Soft Computing (2021) 25:2911–2933
https://doi.org/10.1007/s00500-020-05349-x

METHODOLOGIES AND APPLICATION

Feature selection by using chaotic cuckoo optimization algorithm with levy flight, opposition-based learning and disruption operator

Mahsa kelidari[1] · Javad Hamidzadeh[1]

Published online: 9 October 2020

Abstract

Feature selection, which plays an important role in high-dimensional data analysis, is drawing increasing attention recently. Finding the most relevant and important features for classifications are one of the most important tasks of data mining and machine learning, since all of the datasets have irrelevant features that affect accuracy rate and slow down the classifier. Feature selection is an optimization process, which improves the accuracy rate of data classification and reduces the number of selected features. Applying too many features both requires a large memory capacity and leads to a slow execution speed. Feature selection algorithms are often responsible to decide which features should be selected to be used during a classification algorithm. Traditional algorithms seemed to be inefficient due to the complexity of dimensions of the problem, thus evolutionary algorithms were used to improve the problem solving process. The algorithm proposed in this paper, chaotic cuckoo optimization algorithm with levy flight, disruption operator and opposition-based learning (CCOALFDO), is applied to select the optimal feature subspace for classification. It reduces the randomization in selecting features and avoids getting stuck in local optimum solutions which lead to a more interesting feature subset. Extensive experiments are conducted on 20 high-dimensional datasets to demonstrate the effectiveness and efficiency of the proposed method. The results showed the superiority of the proposed method to state-of-the-art methods in terms of classification accuracy rate. In addition, they prove the ability of the CCOALFDO in selecting the most relevant features for classification tasks. Thus, it is a reasonable solution in handling noise and avoiding serious negative impacts on the classification accuracy rate in real world datasets.

Keywords Feature selection · High-dimensional data · Cuckoo optimization algorithm · Chaotic theory · Levy flight · Disruption operator · Opposition-based learning

1 Introduction

Feature selection in high dimensional data is a preprocessing step that has been widely used to improve the performance of learning algorithms in many fields (Mafarja et al. 2019). This preprocessing technique selects relevant features, removes other features, decreases the time required to learn the predictors and finally leads to an easier interpretation by simplifying the models (Thaher et al. 2020; Yan et al. 2019).

Nowadays, feature selection is an important task in machine learning, since it can affect the performance of high-dimensional data classification. There are different sources of data applicable to solve various problems of present world. Yet, selecting features from these big and complex data sources are one of the most challenging discussions. The real-world data has a continuous form; therefore, we have to extract useful knowledge from this data to solve problems. The large quantities of high dimensional data creates great opportunities and new challenges in the conventional data mining and knowledge discovery researches (Kriegel et al. 2009). The most important challenges in the high-dimensional data analysis

Communicated by V. Loia.

✉ Javad Hamidzadeh
J_Hamidzadeh@sadjad.ac.ir

Mahsa kelidari
Ma.Kelidari117@sadjad.ac.ir

[1] Faculty of Computer Engineering and Information Technology, Sadjad University of Technology, Mashhad, Iran

include limitations in time, memory, and computational cost (Rao et al. 2019).

The data are comprised of samples with several features. Thus, selecting some samples or features is advisable to reduce its volume for better computation (Hamidzadeh et al. 2015). This reduction can be done through extracting or selecting features (Huang et al. 2018). However, selecting a feature would be advantageous, since it simplifies the interpretation of the classification model (Ramírez-Gallego et al. 2017). In other words, the complexity of classifiers decreases through using proper set of selected features. Furthermore, it causes faster convergence among optimum results, which leads to an improved performance of classifiers (Li et al. 2017; da Silva et al. 2017).

Many features of problem solving, may be useless or lack information. Some others even mislead the performance of the classifier and affect the accuracy rate due to the presence of noise. Although failing to remove these features does not cause any information problems, it raises the computational cost and increases the response time of the system. Considering the storage space, it also causes storing a lot of non-useful information along with useful data (Ramírez-Gallego et al. 2017; Lee and Kim 2016). Moreover, the specific feature selection method and the specific problem may be interrelated, so the method would not be applicable to other problems (Fong et al. 2013).

Mathematical algorithms are not interesting anymore because they are too complex and time consuming. Instead, metaheuristic algorithms which can imitate the life of living creatures, and their evolution have become more applicable (Arora and Anand 2019). Metaheuristic methods can be used to discover optimal global solutions (Nayar et al. 2019). These types of algorithms solve complex nonlinear and indeterminate problems more quickly than classical algorithms (Arora and Anand 2019). Unlike traditional algorithms, metaheuristic ones did not need to restart when new data enters or environment changes. Considering their nature, selecting features using evolutionary algorithms, reduce the consumed time and make acceptable accuracy rate in classification (Kumar and Bharti 2019). They can be applied for any problem that can be formularized, also they can be integrated with other optimization techniques. The biggest advantage of these algorithms is that they can solve problems very quickly and with reasonable accuracy rate, without applying common mathematical solutions (Mafarja et al. 2019).

There are two concepts in metaheuristic algorithms: exploration and exploitation. Exploration is a capability for searching the problem space without any attention to the results, while exploitation concentrates on results. In order to reach the best performance in problem solving, these capabilities should be balanced. Most of the evolutionary algorithms have issues with getting trapped in local optima and early convergence, thus several studies have been done on these issues (Arora and Anand 2019). By applying stringent conditions and using random numbers in implementing algorithms to increase the convergence rate, algorithms may be locked into local optimizations, and their accuracy and efficiency may reduce (Fong et al. 2013; Fong et al. 2016; Gandomi and Yang 2014).

One of the newest and most powerful evolutionary optimization methods is cuckoo's search algorithm, which is more capable of finding global optimum points. This algorithm simulates the cuckoo bird's behavior in nesting and laying to solve optimization problems (Gandomi et al. 2013). The skill of laying eggs like the host bird eggs in each generation improves evolutionary and prevents host birds from detecting them. As a result, chickens out of these eggs grow in the host nest. In this algorithm, cuckoos look for the best eggplant or globally best solution for survival (Gandomi et al. 2013; El Aziz and Hassanien 2016).

Similar to other evolutionary algorithms, cuckoo's optimization algorithm has an excessive randomness problem, so this study uses the chaotic theory, levy flight and disruption operator to limit the random behavior of this algorithm. These enhancements improve the exploration phase and feature selection in complex data. By improving the cuckoo optimization algorithm for feature selecting, a feature subset with maximum feature elimination and the highest accuracy rate in classification are obtained.

This paper discusses data preprocessing and its most important task: selecting the best feature set for next processing. The CCOALFDO algorithm is a wrapper-based feature selection method developed by an integrated framework that combines chaos theory and cuckoo optimization algorithm as well as levy flight, opposition-based learning and disruption operator to handle the 20 high-dimensional datasets in selecting the most relevant features. The K-NN classifier (where $K = 5$) is used to evaluate the produced feature subsets. Experimental results on data sets show the efficiency of the proposed method over comparative algorithms.

In summary, what you can find in this study includes:

1 Using chaotic theory, levy flight and disruption operator to reduce the random selection of features and avoid getting stuck in local optimum.
2 The opposition-based learning prevents the solutions from accumulating too much in one place and causes a better search in the problem space
3 The disruption operator searches the opposite points of solutions, so the answers cover more parts of the problem space for searching and get better results.
4 Selecting informative features efficiently and reaching convergence in a reasonable time.

5 The CCOALFDO method for feature selection problem is defined as a supervised feature selection method which maximizes the classification accuracy rate and minimizes the feature set size.

6 In comparison with the five state-of-the-art evolutionary algorithms, the CCOALFDO algorithm ranked first in the accuracy rate and ranked second in reducing the number of features.

