007	0.0028	0.0231	0.0218	0.0026	0.0029	0.0029	**0.0004**

Table 13 Comparison results between the proposed BECSA and other methods based on sensitivity results

Dataset	Measure	BCSA	BECSA	BBBO	BMFO	BPSO	BTLBO	BAFT	BLSHADE	BHBA
Diagnostic	AV	0.8801	**0.8963**	0.8916	0.8728	0.7251	0.6186	0.8654	0.8900	0.8821
	SD	0.0497	0.0413	0.0522	0.1332	0.1973	0.2016	0.0352	**0.0302**	0.0425
Breast	AV	0.9877	**0.9919**	0.9680	0.9294	0.8796	0.7830	0.9574	0.9630	0.9684
	SD	**0.0035**	0.0068	0.0157	0.0982	0.1190	0.1305	0.0100	0.0074	0.0055
Prognostic	AV	0.3214	0.2714	0.1648	0.1855	0.1843	0.1024	0.8441	**0.9027**	0.8675
	SD	0.1508	0.1269	0.1457	0.1541	0.1996	**0.1079**	0.1584	0.1636	0.1244
Coimbra	AV	**0.7332**	0.7225	0.6909	0.6764	0.6275	0.5525	0.7123	0.6504	0.5610
	SD	0.1177	0.1178	0.1006	0.1669	0.2032	0.1849	0.1610	**0.0532**	0.1652
BreastEW	AV	0.9579	0.9454	0.9603	**0.9655**	0.9575	0.9625	0.9397	0.9217	0.9497
	SD	0.0238	0.0234	**0.0182**	0.0197	0.0239	0.0250	0.0215	0.0294	0.0242
Retinopathy	AV	0.6465	**0.6773**	0.6336	0.6654	0.6189	0.6315	0.5936	0.5978	0.5546
	SD	0.0469	0.0500	**0.0396**	0.0402	0.0692	0.0456	0.0412	0.0510	0.0635
Dermatology	AV	**0.9909**	0.9905	0.9828	0.9809	0.9406	0.9366	0.9517	0.9620	0.9594
	SD	0.0188	0.0209	0.0255	0.0362	0.0648	0.0563	**0.0000**	0.0203	0.0263
ILPD	AV	0.8400	**0.8447**	0.8178	0.8423	0.8272	0.8148	0.7736	0.8247	0.8083
	SD	0.0508	0.0488	0.0563	0.0593	0.0444	**0.0398**	0.0470	0.0564	0.0558
Lymphography	AV	0.6541	0.5263	0.6368	0.4850	0.4678	0.4644	0.6351	**0.7109**	0.6299
	SD	0.4712	0.5011	0.4262	0.4370	0.4217	0.4200	0.1329	**0.1093**	0.1989
Parkinsons	AV	0.9075	0.9240	0.9532	**0.9536**	0.9418	0.9397	0.9202	0.9018	0.9152
	SD	0.0578	0.0482	0.0327	0.0384	0.0514	0.0549	0.0468	**0.0304**	0.0415
ParkinsonC	AV	0.8823	**0.9007**	0.8829	0.8753	0.8941	0.8786	0.8228	0.8692	0.8325
	SD	0.0282	0.0345	0.0310	0.0265	**0.0254**	0.0405	0.0588	0.0488	0.0320
SPECT	AV	0.5953	0.5814	0.5240	0.5479	0.5370	0.5332	0.5868	**0.6298**	0.5478
	SD	0.1186	0.1094	0.1500	0.1135	0.1160	0.1149	0.0932	0.1735	**0.0550**
Cleveland	AV	0.1779	0.1735	0.1532	0.1423	0.1361	0.1076	0.6028	**0.6554**	0.6196
	SD	0.0828	0.1428	0.1304	0.1482	0.1159	0.0845	**0.0358**	0.0609	0.0524
HeartEW	AV	0.8788	0.8547	0.8115	0.8458	0.7613	0.7134	0.7699	**0.9172**	0.8189
	SD	0.0652	0.0673	0.0776	0.0789	0.1014	0.1084	0.0571	**0.0400**	0.0548
Hepatitis	AV	0.3337	0.2586	0.3428	0.4067	0.3428	0.2613	0.8155	**0.8997**	0.8795
	SD	0.3451	0.3351	0.3961	0.3533	0.3857	0.3748	0.4183	0.3419	**0.2236**
Saheart	AV	0.2964	0.2798	0.2988	0.2959	0.3094	0.3151	0.7174	**0.7461**	0.7296
	SD	0.0797	0.0761	**0.0687**	0.0740	0.0841	0.1016	0.1262	0.1126	0.1299
Spectfheart	AV	**0.8723**	0.8701	0.8686	0.8516	0.8573	0.8541	0.8622	0.8520	0.8661
	SD	**0.0499**	0.0543	0.0661	0.0640	0.0507	0.0676	0.0610	0.0650	0.0683
Thyroid	AV	0.6909	0.7042	0.7313	0.7663	0.6516	0.5339	0.9264	**0.9594**	0.9508
	SD	0.0926	0.1003	0.0987	0.1167	0.2175	0.2018	0.0490	0.0500	**0.0491**
Heart	AV	0.9280	**0.9396**	0.9161	0.9210	0.7931	0.6613	0.7336	0.7908	0.8146
	SD	0.0478	0.0462	0.0643	0.0531	0.1258	0.1071	0.0578	**0.0361**	0.0692
PimaDiabetes	AV	0.4973	0.4791	0.5713	0.5385	0.4867	0.5499	0.7224	**0.7524**	0.7467
	SD	0.0630	0.0550	0.0648	0.0760	0.1191	0.0768	0.0856	**0.0504**	0.0633
Leukemia	AV	0.7610	0.6932	0.6652	0.6947	0.7117	0.6912	0.8719	**0.9514**	0.9459
	SD	**0.1685**	0.2449	0.2172	0.1998	0.2443	0.2167	0.1404	0.2191	0.2453
Colon	AV	0.5633	0.5972	0.5695	0.6211	0.5490	0.6391	0.8284	**0.8349**	0.8284
	SD	0.2570	0.2216	0.2815	0.2605	0.2224	0.2160	0.3140	0.2033	**0.1394**
ProstateGE	AV	0.8216	0.8305	**0.8947**	0.8704	0.8365	0.8605	0.8497	0.8845	0.8793
	SD	0.1309	0.1214	0.1002	0.1351	0.1433	0.1080	**0.0806**	0.1019	0.0966
COVID-19	AV	0.5997	0.6161	0.6514	0.5871	0.4837	0.3857	0.9074	**0.9227**	0.9188
	SD	0.0928	0.1144	0.1293	0.1872	0.2492	0.2147	0.1080	**0.0241**	0.0959