In Sect. 2, the related literature of feature selection is reviewed. The proposed method is explained thoroughly in Sect. 3. Section 4 reports the experimental results and analyses the performances of different feature selection algorithms. Finally, Sect. 5 contains conclusions and future works.

2 Related works

Feature selection is a very important preprocessing and knowledge discovery tool for data analysis, and it is applied to many domains. In addition, it is used as an important technique for high-dimensional data analysis, since it has shown its promising ability to enhance a variety of other learning tasks, such as data clustering, document analysis, image processing and video processing, to name a few (Huda et al. 2016; Pes et al. 2017; Hossain et al. 2016; Yadav et al. 2018; Zhang et al. 2019; Luo et al. 2018; Bannigidad and Gudada 2019).

Feature selection problems, used to face a tradeoff between the feature set size and the classification quality. In fact, the size of the subspace of features usually remained fixed during previous studies. This means that if we have a dataset with k features, in order to select a subspace of that with s features, there are $k!/((k-s)!s!)$ different ways (Nayar et al. 2019; Mafarja et al. 2019). Selecting s, the size of the subspace of features, is a great challenge because when s is too big, there may be redundant subset of features and if it is too small, some useful information will be lost (Bostani and Sheikhan 2017).

Depending on the label information usage in various datasets, feature selection methods can be classified into three categories (Solorio-Fernández et al. 2020): supervised, semi-supervised and unsupervised.

The supervised learning methods use the label information of data to guide the process of dimensionality reduction. According to some criteria, these methods measure the importance and relevance of the features by utilizing the labeled data to train the feature selection models (Zare et al. 2019; Zhong 2020; Ang et al. 2015). Since unlabeled data consists of samples and features without any information about the natural grouping of data, unsupervised methods are suitable for selecting features without attention to the labels in dataset (Solorio-Fernández et al. 2020).

While the second method, a semi-supervised feature selection, integrates a small amount of labeled data into unlabeled data, the third method lacks this additional information (Ang et al. 2015; Song et al. 2016; Xu et al. 2018; Yang et al. 2018).

Feature selection algorithms are divided into three groups based on the computational time: filters, wrappers and hybrid or embedded (Ramírez-Gallego et al. 2017; Fong et al. 2013).

In filter algorithms, the learner model doesn't have any bias and features and patterns are independent, so there is no need to evolve the patterns as the features become evolved. Also, the structure of this algorithms have simple and clear criteria for feature evaluation (Li et al. 2017). Although the results of wrapper based algorithms using machine learning techniques are more reliable, and their computational cost is so high that they need to use a special method to reduce it (Kumar and Bharti 2019). Comparing with a filter algorithm, a wrapper-based feature selection has a more complex structure and higher accuracy rate, since it works in conjunction (Ramírez-Gallego et al. 2017; Fong et al. 2013; Peng and Fan 2017). To be flexible in solving problems, hybrid algorithms combine two types of algorithms including filter and wrapper ones which enables them to provide a more favorable result (Thaher et al. 2020).

Features are divided into four groups: irrelevant features, redundant features, weakly relevant but non-redundant features and strongly relevant features (Li et al. 2017). In order to reach an optimal feature set, we need algorithms that can choose some strongly relevant and non-redundant features.

Some redundant features may have significant statistical relations with other features. In other words, an irrelevant feature may be highly relevant when combined with other features. That is why feature set in some studies with computed minimal redundancy and maximum relevance (mRMR), is determined based on mutual dissimilarity with other features (Ding and Peng 2005; Masud 2010).

In past decades, researchers introduced a lot of evolutionary algorithms, such as PSO algorithm (Fong et al. 2016; Mistry et al. 2017; Dara et al. 2017; Qi et al. 2017), ACO algorithm (Shunmugapriya and Kanmani 2017; Sivagaminathan and Ramakrishnan 2007; Sweetlin et al. 2017; Harde and Sahare 2016), bee colony algorithm (da Silva et al. 2017; Shunmugapriya and Kanmani 2017), bat algorithm (Jayabarathi et al. 2018; Cheng et al. 2016),the frog leaping algorithm (Hu et al. 2016; Dai Y 2015) and the krill herd optimization algorithm (Rodrigues D 2014; Wang et al. 2019).

Recently, meta-heuristic algorithms like ant colony optimization (ACO) (Beltramo et al. 2019), simulated annealing (SA) (Yan et al. 2019), and Henry gas solubility optimization (HGSO) (Neggaz et al. 2020), have become more popular due to their good performance in solving feature selection problems. Fong, for example, used a metaheuristic algorithm as a particle search to select features. This method is suitable for high-correlation feature spaces and it can be applied with other classification methods as a fitness function (Fong et al. 2013).

Another metaheuristic algorithm is called crow search algorithm (CSA), which purposes a v-shaped binary CSA. However, like other evolutionary algorithms, it suffers from low convergence rate and entrapments in local optima (De Souza 2018). Anter introduced a hybrid crow search algorithm in selecting feature set with combination of chaos function and fuzzy c-means objective function as a cost function (Anter and Ali 2020).

Thaher purposed an algorithm to select a suitable feature set based on a hunting process performed by hawks and rabbits and called it HHO. This algorithm is a greedy search with improvement of levy flight in continuous space which transfers to binary space (Thaher et al. 2020).

Huang proposed a distributed PSO-SVM approach, which includes the binary PSO for feature selection and standard PSO for parameter optimization of SVM (Huang and Dun 2008). Also, a genetic algorithm (GA) was proposed to reduce the number of features which try to optimize the feature subset without degrading the classification accuracy (Himabindu et al. 2019).

The whale optimization algorithm (WOA) is also a recent algorithm used in feature selection problems which was not successful in reducing features (Hussien et al. 2019). Mafarja used eight different shapes of movement to improve the WOA; the results showed that most of V-shaped transfer functions had better performance in minimize the feature number and less information loss against S-shaped transfer functions (Mafarja et al. 2019).

Multi-strategy ensemble algorithm tried to enhance the processes of feature selection problems with tune gray wolf optimization (GWO) parameters (Hussien et al. 2019). Anter applied the GWO algorithm only in medical datasets (Anter et al. 2019).

Another study expecting more improvement in accuracy rate, combined GA selection with SA to avoid getting into the local optima called GA–SA algorithm (Yan et al. 2019).

The previously mentioned studies were not successful in making a balance between the classification accuracy rate and the feature reduction rate. This paper introduces a new wrapper feature selection method, in which the irrelevant features are discarded by using cuckoo optimization algorithm (COA) with the chaotic theory, levy flight and disruption operator. So, the aim of this paper is to provide a method which is able to improve classification accuracy rate and optimum reduction of the number of features simultaneously.

3 Background

In this section, cuckoo optimization algorithm is described in Sect. 3.1, the definition of chaotic function is given in Sect. 3.2, levy flight algorithm is described in Sect. 3.3, the definition of opposition-based learning is given in Sect. 3.4 and finally in Sect. 3.5, disruption operator is briefly reviewed.

3.1 Cuckoo optimization algorithm

Choosing an optimal subspace of the dataset is critical to speed up and grade the classification especially for high volume data. In this study, the cuckoo's algorithm which was introduced by Yang in 2009 (Gandomi et al. 2013) is used for selecting features. Cuckoos are interesting birds that have a spatial strategy for aggressive reproduction. In fact, they put their eggs in the nests of other birds called hosts. The cuckoo eggs will not have any chance to grow up if the host birds find them in their nest and throw them out. However, if the host birds do not find the cuckoo's egg, it will be the cuckoo's chicken, which grows faster than host's chickens and throws them out of the nest. Thus, finding the best place with similar eggs for egg laying, is essential for cuckoo's generation to survive.

Cuckoo search algorithm rules include:

- Number of nests (habitats) are fixed.
- A cuckoo lays a number between 5 and 10 eggs each time and then choose a nest.
- Best nests will be used in the next generation.
- For N_{var} dimensional problem an array with 1 row and N_{var} column are used.