proposed BECSA ranked fourth although it has the best sensitivity results in 6 datasets. BBBO and BMFO came in fifth and sixth places with the best sensitivity results only in 2 and 1 datasets, respectively. Finally, BAFT, BPSO, and BTLBO did not report any distinct sensitivity result in any of the datasets, where they were in seventh, eighth, and ninth places among all other competing algorithms. Regarding the standard deviation values of the proposed BECSA, they are small, indicating that the stability of BECSA is entrenched.

Similarly, the average and standard deviations of the specificity results for BECSA, BCSA, and all other comparative methods are summarized in Table 14.

Reading the specificity results listed in 14, one can notice that BCSA and BECSA ranked first and second, where each having exclusively the highest specificity values in a total of 4 datasets out of 24. BHBA placed third by getting the best specificity results in 3 datasets. BLSHADE, BTLBO, BAFT, and BBBO exclusively got the highest specificity values in 6, 3, 2, and 2 datasets, respectively. BMFO and BPSO did not achieved any highest specificity results in any of the datasets, but their obtained results are reasonable and better than other competing algorithms such as BTLBO and BAFT. In terms of standard deviation results, the proposed BECSA has very small SD values in most of the test datasets in comparison to other rivals. These findings confirm that the superiority of BECSA is stable.

The number of selected features is also considered to study the performance of the proposed BECSA against BCSA and other methods available in the literature as shown in Table 15.

The amount of features selected during the classification process is as important as the classification accuracy in assessing any feature selection algorithm. When comparing the proposed algorithm to the eight rival algorithms, Table 15 reveals a wide variety in the results. According to the results in the average number of features, the nine competing algorithms can be split as follows: BCSA outperformed the other rivals by exclusively acquiring the fewest number of selected features in a total of in 11 out of 24 datasets. BECSA ranked second with the exclusive minimum number of features in 4 datasets, and shared the minimum number of features for the Sahear dataset with BHBA, which exclusively received the minimum number of features in the Coimbra, SPECT, and Hepatitis datasets. BMFO had the lowest number of features in the Breast, ILPD, Lymphography, and Heart datasets, while BPSO reduced the number of features only in PimaDiabetes dataset. The BLSHADE, BAFT, BBBO, and BTLBO algorithms did not reach any minimum number of features in any dataset considered. It may be deduced from a second reading of the findings in Table 15 that the BECSA performed more robustly than its rivals because it achieved the lowest SD results in 10 out of 24 datasets. This implies that by carrying out the algorithm 30 independent times, BECSA was able to reach around the same amount of selected features.

The results shown in Tables 11, 12, 13, 14, and 15 reveal the robustness of the proposed BECSA in comparison with other state-of-the-art feature selection algorithms available in the literature. By taking a closer look at these results and observing the margin differences between BECSA and other competing algorithms, one can see that the algorithms such as BTLBO, BBBO, and BPSO lag far behind BECSA. Moreover, in regards to the standard divisions of the proposed BECSA, they are tiny and inferior to those of other competing algorithms. This confirms that the superiority of this proposed algorithm is solid. The key factors for the reasonable degree of performance of BECSA is the sought-after balance between exploration and exploitation features of this algorithm on account of the use of the proposed cognitive and social models for the velocities of capuchins as well as the proposed mathematical model of this algorithm. This mathematical model of BECSA assisted the capuchins to explore and exploit each promising area in the search space, thus reviving a sensible balance between exploitation and exploration. In this respect, if the capuchins become stuck in local optimums, they have a chance to leave their local neighborhood.

Statistical Test

For further evaluation of the proposed methods, a non-parametric Friedman's statistical test was used to highlight the algorithm that has superior results compared to other comparative algorithms. Table 16 shows the average ranking results of the statistical evaluation of BECSA, BCSA, and other comparative methods using Friedman's test based on classification accuracy, fitness value, sensitivity, specificity, and number of selected features.

As can be inferred from Table 16, the lowest ranking value reflects better performance. The p-value was calculated using Friedman's test as shown in Table 16, where all p-values were below the significance level $\alpha = 0.5$. This leads to the rejection of the null hypothesis and the acceptance of the alternative hypothesis. The null hypothesis states that all the compared algorithms have the same performance behavior when used to solve an optimization problem, while the alternative hypothesis means that there is a difference between the performance behaviors of the algorithms when they are used to solve an optimization problem. According to the statistical results presented in Table 16, BECSA is statistically significant and is the most effectual method among all other methods. In this, it can be seen that the proposed BECSA ranked first in classification accuracy, first in fitness value, and first in specificity, while BECSA ranked fourth in sensitivity and second in number of selected features.