A position of habitat or nest in space is like below:

$$\mathrm{Habitat} = [x_1, x_2, \ldots, x_{\mathrm{Nvar}}] \tag{1}$$

Habitats which are more suitable for egg laying are desirable; therefore, a fitness function is applied to first evaluate every habitat then describe the profit function as follow:

$$\mathrm{fitness} = f_p(\mathrm{habitat}) = \mathrm{f_p}(x_1, x_2, \ldots, xx_{\mathrm{Nvar}}) \tag{2}$$

The space needs to be limited in the optimization algorithm, then $\mathrm{var_{hi}}$ is applied for upper bound and $\mathrm{var_{low}}$ for lower bound. Every cuckoo has an egg laying radius (ELR) which is determined by the proportion between the number of eggs of this cuckoo and the total number of eggs in bound. α is tuning the maximum value of ELR. ELR is formulated in Eq 3:

$$\mathrm{ELR} = \alpha \times \frac{\text{Number of current cuckoo's eggs}}{\text{Total number of eggs}} \times (\mathrm{var}_{\mathrm{hi}} - \mathrm{var}_{\mathrm{low}}) \tag{3}$$

In this algorithm, the first place for egg laying is selected randomly, but the next generations use the best solutions definitely.

3.2 Chaotic function

Chaos is a deterministic dynamic function that is sensitive to its initial conditions and parameters. The nature of chaos is apparently random and unpredictable; however, it also possesses an element of regularity (Alatas et al. 2009).

Chaos is a nonlinear function which can change the algorithm behavior. Chaotic maps represent a pseudo-random deterministic process, that is non-converging bounded (dos Santos and Mariani 2008).

It can solve the early convergence in evolutionary algorithms by avoiding local minimums. There are a lot of chaotic maps. Ten one-dimensional maps that can be used as initial position pattern in cuckoo optimization algorithms are introduced in Table 1 (Sayed et al. 2019; Saremi et al. 2014; Hamidzadeh and Namaei 2018; Hamidzadeh et al. 2017):

3.3 Levy flight

A levy flight random walk is suggested as a permutation to perform a local search (Emary and Zawbaa 2018). Levy flight process represents the optimum random search pattern and is frequently detected in nature (Viswanathan et al. 2008). This distribution has infinite mean and infinite variance and also has a simple power law formula. S is the step length in levy flight algorithm. The parameters u and v are given by the normal distributions in Eq. 6. The variance, σ, is calculated as Eq. 7 with $1 \leq \beta \leq 2$ and $\Gamma(\mathcal{Z})$ is the gamma function (Yang and Deb 2009; Syberfeldt 2014).

$$L(S) \sim |S|^{-1-\beta} \quad 0 < \beta \leq 2,\ S \to \infty \tag{4}$$

$$S = \frac{u}{|v|^{\frac{1}{\beta}}} \tag{5}$$

$$u \sim N(0, \sigma_u^2)\,,\, v \sim N(0, \sigma_v^2) \tag{6}$$

$$\sigma_u = \left(\frac{\Gamma(1+\beta) * \sin(\pi\beta/2)}{\Gamma[(1+\beta)/2] * \beta * 2^{(\beta-1)/2}}\right)^{1/\beta}, \sigma_v = 1 \tag{7}$$

3.4 Opposition-based learning

Opposition-based learning (OBL) is a useful way to better explore problem space. This algorithm can help increase efficiency by searching for the solution and its opposite at the same time. Although this strategy raises the algorithm's computational load, but it helps a lot to achieve faster convergence (Oliva and Abd Elaziz 2020; Tizhoosh 2005; Aladeemy et al. 2020).

The OBL strategy in D-dimensional space is as Eq. 8.

$$\bar{x}_j = u_j + l_j - x_j \quad j = 1, 2, \ldots, D \tag{8}$$

In Eq. 8 the x is the habitat in search space and $\bar{x}$ is the opposite sides of the habitat which are generated within the interval [u, l]. Clearly, for each dimension the opposite number should be calculated which is represented by j. Finally, $\bar{x}$ represents the opposite habitat states in D dimension.

3.5 Disruption operator

Disruption operator (D_{op}) is implemented as a solution to enhance the population diversity. This phenomenon was inspired from the astrophysics. Harwit belives:

> When a set of gravitationally bound particles (with total mass m) is very close to a massive object (with mass M), then the set becomes a torn apart. Similar to this, when a solid body that held together through gravitational forces, approaches a much more massive object.

This operator is used because CCOALFDO algorithm searches better in the problem space and maintains the balance between exploration and exploitation processes. D_{op} is formulated in Eq. 9:

$$D_{\mathrm{op}} = \begin{cases} D_{is_{i,j}} \times \delta(-2, 2) & \text{if} D_{is_{i,\mathrm{best}}} \geq 1 \\ 1 + D_{is_{i,\mathrm{best}}} \times \delta\left(\frac{-10^{-4}}{2}, \frac{10^{-4}}{2}\right) & \text{otherwise} \end{cases}. \tag{9}$$

In Eq. 9 the $D_{is_{i,j}}$ represents the Euclidean distance between the ith solution and its nearest neighborhood jth solution, and $D_{is_{i,\mathrm{best}}}$ is the Euclidean distance between the ith solution and the best solution. Also, the $\delta(a, b)$ is a random number which is generated within the interval [a, b] (Neggaz et al. 2020).

Table 1 Chaotic maps

NO	Name	Chaotic map	Range
1	Chebyshev (Wang et al. 2001)	$x_{i+1} = \cos(i\cos^{-1}(x_i))$	(− 1,1)
2	Circle (Li-Jiang and Tian-Lun 2002)	$x_i = \mathrm{mod}\left(x_i + b - \left(\frac{a}{2\pi}\right)\sin(2\pi x_k), 1\right),$ $a = 0.5$ and $b = 0.2$	(0,1)
3	Gauss/mouse (Jothiprakash and Arunkumar 2013)	$x_{i+1} = \begin{cases} 1 & x_i = 0 \\ \frac{1}{\mathrm{mod}(x_i, 1)} & \text{otherwise} \end{cases}$	(0,1)
4	Iterative (Zhenyu 2006)	$x_{i+1} = \sin\left(\frac{a\pi}{x_i}\right),\ a = 0.7$	(− 1,1)
5	Logistic (Zhenyu 2006)	$x_{i+1} = ax_i(1 - x_i)$, a = 4	(0,1)
6	Piecewise (Saremi et al. 2014)	$\begin{cases} \frac{x_i}{p} & 0 \le x_i < p \\ \frac{x_i - p}{0.5 - p} & p \le x_i < 0.5 \\ \frac{1 - p - x_i}{0.5 - p} & 0.5 \le x_i < 1 - p \\ \frac{1 - x_i}{p} & 1 - p \le x_i < 1 \end{cases}, p = 0.4$	(0,1)
7	Sine (Wang et al. 2014)	$x_{i+1} = \frac{a}{4}\sin(\pi x_i),\ a = 4$	(0,1)
8	Singer (Simon 2008)	$x_{\mathrm{i}+1} = \mu\left(7.86x_{\mathrm{i}} - 23.31x_{\mathrm{i}}^2 + 28.75x_{\mathrm{i}}^3 - 13.302875x_{\mathrm{i}}^4\right),$ $\mu = 1.07$	(0,1)
9	Sinusoidal (Du et al. 2009)	$x_{i+1} = ax_i^2\sin(\pi x_i),\ a = 2.3$	(0,1)
10	Tent (Bhattacharya and Chattopadhyay 2010)	$x_{i+1} = \begin{cases} \frac{x_i}{0.7} x_i < 0.7 \\ \frac{10}{3}(1 - x_i) x_i \ge 0.7 \end{cases}$	(0,1)

4 Proposed chaotic cuckoo optimization algorithm with levy flight and disruption operator for feature selection

In this section, the proposed wrapper-based chaotic cuckoo optimization algorithm with levy flight and disruption operator (CCOALFDO) is introduced as a feature selection method. In the basic cuckoo optimization algorithm, the cuckoos move in the search space to modify their positions to any point in the space which is called the continuous space. Regarding the nature of the feature selection issues, the solutions are limited to the binary space [0,1] values.