Table 14 Comparison results between the proposed BECSA and other methods based on specificity results

Dataset	Measure	BCSA	BECSA	BBBO	BMFO	BPSO	BTLBO	BAFT	BLSHADE	BHBA
Diagnostic	AV	0.9625	**0.9703**	0.9685	0.9515	0.9279	0.9233	0.9288	0.9257	0.9361
	SD	0.0223	0.0260	0.0251	0.0453	0.0652	0.0538	0.0209	0.0259	**0.0147**
Breast	AV	**0.9988**	0.9934	0.9792	0.9070	0.8194	0.6768	0.9485	0.9649	0.9578
	SD	**0.0045**	0.0083	0.0168	0.1493	0.2046	0.2120	0.0153	0.0083	0.0078
Prognostic	AV	0.8932	**0.9207**	0.8936	0.9132	0.9144	0.9145	0.8736	0.8739	0.8879
	SD	0.0692	0.0489	0.0629	0.0613	0.0545	**0.0461**	0.0472	0.0652	0.0718
Coimbra	AV	0.7923	**0.8202**	0.7138	0.6919	0.6906	0.6575	0.7918	0.7942	0.7355
	SD	0.1066	0.0979	0.1458	0.1589	0.1546	0.1644	0.0708	0.2436	**0.0517**
BreastEW	AV	0.9220	**0.9323**	0.9309	0.9293	0.9203	0.9366	0.8967	0.8902	0.9071
	SD	0.0408	0.0457	0.0422	0.0366	0.0376	0.0436	0.0298	0.0468	**0.0104**
Retinopathy	AV	0.6828	0.6961	0.7056	0.7036	0.6521	0.6562	0.6886	0.7106	**0.7210**
	SD	0.0467	0.0495	0.0507	0.0446	0.0767	0.0466	**0.0236**	0.0241	0.0570
Dermatology	AV	0.9073	0.9399	**0.9484**	0.8861	0.8554	0.8166	0.9281	0.8982	0.8950
	SD	0.0482	0.0456	0.0327	0.0859	0.0762	0.1058	0.0272	**0.0207**	0.0489
ILPD	AV	0.3097	0.2757	0.3159	0.3018	0.3173	0.3084	0.7084	**0.7518**	0.7307
	SD	0.0991	0.0774	0.0807	0.0951	0.0781	0.0849	0.0739	0.0887	**0.0659**
Lymphography	AV	**0.8771**	0.8663	0.7888	0.7302	0.7244	0.7021	0.5795	0.6132	0.6111
	SD	0.0729	0.0711	0.0606	0.0739	0.0917	0.0863	0.1025	**0.0357**	0.1065
Parkinsons	AV	0.5785	0.5812	0.6713	0.7042	0.5804	0.5687	0.9243	**0.9353**	0.9313
	SD	0.1538	0.1931	0.1233	0.1438	0.1764	0.1812	0.2560	**0.1006**	0.1495
ParkinsonC	AV	0.3262	0.4038	0.2928	0.2861	0.2457	0.2878	0.7138	**0.7569**	0.7399
	SD	0.1219	0.0847	0.0729	0.0710	0.0697	0.0734	0.0829	0.0819	**0.0455**
SPECT	AV	0.7408	0.7537	0.7533	0.7093	0.7328	0.7322	0.7794	**0.8036**	0.7645
	SD	0.1183	0.0983	0.1296	0.1181	0.1362	0.1270	**0.0494**	0.0186	0.1047
Cleveland	AV	0.6516	**0.6642**	0.6235	0.6421	0.5891	0.5693	0.6164	0.6433	0.6258
	SD	0.0587	0.0715	0.0841	0.0575	0.0944	0.0685	**0.0307**	0.0722	0.0508
HeartEW	AV	0.7139	0.7481	0.7012	0.7096	0.6334	0.5675	0.7831	**0.8037**	0.7867
	SD	0.1234	0.0964	0.0847	0.1002	0.1465	0.0897	0.0797	**0.0764**	0.0847
Hepatitis	AV	**0.9583**	0.9535	0.9582	0.9060	0.9168	0.9039	0.7960	0.7480	0.7932
	SD	0.0529	0.1029	0.0497	0.1989	0.1876	0.2496	0.0654	0.2000	**0.0000**
Saheart	AV	0.7623	0.7833	0.8285	0.8205	0.7908	0.7817	**0.9269**	0.9243	0.9126
	SD	0.0516	**0.0471**	0.0595	0.0576	0.0612	0.0527	0.0747	0.0579	0.0544
Spectfheart	AV	0.4145	0.4380	0.3894	0.4133	0.3614	0.3972	0.7921	0.8238	**0.8448**
	SD	0.1639	0.1685	0.1665	0.1698	**0.1359**	0.1965	0.1433	0.2457	0.1828
Thyroid	AV	**0.9876**	0.9872	0.9821	0.9769	0.9614	0.9493	0.9513	0.9617	0.9621
	SD	0.0031	0.0036	0.0049	0.0170	0.0188	0.0126	0.0054	0.0055	**0.0028**
Heart	AV	0.6416	0.6238	0.6317	0.5990	0.5956	0.5728	**0.7662**	0.7615	0.7524
	SD	0.1307	0.1143	0.1095	0.1788	0.1342	0.0971	0.2024	**0.0612**	0.1418
PimaDiabetes	AV	0.8008	0.8203	0.8223	**0.8233**	0.8015	0.8087	0.7843	0.8148	0.7796
	SD	0.0318	**0.0314**	0.0456	0.0326	0.0477	0.0489	0.0824	0.0526	0.0431
Leukemia	AV	0.9735	0.9782	0.9776	0.9661	0.9459	**0.9880**	0.9546	0.9665	0.9653
	SD	0.0449	0.0452	0.0458	0.0491	0.0736	0.0368	0.0373	0.0994	**0.0000**
Colon	AV	0.8640	0.8328	0.8553	0.8610	0.8504	0.8956	0.8657	**0.9168**	0.8478
	SD	0.1201	0.1669	0.1472	0.1281	0.1323	0.1210	0.1087	**0.0786**	0.1337
ProstateGE	AV	0.8198	0.8487	0.8273	0.8556	0.8287	0.8385	0.8659	0.8643	**0.8683**
	SD	0.1264	0.1178	**0.0966**	0.0991	0.1323	0.1061	0.1251	0.1911	0.1218
COVID-19	AV	0.9794	0.9788	0.9754	0.9788	0.9788	**0.9827**	0.9425	0.9542	0.9484
	SD	**0.0108**	0.0109	0.0124	0.0113	0.0143	0.0117	0.0134	0.0135	0.0144