With N_{pop} cuckoos, one can make a $N_{\mathrm{pop}} \times N_{\mathrm{var}}$ matrix. In the usual algorithm, at the first place of egg laying, cuckoos randomly select some habitats that are built in the algorithm, chaotic function speeds up the convergence of the algorithm.

Searching for the optimal solution in a chaotic way among the local minima using ergodicity, regularity and semi-stochastic properties, leaded to a high probable global optimal solution. In Table 1, chaos variables which are generated by Guass/mouse map in the CCOALFDO algorithm are described. However, chaos variables are now assumed to be generated via an arbitrary one-dimensional map which enables them to either modify in the range of (0.1) or we scale them in this range.

The initial value for Gauss/mouse chaotic function is a random number then the sequence values are calculated by Eq. 10:

$$x_0 = \mathrm{rand}$$
$$x_{i+1} = \begin{cases} 1 & x_i = 0 \\ \frac{1}{\mathrm{mod}(x_i, 1)} & \text{otherwise} \end{cases} \tag{10}$$

In order to find habitat in the next generation, cuckoos should search around to find the best.

Though there are many different evolutionary variants in the literature, cuckoo optimization algorithm problems of premature convergence and generating inefficient results are still persisting. Levy flights method is used to solve these problems and enable the CCOALFDO algorithm to generate more efficient results. Also, the disruption operator observed population diversity of habitats. Using this method, it is ensured that COA, which is unable to perform global search well, would do it more effectively and would not be trapped in local minima. Thus, random walks drawn from levy stable distribution are used in the proposed method to define steps. Then, the step size is calculated by Eq. 11. Here, the factor 0.01 is the typical length scale to avoid exaggerating the size of the step and make the solutions jump out of the design domain.

$$\text{Step size } = 0.01 \times S \tag{11}$$

In Eq. 12 the global random search done by the concept of levy flights is presented in which $x_i^{(t)}$ is the current state of the cuckoo i in its last search for finding a habitat after t iterations. Levy flight is applied to generate a new state $x_i^{(t+1)}$ of $x_i^{(t)}$. By levy flight, the new state of the habitat $x_i^{(t)}$ is calculated by Eq. 12 and *ELR* is described in Eq. 3.

$$x_{\mathrm{i}}^{(\mathrm{t}+1)} = x_{\mathrm{i}}^{(\mathrm{t})} + \mathrm{ELR} \times \mathrm{levy(s)} \tag{12}$$

For all habitat the OBL strategy in *D*-dimensional space should be calculated as Eq. 8: This doubles the number of habitats. The disruption operator should be applied to each habitat and their opposites. As mentioned earlier, D_{op} is calculated by Eq. 9. It is necessary to be sure that all the answers are not too close together and make the balance between exploration and exploitation processes. The inclusion of OBL and DO permits to increase the diversity of the habitats.

Since the continuous space of features ought to be transformed to the corresponding binary space, a new position of the habitat is used in the next formula in which $y^{j,t+1}$ refers to the cuckoo with *j* index after *t* iteration:

$$y^{\mathrm{j,t+1}} = \mathrm{f}(x) = \begin{cases} 1 & \text{if} \left(w\left(x^{\mathrm{j,t+1}}\right)\right) \geq \text{rand} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

$w(x^{j,t+1})$ is calculated by Eq. 14.

$$w(a) = \frac{1}{1 + \mathrm{e}^{10(\mathrm{a}-0.5)}} \tag{14}$$

To solve the feature selection problems, the continuous space (free position) must be transformed to their corresponding binary solutions.

For example, if the value of a bit equals to 1, its corresponding feature is selected in the feature subset, while 0 indicates nothing (Fig. 1). It means that the second and fourth features are selected for the feature subset.Fitness function described as Eq. 15:

$$\text{fitness}_t = \max\left((1-b) \times \text{ACC} + b \times \left(1 - \frac{m}{n}\right)\right) \tag{15}$$

in Eq. 16, *n* is the total number of features, *m* is the selected feature subset length, and b is a parameter corresponding to create a balance between the classification accuracy and feature reduction which can be in [0, 1]. It is illustrated in the experiments that *b_0.2*. The ACC parameter is the classification accuracy rate made by calling a rapid classifier such as *KNN* to evaluate the feature sets and generate the accuracy value based on Euclidean distance measure. The *K* parameter equals 5. ACC is described as Eq. 16:

$$\begin{aligned} \text{ACC} = \ & \text{correct classificationrate with } K \text{ Neighbors} \\ & \text{Classifier } (N\_\text{neighbors} = 5) \end{aligned} \tag{16}$$

During every iteration, fitness values are assigned to cuckoo's habitat in the search space. These positions are evaluated on each iteration, and the best position is selected as the best solution.

Figure 2 shows the framework of CCOALFDO algorithm for problem optimization. In general, the proposed algorithm depends on improving the behaviors of the cuckoo optimization algorithm by using the chaotic algorithm, levy flight, opposition-based learning and disruption operator. Clearly, each of these operators have their own tasks. The aim of using the chaotic algorithm is to improve the start position in the first step. The updating mechanism was used in the levy flight operation, since the cuckoos have the largest effect on the convergence of solutions leading to the global solution. Also, opposition-based learning is an intelligent solution for searching the opposite side of space and enhancing the exploration power. Whereas the disruption operator is used to enhance the

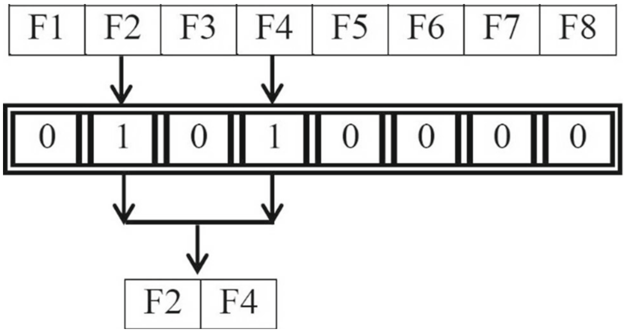
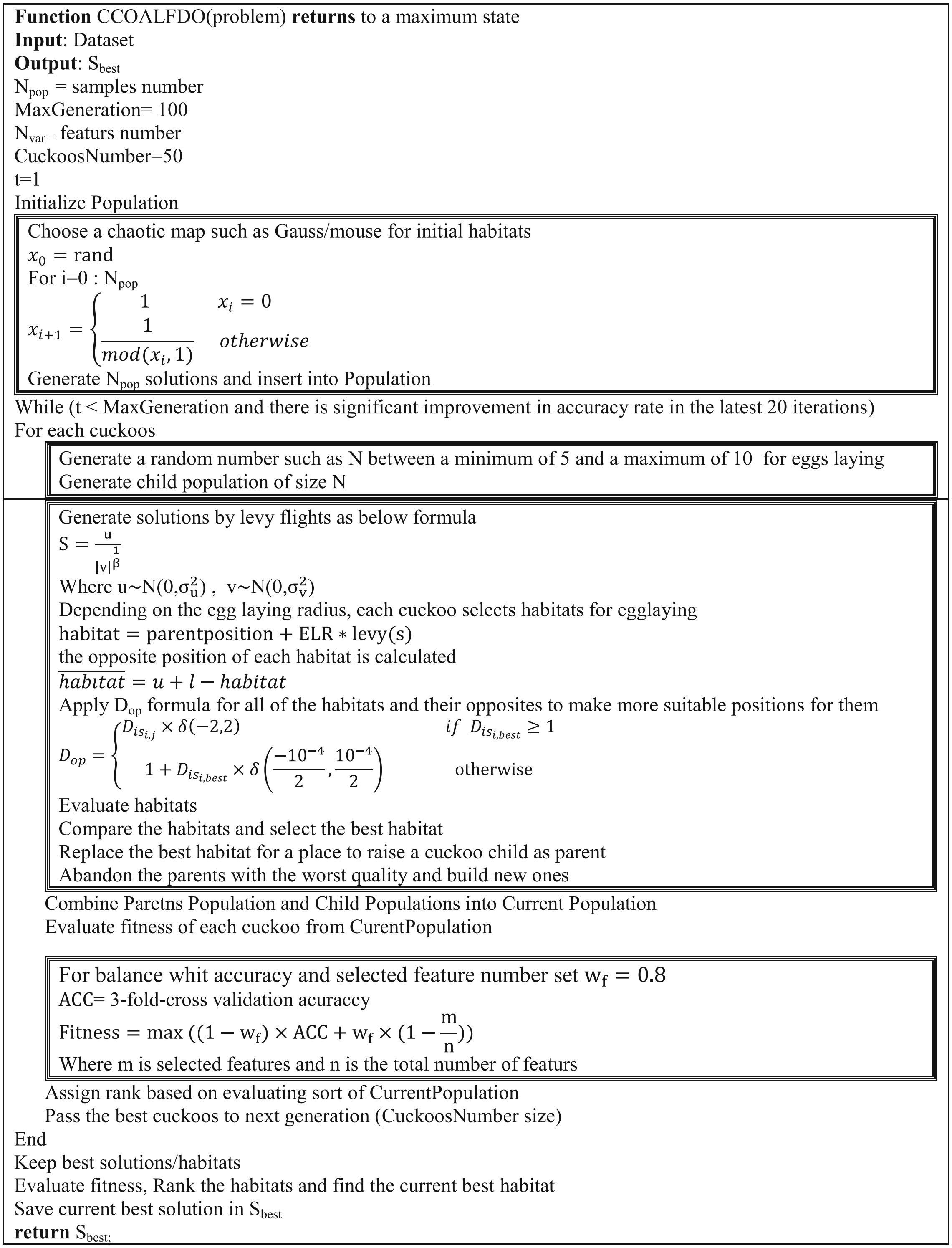


Fig. 1 Example of selected features in each sample

diversification of the whole population, which leads to improving the exploration ability of the CCOALFDO and convergence of the optimal solution. At the end, calling a rapid classifier such as KNN was done to evaluate the candidate solution and generate the fitness value based on the reduced dataset.

In CCOALFDO algorithm, two criteria were selected to terminate the algorithm:

1 maximum iteration (or generation) number
2 no significant improvement in accuracy rate in the latest 20 iterations



```
Function CCOALFDO(problem) returns to a maximum state
Input: Dataset
Output: $S_{best}$
$N_{pop}$ = samples number
MaxGeneration= 100
$N_{var\,=}$ featurs number
CuckoosNumber=50
t=1
Initialize Population
    Choose a chaotic map such as Gauss/mouse for initial habitats
    $x_0 = \text{rand}$
    For i=0 : $N_{pop}$
    $x_{i+1} = \begin{cases} 1 & x_i = 0 \\ \dfrac{1}{mod(x_i, 1)} & otherwise \end{cases}$
    Generate $N_{pop}$ solutions and insert into Population
While (t < MaxGeneration and there is significant improvement in accuracy rate in the latest 20 iterations)
For each cuckoos
        Generate a random number such as N between a minimum of 5 and a maximum of 10  for eggs laying
        Generate child population of size N
        Generate solutions by levy flights as below formula
        $S = \frac{u}{|v|^{\frac{1}{\beta}}}$
        Where $u \sim N(0,\sigma_u^2)$ ,  $v \sim N(0,\sigma_v^2)$
        Depending on the egg laying radius, each cuckoo selects habitats for egglaying
        $\text{habitat} = \text{parentposition} + \text{ELR} * \text{levy}(s)$
        the opposite position of each habitat is calculated
        $\overline{habitat} = u + l - habitat$
        Apply $D_{op}$ formula for all of the habitats and their opposites to make more suitable positions for them
        $D_{op} = \begin{cases} D_{is_{i,j}} \times \delta(-2,2) & if\ \ D_{is_{i,best}} \geq 1 \\ 1 + D_{is_{i,best}} \times \delta\left(\frac{-10^{-4}}{2}, \frac{10^{-4}}{2}\right) & \text{otherwise} \end{cases}$
        Evaluate habitats
        Compare the habitats and select the best habitat
        Replace the best habitat for a place to raise a cuckoo child as parent
        Abandon the parents with the worst quality and build new ones
    Combine Paretns Population and Child Populations into Current Population
    Evaluate fitness of each cuckoo from CurentPopulation

        For balance whit accuracy and selected feature number set $w_f = 0.8$
        ACC= 3-fold-cross validation acuraccy
        $\text{Fitness} = \max\left((1 - w_f) \times \text{ACC} + w_f \times (1 - \frac{m}{n})\right)$
        Where m is selected features and n is the total number of featurs
    Assign rank based on evaluating sort of CurrentPopulation
    Pass the best cuckoos to next generation (CuckoosNumber size)
End
Keep best solutions/habitats
Evaluate fitness, Rank the habitats and find the current best habitat
Save current best solution in $S_{best}$
return $S_{best;}$
```