Table 15 Comparison results between the proposed BECSA and other methods based on the average number of selected features

Dataset	Measure	BCSA	BECSA	BBBO	BMFO	BPSO	BTLBO	BAFT	BLSHADE	BHBA
Diagnostic	AV	**5.30**	8.47	15.70	11.03	11.00	19.43	17.40	17.40	13.00
	SD	3.42	1.74	3.80	1.61	2.42	2.93	3.21	2.51	**1.22**
Breast	AV	5.00	5.00	4.07	**3.17**	4.17	6.23	4.20	4.80	3.40
	SD	**0.00**	**0.00**	0.78	0.91	1.18	1.52	1.31	1.30	0.55
Prognostic	AV	**9.47**	11.67	17.17	14.53	13.90	21.63	18.40	16.00	12.80
	SD	4.01	1.65	2.96	2.21	2.80	2.13	2.88	**1.41**	1.92
Coimbra	AV	4.27	4.00	4.27	3.67	4.37	5.77	6.40	6.80	**3.60**
	SD	0.45	**0.00**	0.74	0.76	1.27	1.41	0.89	0.84	1.34
BreastEW	AV	13.43	**12.20**	18.67	16.53	15.27	18.83	18.60	16.40	16.40
	SD	3.35	**1.30**	2.26	2.24	2.48	2.25	1.82	1.67	2.61
Retinopathy	AV	7.80	**7.50**	10.83	7.73	8.03	12.77	10.00	9.80	9.40
	SD	2.44	2.22	2.60	1.31	2.14	1.94	2.65	1.30	**1.14**
Dermatology	AV	**14.50**	16.07	19.87	15.33	16.07	21.83	24.40	20.6	17.20
	SD	4.51	3.19	2.79	**1.75**	2.77	2.49	2.30	2.51	3.96
ILPD	AV	3.37	3.00	4.87	**2.93**	4.47	6.10	4	4.20	3.00
	SD	0.67	**0.00**	1.43	0.78	1.07	1.69	1.41	1.30	**0.00**
Lymphography	AV	8.73	8.93	9.73	**6.60**	7.17	11.43	10.20	8.40	9.20
	SD	1.86	1.96	1.72	1.35	2.07	1.79	2.28	**0.55**	1.30
Parkinsons	AV	**1.43**	3.90	11.67	7.50	8.00	13.87	11.20	9.20	5.80
	SD	**1.04**	2.23	2.43	1.36	1.98	1.93	1.10	2.17	1.30
ParkinsonC	AV	**177.47**	367.17	409.03	413.43	365.07	490.30	471.20	439.80	379.00
	SD	64.22	15.01	57.48	14.45	14.21	14.24	**13.86**	64.52	34.38
SPECT	AV	9.30	9.47	13.07	10.70	10.80	13.67	13.60	12.60	**9.20**
	SD	2.51	1.76	**1.46**	2.35	2.52	2.71	1.95	2.30	2.28
Cleveland	AV	4.73	**3.33**	6.67	4.73	5.90	8.90	7.60	6.20	4.00
	SD	2.30	**0.96**	1.75	1.14	1.58	1.47	1.52	1.92	1.00
HeartEW	AV	**4.30**	4.87	5.53	5.23	5.30	8.17	6.00	5.40	5.80
	SD	1.12	**0.51**	1.41	1.52	1.49	2.05	9.80	6.20	6.80
Hepatitis	AV	2.30	3.50	6.87	4.60	6.47	12.27	1.73	0.89	**0.84**
	SD	1.06	1.68	1.33	1.00	2.75	2.53	**0.84**	1.30	1.64
Saheart	AV	3.03	**3.00**	5.90	4.63	4.70	5.80	4.40	3.60	**3.00**
	SD	0.18	**0.00**	0.71	0.72	1.18	1.06	0.89	1.82	**0.00**
Spectfheart	AV	19.80	**18.27**	24.80	22.70	20.67	28.73	29.00	22.40	27.40
	SD	5.54	2.92	4.15	3.21	2.40	3.06	**1.41**	7.70	2.79
Thyroid	AV	**4.87**	6.47	10.53	8.10	8.27	13.37	12.40	10.20	8.40
	SD	0.57	1.11	1.96	1.60	2.05	1.71	1.95	1.79	**0.55**
Heart	AV	4.90	5.33	5.70	**4.03**	5.73	8.57	8.00	7.80	6.00
	SD	1.09	0.96	1.49	0.96	1.36	1.77	1.58	3.27	**0.00**
PimaDiabetes	AV	4.00	4.00	4.40	3.63	**3.60**	5.03	5.60	4.20	4.00
	SD	**0.00**	**0.00**	0.56	0.96	1.07	1.40	0.89	0.45	**0.00**
Leukemia	AV	**878.53**	3347.43	3789.33	3898.57	3462.80	4588.53	4487.20	3500.00	2899.80
	SD	170.66	**13.62**	487.92	35.00	40.27	52.02	49.09	17.68	1157.29
Colon	AV	**206.10**	902.07	1064.47	1119.33	971.37	1300.20	1249.40	985.00	1051.80
	SD	27.16	**6.50**	140.25	26.95	20.16	20.23	11.72	23.72	98.62
ProstateGE	AV	**841.50**	2842.37	3195.90	3267.93	2935.20	3889.83	3728.00	2904.40	1512.20
	SD	106.92	10.29	390.89	59.76	33.83	26.71	19.66	**7.37**	23.73
COVID-19	AV	**3.30**	3.43	7.60	5.23	6.30	9.07	8.20	7.40	5.60
	SD	0.79	0.94	1.28	1.01	1.47	2.10	0.84	1.14	**0.55**