Algorithm 1: The pseudo code for CCOALFDO algorithm

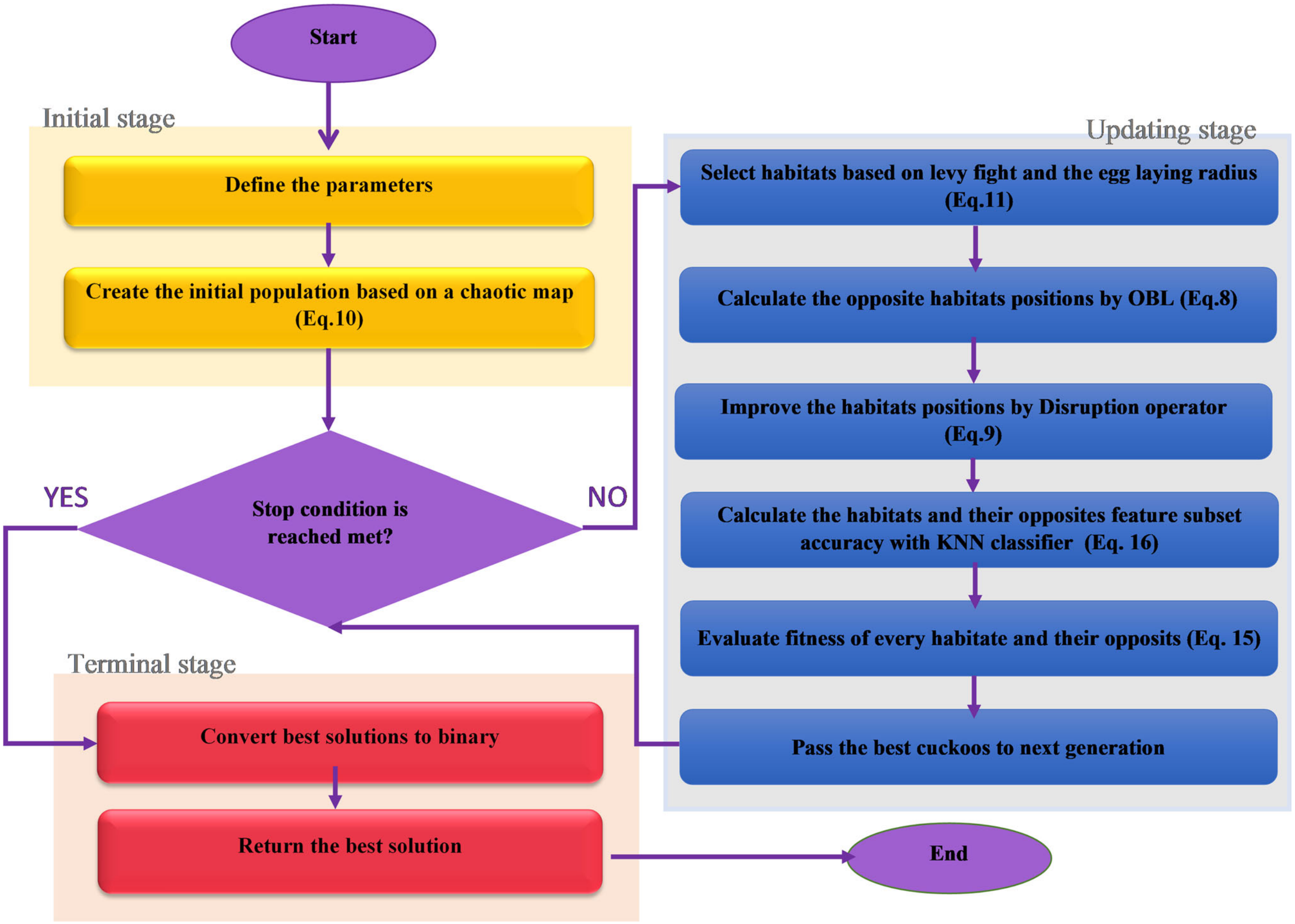

Start
Initial stage
Define the parameters
Create the initial population based on a chaotic map (Eq.10)
YES
Stop condition is reached met?
NO
Terminal stage
Convert best solutions to binary
Return the best solution
End
Updating stage
Select habitats based on levy fight and the egg laying radius (Eq.11)
Calculate the opposite habitats positions by OBL (Eq.8)
Improve the habitats positions by Disruption operator (Eq.9)
Calculate the habitats and their opposites feature subset accuracy with KNN classifier (Eq. 16)
Evaluate fitness of every habitate and their opposits (Eq. 15)
Pass the best cuckoos to next generation

Fig. 2 The framework of the proposed CCOALFDO method

The pseudo-code of CCOALFDO is shown in Algorithm 1. The algorithm starts by generating a random population of solutions with chaotic formula, then the iteration starts. The basic task in each iteration is determination of the best solution, and the coefficients in the next step. In the following phase, the solutions in the population are updated using levy flight and also the opposite position of each nest is calculated as new nests. It then applies D_{op} formula for all of the habitats and their opposites to make more suitable positions for them. This process is repeated until reaching a satisfying stopping constraint. Finally, the algorithm returns to the best solution.

The computational complexity of the proposed algorithm depends on several cases, including the cuckoo optimization algorithm (COA), the chaotic map (CM), the levy flight (LF) and disrupt operator (DO). Therefore, the complexity of the CCOALFDO algorithm is calculated by Eq. 17.

$$\theta(\text{CCOALFDO}) = K_s\theta(\text{COA}) + N \times \theta(\text{CM}) + (N - k_s)\theta(\text{LF}) + \theta(\text{DO}) \tag{17}$$

where $\theta(\text{COA})$ is calculated by Eq. 18.

$$\theta(\text{COA}) = \theta(t(\text{Dim} \times N + C \times N + N\log N)) \tag{18}$$

and $\theta(\text{CM})$ is calculated by Eq. 19.

$$\theta(\text{CM}) = \theta(t \times N) \tag{19}$$

and $\theta(\text{LF})$ is described as Eq. 20.

$$\theta(\text{LF}) = \theta(t(\text{Dim} \times N + C \times N) \tag{20}$$

And finally, $\theta(\text{DO})$ is calculated by Eq. 21.

$$\theta(\text{DO}) = \theta(t \times N) \tag{21}$$

In Eq. 17 to 21, t represents the number of iterations, Dim is the number of the features. N represents the number of feature subsets and also, Ks represents the number of habitats which updated in every iteration of the CCOALFDO algorithm, and C is the cost of objective function.

5 Experimental result

In this section, we perform the experiments on various high-dimensional datasets to compare the proposed CCOALFDO method with five other feature selection methods. At first, in Sect. 5.1 applied datasets are introduced concisely. Then, in Sect. 5.2, the experimental setups and parameters are discussed. Finally, in Sect. 5.3 the performance of the proposed method against other relevant methods is evaluated.

5.1 Datasets

The results presented in the experiments are based on the 15 real-world datasets from UCI machine learning repository (Lichman 2013), and the 5 medical datasets were used (Statnikov et al. 2005) that are listed in Table 2 and cover different sizes and dimensions. In Table 2, the number of data samples, features and classes for each dataset are given. In all datasets, the missing value is replaced with the median of that feature.

5.2 Experimental setup and parameters setup

The experiment was conducted on the Intel core i3 computing platform with 4 GB RAM, 2.13 GHz frequency. The programming environment is MATLAB R2014a in the Windows 10 operating system.

To compare the CCOALFDO algorithm, we used some other evolutionary algorithms such as whale optimization algorithm (WOA) (Mafarja and Mirjalili 2018), gray wolf optimizer (GOW) (Emary et al. 2016), particle swarm optimization (PSO) (Eberchart and Kennedy 1995), crow search algorithm (CSA) (Sayed et al. 2019) and cuckoo optimization algorithm (Rajabioun 2011). Table 3 shows the parameter setting of the algorithms used for comparing.

In order to train and validate $K-$ onefolds are used, while the last fold is used for testing in the k-fold cross-validation. The proposed method's performance should be confirmed by applying the fivefold cross validation to generate the five equal portions in the dataset.

The role of the training part involves training the applied classifier through optimization at the final evaluation, while the validation part assesses the performance of the classifier during the training time and the testing part evaluates features given the trained classifier which are picked up eventually. Based on fivefold cross validation, 80% of training partition is used for training, while 20% is used for the fitness function validation.

5.3 Result interpretation

In this section, we report and analyze the experimental results of the proposed CCOALFDO method and the compared feature selection methods. To provide a fair comparison, each method is conducting 10 solutions, and its average percentages are reported in comparison with other methods. All experiments use the same parameters for initialization and evolution. To execute CCOALFDO algorithm, the population size equals 50 and also the iteration number equals 100. It is noteworthy to mention that all the experiments were repeated for 30 independent times to obtain meaningful results.

Using the KNN algorithm for classification and dividing the correct sample classification to the overall number of samples, the accuracy rate of the algorithm (ACC) is found. In KNN algorithm, the K parameter equals 5. The optimum result would be the one with maximum accuracy rate and minimum number of features.

Table 4 presents the results of the proposed algorithm in each dataset. SF shows selected features in percent, and ACC is the accuracy rate in percent that reached to threefold cross-validation.

According to Table 4 the Gauss/mouse map obtains similar or even higher accuracy rates on all datasets in comparison with 10 other maps. However, it obtains insignificant lower accuracy rate in D3, D5, D7, D8, D13and D19 datasets. Also, the results are graphically shown in Fig. 3. As shown in Fig. 3, the red bar is usually higher than the other colors.

The Gauss/mouse map results of CCOALFDO are not much higher than other maps; therefore, it is used to compare CCOALFDO algorithm with other five algorithms in each dataset. Figure 4 depicted the convergence of the proposed method with Gauss/mouse map. As it is seen in Fig. 4, larger datasets such as D2, D6, D12 and D19 needs more than 100 iteration numbers to increase the accuracy rate, and if this number of iterations increases, they may perform better at the correct rate. The smallest datasets such as D7, D9, D10 and etc., are also the same. More iterations are not needed as 50 iterations become fixed.

Table 5 presents the accuracy and selected feature rate results. Because of the limitation of random behavior of cuckoos, the proposed algorithm caused faster convergence in most datasets.

According to Table 6, considering the average scores in the datasets, the CCOALFDO algorithm eventually shows the best performance in accuracy rate. The results are graphically shown in Fig. 5. Comparing the accuracy rate of different algorithms, the proposed algorithm shows greater accuracy than other algorithms in most datasets. In terms of ACC, the proposed CCOALFDO method achieves

Table 2 Experimental datasets (Lichman 2013; Statnikov et al. 2005)

No.	Dataset	Number of data samples	Number of features	Number of classes
1	Internet Advertisements (AD)	3279	1585	2
2	Dexter	300	20,000	2
3	Gisette	13,500	5000	2
4	DrivFace	606	6400	4
5	CNAE-9	1080	857	9
6	Gene expression cancer RNA-Seq	801	20,531	5
7	Madelon	2000	449	151
8	Arcene	900	10,000	2
9	Secom	1567	589	612
10	Semeion	1593	265	2
11	Daily and Sports Activities	9120	5625	19
12	DOROTHEA	1950	100000	2
13	P53 Mutants	16772	5409	2
14	ISOLET	7797	617	26
15	MICROMASS	931	1300	20
16	Brain_Tumor1	90	5920	5
17	DLBCL	77	5469	2
18	Leukemia1	72	5327	3
19	Prostate_Tumor	102	10509	2
20	SRBCT	83	2308	4

Table 3 State- of-the-art metaheuristic algorithm parameter settings

Algorithm	Parameter	Value
WOA (Mafarja and Mirjalili 2018)	A	2
	A2	− 1
	B	1
	The number of iterations	30
GOW (Emary et al. 2016)	A	2
	The number of iterations	30
	Population size	30
PSO (Eberchart and Kennedy 1995)	An inertial weight	1
	A inertia weight damping ratio	0.9
	population size	50
	Personal learning coefficient	1.5
	Global learning coefficient	2.0
	The number of iterations	30
CSA (Sayed et al. 2019)	Population size	50
	The number of iterations	30
	AP	0.1
	Fl	2
COA (Rajabioun 2011)	Population size	30
	The number of iterations	50
	Pa (discovery rate)	0.25
CCOALFDO (the proposed method)	Population size	50
	The number of iterations	100
	ELR	0.25

Table 4 Experiment chaotic maps on different datasets

		D1		D2		D3		D4		D5	
	Chaotic maps	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)
1	Chebyshev	47.62	78.39	25.62	87.52	43.05	72.18	24.32	81.32	42.85	**86.65**
2	Circle	43.32	86.95	26.03	90.70	31.82	93.65	29.85	81.65	31.42	84.25
3	Gauss/mouse	44.60	**89.22**	27.51	**91.66**	33.53	91.76	27.62	**82.31**	33.33	85.62
4	Iterative	48.92	85.44	30.26	88.03	31.65	**93.85**	20.65	80.92	31.25	86.45
5	Logistic	49.72	80.63	27.14	86.06	38.77	90.99	22.74	79.35	38.57	81.62
6	Piecewise	46.72	83.65	24.62	87.37	41.62	91.88	23.25	79.49	41.42	83.32
7	Sine	47.29	80.36	23.93	85.90	27.74	90.02	28.77	81.32	27.14	85.84
8	Singer	39.62	73.73	33.55	84.69	43.05	89.19	24.26	80.74	42.85	85.77
9	Sinusoidal	38.80	72.35	26.09	78.83	31.34	89.49	29.73	81.82	31.14	82.62
10	Tent	45.63	89.00	27.42	87.01	37.54	92.93	22.00	80.31	37.14	83.25
		D6		D7		D8		D9		D10	
	Chaotic maps	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)
1	Chebyshev	22.35	87.36	53.65	86.65	24.62	91.62	29.52	55.69	45.31	88.96
2	Circle	24.62	90.63	55.54	**89.28**	27.44	93.15	25.63	57.25	40.62	94.20
3	Gauss/mouse	26.12	**93.67**	56.98	87.75	21.22	91.26	30.62	**65.94**	47.62	**95.34**
4	Iterative	24.65	89.92	55.17	86.90	32.75	88.77	33.21	59.62	42.36	87.62
5	Logistic	25.62	88.99	56.73	85.32	32.10	87.92	28.05	59.73	35.94	86.25
6	Piecewise	23.37	89.35	54.65	84.72	27.95	89.11	35.52	58.15	42.62	87.11
7	Sine	25.91	88.60	56.52	87.98	23.27	90.23	37.15	59.63	36.15	85.62
8	Singer	26.66	90.25	57.38	85.56	24.36	85.60	29.23	62.75	43.32	80.61
9	Sinusoidal	24.25	85.22	57.42	83.65	26.25	**93.62**	27.64	60.19	42.21	79.45
10	Tent	25.25	92.92	56.21	89.00	29.31	92.88	31.29	63.95	35.52	85.78
		D11		D12		D13		D14		D15	
	Chaotic maps	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)
1	Chebyshev	24.62	83.62	23.62	87.52	35.92	78.35	26.63	81.93	34.62	78.87
2	Circle	25.95	88.66	25.93	89.08	34.84	77.34	25.74	**85.17**	47.63	77.36
3	Gauss/mouse	28.18	**90.72**	24.35	**90.36**	33.34	79.03	27.27	82.25	36.12	**83.67**
4	Iterative	27.62	87.43	24.15	90.11	27.00	72.62	29.61	82.07	35.62	77.23
5	Logistic	21.21	85.85	24.95	90.13	37.62	74.01	25.63	80.88	39.34	75.61
6	Piecewise	20.95	84.01	27.11	86.25	34.30	75.87	29.21	79.93	38.62	76.42
7	Sine	24.62	87.31	26.53	85.35	41.99	78.63	28.68	80.22	38.14	77.33
8	Singer	19.77	86.25	25.39	89.90	37.35	78.58	28.27	80.75	42.62	75.51
9	Sinusoidal	22.34	84.42	24.91	88.81	26.16	77.95	19.50	81.93	39.01	73.94
10	Tent	19.91	85.60	27.69	89.09	27.62	**79.21**	17.99	82.13	40.67	78.61
		D16		D17		D18		D19		D20	
	Chaotic maps	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)

Table 4 (continued)

		D16		D17		D18		D19		D20	
	Chaotic maps	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)
1	Chebyshev	31.72	82.31	25.85	84.83	27.22	74.11	26.14	82.22	45.73	79.51
2	Circle	22.62	87.26	23.65	86.02	26.39	76.66	25.74	**84.71**	42.36	79.23
3	Gauss/mouse	24.32	**90.22**	23.92	**88.58**	27.25	**85.85**	27.27	84.66	26.95	**84.25**
4	Iterative	26.28	83.11	24.62	87.21	25.68	80.11	29.15	83.71	35.29	82.74
5	Logistic	25.64	83.73	23.17	88.00	31.26	81.75	25.63	81.52	39.98	79.65
6	Piecewise	25.76	82.20	26.94	85.25	25.58	78.59	29.37	82.74	36.51	89.56
7	Sine	29.33	83.65	24.32	85.91	33.65	79.98	28.68	79.19	37.49	78.99
8	Singer	26.26	79.93	25.28	87.33	25.27	78.12	28.46	80.83	28.34	83.12
9	Sinusoidal	28.11	78.24	26.58	87.25	24.23	77.00	29.56	81.25	33.94	81.68
10	Tent	29.79	85.03	25.95	88.07	34.67	75.36	37.92	82.64	41.11	81.62

Bold fonts are best results

the best performance on the D2, D3, D4, D5, D6, D8, D11, D12, D13, D15, D16, D17, D18, D19, D20 datasets. Other datasets could not have the best accuracy rate which might happen due to the large number of classes. Figures 5 shows the boxplots of classification accuracy rate for the CCOALFDO, which is compared with 5 other evolutionary algorithms including WOA, GOW, PSO, CSA and COA that have been implemented and tested on twenty datasets in the same environment.