Table 16 Average rankings of all competitor algorithms using Friedman's test

Algorithm	Accuracy	Fitness	Sensitivity	Specificity	Features
BCSA	2.8750	2.9791	4.2916	4.5000	2.3541
BECSA	2.3541	2.5625	4.4166	3.7083	2.7083
BBBO	4.4374	5.3125	4.6458	4.4583	6.5208
BMFO	5.8124	6.7499	4.9583	5.2916	3.8125
BPSO	7.7083	8.5833	6.5208	6.8333	4.2291
BTLBO	8.5000	4.2916	7.1249	6.5000	8.7500
BAFT	5.8125	6.4166	5.1041	5.0833	7.3958
BLSHADE	4.4583	4.9166	3.5416	3.9166	5.5416
BHBA	3.0416	3.1875	4.3958	4.7083	3.6875
p-value	7.2019E-11	4.7321E-11	6.2589E-05	2.7460E-04	9.7105E-11

Furthermore, BCSA ranked first in number of selected features, second in classification accuracy, second in fitness, second in sensitivity, and fourth in sensitivity. These outcomes denote the robustness of the proposed BECSA and BCSA among other rivals. Finally, it is clear that BLSHADE obtained the first rank in respect of the sensitivity, where it got a rank of 3.5416 which is the lowest of all other ranks that the other algorithms have.

Holm's test was then utilized as *post-hoc* approach to show the significant difference between the control algorithm and the other competitors. In view of this, the algorithm with the first rank in each assessment measure using Friedman's test is the control algorithm. The statistical results got by Holm's procedure are presented in Table 17. In Table 17, R_0 is the Friedman's rank assigned to the control algorithm, R^i is the Friedman's rank assigned to algorithm i, ES is the effect size of the control method on method i, and z represents the statistical difference between two methods.

A comparison of BECSA with other FS methods was conducted by applying Holm's test as presented in Table 17, where this test discards hypotheses with p-values ≤ 0.02500, ≤ 0.01666, ≤ 0.00833, ≤ 0.00833, and ≤ 0.01250 in classification accuracy, fitness value, sensitivity, specificity, and number of selected features, respectively. Reading the results given in Table 17, one can conclude that there is a significant difference between BECSA and BBBO, BLSHADE, BMFO, BAFT, BPSO, and BTLBO in terms of classification accuracy, while there is no significant difference between BECSA and the remaining two algorithms (i.e., BCSA and BHBA). As per the statistical fitness results computed according to Friedman's and Holm's test, there is no significant difference between BECSA and two other competing algorithms (i.e., BCSA and BHBA). However, there is a notable difference between BECSA and the other remaining algorithms (i.e., BTLBO, BLSHADE, BBBO, BAFT, BMFO, and BPSO). According to the specificity results, there is a significant difference between BECSA and three other algorithms (i.e., BMFO, BTLBO, and BPSO), while there is no significant difference between BECSA and the other competing algorithms including BLSHADE, BBBO, BCSA, BHBA, and BAFT. On the other hand, regarding the sensitivity results, there is no significant difference between BECSA and the control algorithm (i.e., BLSHADE). Similarly, in terms of the number of selected features, there is not much difference between BECSA and the control algorithm (i.e., BCSA). As can be realized from the results in Tables 16 and 17, BECSA and BCSA are effective FS method in getting promising results for the datasets under study, and they are much better than the other competing methods.

One important conclusion drawn from the statistical analysis results discussed above is that, on average, BECSA surpassed other state-of-the-art FS techniques mentioned in the literature, including BLSHADE, BHBA, BPSO, and BMFO. This highlights the strong performance of BECSA and shows that this algorithm can effectively explore the search space whether there are a single or many optimums present, or if the feature selection problems are low, medium, or high dimensional. Additionally, the average ranking of the algorithms in terms of sensitivity results divulges that the performance score of BECSA is not far behind that of BLSHADE and BHBA, whereas the performance of BECSA, BLSHADE, and BHBA is far from all other rivals such as BPSO and BMFO. Specifically, we may infer that the excellent superiority of BECSA in addressing feature selection problems is due to its thoughtful mathematical model. In conclusion, the results of this statistical study show that BECSA is a good and trustworthy method with reasonable exploration and exploitation aspects. These conclusions offer positive reasons to utilize the proposed method to address more challenging real-world applications in the field of healthcare.

Table 17 Holm's test results between the control algorithm and all other comparative methods

i	Algorithm	$z = \frac{(R_0 - R^i)}{SE}$	p-value	$\alpha \div i$	Hypothesis
Classification accuracy (BECSA is the control algorithm)					
8	BTLBO	7.7739	7.60E-15	0.0062	Reject
7	BPSO	6.7725	1.26E-11	0.0071	Reject
6	BAFT	4.3744	1.21E-05	0.0083	Reject
5	BMFO	4.3744	1.21E-05	0.0100	Reject
4	BLSHADE	2.6615	0.0078	0.0125	Reject
3	BBBO	2.6352	0.0084	0.0167	Reject
2	BHBA	0.8696	0.3845	0.0250	Not reject
1	BCSA	0.6588	0.5100	0.0500	Not reject
Fitness (BECSA is the control algorithm)					
8	BPSO	7.6158	2.62E-14	0.0062	Reject
7	BMFO	5.2968	1.18E-07	0.0071	Reject
6	BAFT	4.8752	1.09E-06	0.0083	Reject
5	BBBO	3.4785	5.04E-04	0.0100	Reject
4	BLSHADE	2.9778	0.0029	0.0125	Reject
3	BTLBO	2.1872	0.0287	0.0167	Reject
2	BHBA	0.7906	0.4292	0.0250	Not reject
1	BCSA	0.5270	0.5982	0.0500	Not reject
Sensitivity (BLSHADE is the control algorithm)					
8	BTLBO	4.5326	5.83E-06	0.0062	Reject
7	BPSO	3.7684	1.64E-04	0.0071	Reject
6	BAFT	1.9764	0.0481	0.0083	Reject
5	BMFO	1.7919	0.0731	0.0100	Not reject
4	BBBO	1.3967	0.1625	0.0125	Not reject
3	BECSA	1.1068	0.2684	0.0167	Not reject
2	BHBA	1.0804	0.2799	0.0250	Not reject
1	BCSA	0.9487	0.3428	0.0500	Not reject
Specificity (BECSA is the control algorithm)					
8	BPSO	3.9528	7.72E-05	0.0062	Reject
7	BTLBO	3.5312	4.14E-04	0.0071	Reject
6	BMFO	2.0028	0.0452	0.0083	Reject
5	BAFT	1.7393	0.0819	0.0100	Not reject
4	BHBA	1.2649	0.2059	0.0125	Not reject
3	BCSA	1.0014	0.3166	0.0167	Not reject
2	BBBO	0.9487	0.3428	0.0250	Not reject
1	BLSHADE	0.26352	0.7921	0.0500	Not reject
Features (BCSA is the control algorithm)					
8	BTLBO	8.0901	5.96E-16	0.0062	Reject
7	BAFT	6.3773	1.80E-10	0.0071	Reject
6	BBBO	5.2705	1.36E-07	0.0083	Reject
5	BLSHADE	4.0319	5.53E-05	0.0100	Reject
4	BPSO	2.3717	0.0177	0.0125	Reject
3	BMFO	1.8447	0.0651	0.0167	Not reject
2	BHBA	1.6865	0.0917	0.0250	Not reject
1	BECSA	0.4479	0.6542	0.0500	Not reject

Conclusion and Future Works

In this paper, three enhanced binary cognitive computation methods based on capuchin search algorithm (CSA) are proposed for feature selection (FS) problems in medical diagnostic applications. These methods are referred to as binary exponential CSA (BECSA), binary power CSA (BPCSA), and binary S-shaped CSA (BSCSA). Each version utilizes a different growth function to update the values of the cognitive and social parameters during the iterative process. The goals of these FS algorithms include creating simple and comprehensive models, enhancing data-mining performance, and helping prepare clear and non-redundant data. In the meantime, these proposed methods could be successfully used to reduce the dimensionality of data for machine learning tasks. The performance of these methods was assessed on 24 datasets using several assessment criteria. Initially, the results produced by the three proposed versions of CSA are compared together in addition to those produced by the native version of the binary CSA. For comparative evaluations, the proposed BECSA and basic binary CSA are compared with other well-established algorithms. Evaluation based on Friedman's and Holm's tests showed that BECSA is able to rank first in terms of classification accuracy, fitness value, and specificity. As the proposed binary versions of CSA revealed attractive performance in handling FS problems, further extensions of these versions could be made for future research. For example, these methods might potentially be used by researches working on multi-objective optimization problems. Gene selection, as a high-dimensional dataset, could also be used to further validate the suitability of these methods. Other transfer functions such as U-shape, V-shape, and X-shape could be used to check the effect of these transfer functions on the performance of the proposed methods. Due to the differences in accuracy between the classes since some of them come from different datasets. This can help convolutional neural networks (CNN) enrich the features according to the properties of each dataset, even with the ensemble methods' presence.

Data Availability The datasets in the current study are available in the UCI, KEEL, and Kaggle repositories and at https://jundongl.github.io/scikit-feature/datasets.html.

Declarations

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed Consent None.

Conflict of Interest The authors declare no competing interests.

References

1. Braik M. Enhanced ali baba and the forty thieves algorithm for feature selection. Neural Comput Applic. 2022:1–32.
2. Arora S, Anand P. Binary butterfly optimization approaches for feature selection. Expert Syst Appl. 2019;116:147–60.
3. Malik PK, Sharma R, Singh R, Gehlot A, Satapathy SC, Alnumay WS, Pelusi D, Ghosh U, Nayak J. Industrial internet of things and its applications in industry 4.0: State of the art. Comput Commun. 2021;166:125–39.
4. Awadallah MA, Al-Betar MA, Braik MS, Hammouri A, Doush IA, Zitar RA. An enhanced binary rat swarm optimizer based on local-best concepts of PSO and collaborative crossover operators for feature selection. Comput Biol Med. 2022:105675.
5. Awadallah MA, Hammouri A, Al-Betar MA, Braik MS, AbdElaziz M. Binary horse herd optimization algorithm with crossover operators for feature selection. Computers Biol Med. 2022:105152.
6. Albashish D, Hammouri A, Braik M, Atwan J, Sahran S. Binary biogeography-based optimization based SVM-RFE for feature selection. Appl Soft Comput. 2021;101:107026.
7. Zhang C, Soda P, Bi J, Fan G, Almpanidis G, Garcia S, Ding W. An empirical study on the joint impact of feature selection and data resampling on imbalance classification. Appl Intell. 2022:1–13.
8. Chong J, Tjurin P, Niemelä M, Jämsä T, Farrahi V. Machine-learning models for activity class prediction: A comparative study of feature selection and classification algorithms. Gait Posture. 2021;89:45–53.
9. Mafarja M, Qasem A, Heidari AA, Aljarah I, Faris H, Mirjalili S. Efficient hybrid nature-inspired binary optimizers for feature selection. Cogn Comput. 2020;12(1):150–75.
10. Zhou R, Niu L. Feature selection of network data via $\ell_{2,p}$ regularization. Cogn Comput. 2020;12(6):1217–32.
11. Nanda Gopal V, Al-Turjman F, Kumar R, Anand L, Rajesh M. Feature selection and classification in breast cancer prediction using IoT and machine learning. Measurement. 2021;178:109442.
12. Iqra Batool and Tamim Ahmed Khan. Software fault prediction using data mining, machine learning and deep learning techniques: A systematic literature review. Comput Electr Eng. 2022;100:107886.
13. Mehmood A, Khan MA, Sharif M, Khan SA, Shaheen M, Saba T, Riaz N, Ashraf I. Prosperous human gait recognition: an end-to-end system based on pre-trained CNN features selection. Multimed Tools Appl. 2020:1–21.
14. Cai W, Wei Z. Remote sensing image classification based on a cross-attention mechanism and graph convolution. IEEE Geosci Remote Sens Lett. 2020.
15. Raj DM, Mohanasundaram R. An efficient filter-based feature selection model to identify significant features from high-dimensional microarray data. Arab J Sci Eng. 2020;45(4):2619–30.
16. Iwendi C, Bashir AK, Peshkar A, Sujatha R, Chatterjee JM, Pasupuleti S, Mishra R, Pillai S, Jo O. COVID-19 patient health prediction using boosted random forest algorithm. Front Public Health. 2020;8:357.
17. Bhosale YH, Singh P, Sridhar Patnaik K. COVID-19 and associated lung disease classification using deep learning. In International Conference on Innovative Computing and Communications: Proceedings of ICICC 2022. 2022;3:283–95. Springer.
18. Bhosale YH, Sridhar Patnaik K. Application of deep learning techniques in diagnosis of COVID-19 (coronavirus): a systematic review. Neural Process Lett. 2022:1–53.
19. Bhosale YH, Sridhar Patnaik K. Puldi-covid: Chronic obstructive pulmonary (lung) diseases with COVID-19 classification using ensemble deep convolutional neural network from chest x-ray images to minimize severity and mortality rates. Biomed Signal Process Control. 2023;81:104445.
20. Singh D, Mathioudakis AG, Higham A. Chronic obstructive pulmonary disease and COVID-19: interrelationships. Curr Opin Pulm Med. 2022;28(2):76.
21. Renuka Devi D, Sasikala S. Online feature selection (OFS) with accelerated bat algorithm (ABA) and ensemble incremental deep multiple layer perceptron (EIDMLP) for big data streams. J Big Data. 2019;6(1):1–20.
22. Chen R-C, Dewi C, Huang S-W, Caraka RE. Selecting critical features for data classification based on machine learning methods. J Big Data. 2020;7(1):1–26.
23. Hammami M, Bechikh S, Hung C-C, BenSaid L. A multi-objective hybrid filter-wrapper evolutionary approach for feature selection. Memetic Computing. 2019;11(2):193–208.
24. Messaoudi I, Kamel N. A multi-objective bat algorithm for community detection on dynamic social networks. Appl Intell. 2019;49(6):2119–36.
25. Abdollahzadeh B, Gharehchopogh FS. A multi-objective optimization algorithm for feature selection problems. Eng Comput. 2022;38(3):1845–63.
26. Yanyu H, Zhao L, Li Z, Dong X, Tiantian X, Zhao Y. Classifying the multi-omics data of gastric cancer using a deep feature selection method. Expert Syst Appl. 2022;200:116813.
27. Braik M, Sheta A, Al-Hiary H. A novel meta-heuristic search algorithm for solving optimization problems: capuchin search algorithm. Neural Comput Appl. 2021;33(7):2515–47.
28. Braik M. A hybrid multi-gene genetic programming with capuchin search algorithm for modeling a nonlinear challenge problem: Modeling industrial winding process, case study. Neural Process Lett. 2021;53(4):2873–916.
29. Ramu S, Ranganathan R, Ramamoorthy R. Capuchin search algorithm based task scheduling in cloud computing environment. Yanbu J Eng Sci. 2022;19(1):18–29.
30. Song X-F, Zhang Y, Gong D-W, Gao X-Z. A fast hybrid feature selection based on correlation-guided clustering and particle swarm optimization for high-dimensional data. IEEE Trans Cybernetics. 2021.
31. Zhang F, Mei Y, Nguyen S, Zhang M. Evolving scheduling heuristics via genetic programming with feature selection in dynamic flexible job-shop scheduling. IEEE Trans Cybernetics. 2020;51(4):1797–811.
32. Zhang Y, Gong D-W, Gao X-Z, Tian T, Sun X-Y. Binary differential evolution with self-learning for multi-objective feature selection. Inform Sci. 2020;507:67–85.
33. Ahn G, Hur S. Efficient genetic algorithm for feature selection for early time series classification. Comput Ind Eng. 2020;142:106345.
34. Awadallah MA, Al-Betar MA, Hammouri A, Alomari OA. Binary JAYA algorithm with adaptive mutation for feature selection. Arab J Sci Eng. 2020;45(12):10875–90.
35. Kirkpatrick S. C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing science. 1983;220(4598):671–80.
36. Jeong IS, Kim HK, Kim TH, Lee DH, Kim KJ, Kang SH. A feature selection approach based on simulated annealing for detecting various denial of service attacks. Softw Netw. 2018;2018(1):173–90.
37. Mafarja MM, Mirjalili S. Hybrid whale optimization algorithm with simulated annealing for feature selection. Neurocomputing. 2017;260:302–312.
38. Yan C, Ma J, Luo H, Patel A. Hybrid binary coral reefs optimization algorithm with simulated annealing for feature selection in high-dimensional biomedical datasets. Chemom Intell Lab Syst. 2019;184:102–11.
39. Jia H, Li J, Song W, Peng X, Lang C, Li Y. Spotted hyena optimization algorithm with simulated annealing for feature selection. IEEE Access. 2019;7:71943–62.

40. Bindu MG, Sabu MK. A hybrid feature selection approach using artificial bee colony and genetic algorithm. In 2020 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA). 2020;211–6. IEEE.
41. Abdel-Basset M, Ding W, El-Shahat D. A hybrid harris hawks optimization algorithm with simulated annealing for feature selection. Artif Intell Rev. 2021;54(1):593–637.
42. Hancer E. Differential evolution for feature selection: a fuzzy wrapper-filter approach. Soft Comput. 2019;23(13):5233–48.
43. Jiang Y, Liu X, Yan G, Xiao J. Modified binary cuckoo search for feature selection: a hybrid filter-wrapper approach. In 2017 13th International Conference on Computational Intelligence and Security (CIS). 2017:488–91. IEEE.
44. Lai C-M, Yeh W-C, Chang C-Y. Gene selection using information gain and improved simplified swarm optimization. Neurocomputing. 2016;218:331–8.
45. Ke L, Li M, Wang L, Deng S, Ye J, Yu X. Improved swarm-optimization-based filter-wrapper gene selection from microarray data for gene expression tumor classification. Pattern Anal Appl. 2022:1–18.
46. Liu W, Wang J. 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). 2019;424–9. IEEE.
47. Musa JD. A theory of software reliability and its application. IEEE Trans Softw Eng. 1975;(3):312–27.
48. Sheta A. Reliability growth modeling for software fault detection using particle swarm optimization. In 2006 IEEE International Conference on Evolutionary Computation. 2006:3071–8. IEEE.
49. Crow LH. Reliability analysis for complex repairable systems, soc. industrial and applied mathematics, reliability and biometry. Proceedings of Statistical Analysis of Life Length. 1974;25:248–53.
50. Yamada S, Ohba M, Osaki S. S-shaped software reliability growth models and their applications. IEEE Trans Reliab. 1984;33(4):289–92.
51. Mirjalili S, Lewis A. S-shaped versus v-shaped transfer functions for binary particle swarm optimization. Swarm Evol Comput. 2013;9:1–14.
52. Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. Am Stat. 1992;46(3):175–85.
53. Jin Liu Y, Sheng WL, Guo R, Wang Y, Wang J. Improved asd classification using dynamic functional connectivity and multi-task feature selection. Pattern Recogn Lett. 2020;138:82–7.
54. Bhosale YH, Sridhar Patnaik K. IoT deployable lightweight deep learning application for COVID-19 detection with lung diseases using raspberrypi. In 2022 International Conference on IoT and Blockchain Technology (ICIBT). 2022;1–6. IEEE.
55. Bhosale YH, Zanwar S, Ahmed Z, Nakrani M, Bhuyar D, Shinde U. Deep convolutional neural network based COVID-19 classification from radiology x-ray images for iot enabled devices. In 2022 8th International Conference on Advanced Computing and Communication Systems. 2022;1:398–1402. IEEE.
56. Simon D. Biogeography-based optimization. IEEE Trans Evol Comput. 2008;12(6):702–13.
57. Mirjalili S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowl-Based Syst. 2015;89:228–49.
58. Venkata Rao R, Savsani VJ, Vakharia DP. Teaching–learning-based optimization: an optimization method for continuous nonlinear large scale problems. Inform Sci. 2012;183(1):1–15.
59. Viktorin A, Pluhacek M, Senkerik R. Success-history based adaptive differential evolution algorithm with multi-chaotic framework for parent selection performance on CEC2014 benchmark set. In 2016 IEEE Congress on Evolutionary Computation (CEC). 2016:4797–803. IEEE.
60. Kennedy J, Eberhart R. Particle swarm optimization. In Proceedings of ICNN'95-International Conference on Neural Networks. 1995;4:1942–8. IEEE.
61. New metaheuristic algorithm for solving optimization problems. Fatma A Hashim, Essam H Houssein, Kashif Hussain, Mai S Mabrouk, and Walid Al-Atabany. Honey badger algorithm. Math Comput Simul. 2022;192:84–110.