The Wilcoxon statistical test is also used to find about the proposed method more precisely. It is a non-parametric statistical test implemented to determine statistical significance between algorithms (Statnikov et al. 2005). The Wilcoxon test is conducted at 5% significance level to verify whether there is a statistical difference between obtained results of accuracy rate. In other words, if p value $< 0.~05$, then it indicates the proposed approach has a significant difference.

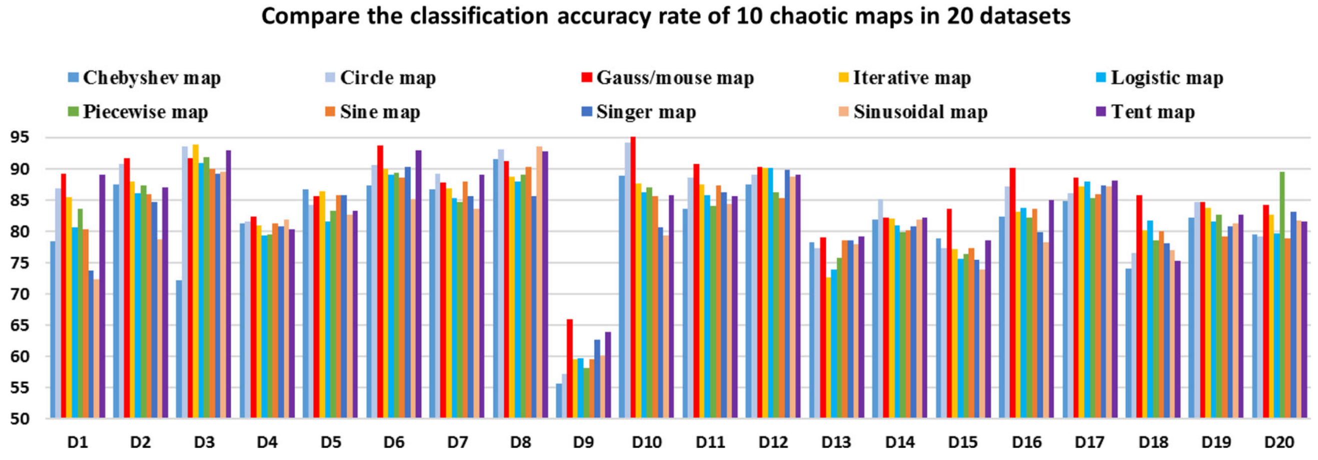


Fig. 3 Comparison of the classification accuracy rate of 10 chaotic maps in 20 datasets

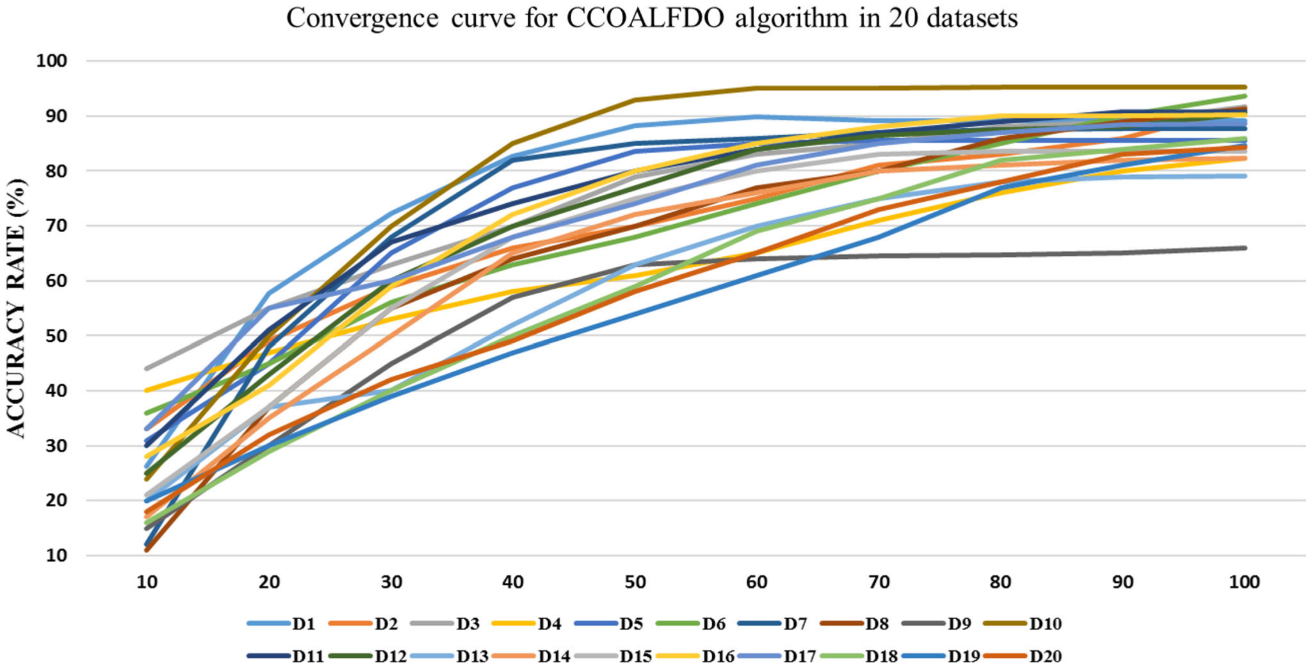


Fig. 4 Convergence curve for CCOALFDO with Gauss/mouse map on 20 datasets

According to a normative analysis on Wilcoxon test results, the outcome of the proposed method is meaningfully distinguishable from others. Considering accuracy, the CCOALFDO method performs significantly better than the competing methods.

Table 7 compares the results of the CCOALFDO with WOA, GOW, PSO, CSA and COA where bold values represent p-value > = 0.05. According to the results, the proposed algorithm showed more significant improvements comparing to other methods.

The comparison between CCOALFDO accuracy rate and five other evolutionary algorithms are presented in Fig. 5 and 6. These boxplots are created to report the accuracy rates of 30 independent runs for CCOALFDO and five others evolutionary algorithms. The boxplots prove the superiority of the proposed algorithm over other mentioned algorithms. It creates more compact boxes with a higher median than all other algorithms.

As Table 8 and Fig. 7 represent, CCOALFDO placed as the second-best method considering the number of features. On the D2, D6, D12, D15, D17, D19, D20 datasets, it obtained significantly better SF rank, while obtaining the best ACC rank in Table 6. The results show that the proposed method acts better in feature reduction in larger datasets. To evaluate the quality of each algorithm, standard division is also measured. The CCOALFDO achieved the smallest standard division value in most of datasets which proves the robustness of the algorithm and fixes it as the most stable method in terms of quality of feature selection.

Considering the p-values in Table 9, the proposed CCOALFDO outperforms others in most of the cases, significantly. The Wilcoxon test measure shows the superiority of the proposed method over WOA, PSO, CSA, COA algorithm. The results also denote that the GOW algorithm got higher p-values than others and achieved better rank comparing with CCOALFDO algorithm, however; its accuracy rate was lower.

According to Tables 6 and 8, the superiority of the proposed method over others in term of classification accuracy rate and feature reduction rate is clear. In all 20 real-world datasets, the CCOALFDO algorithm achieved the best rank in accuracy rate. There are some reasons which create this advantage. The first one is using the chaotic function to enhance the first positions of the cuckoos that leaded to maintaining the population and improved the search ability by levy flight for finding the best solution. Second, applying the DO operator which avoided getting trapped in local optima. And finally, using OBL improved the ability of exploration which guaranteed the diversity of the population. As a result, the performance of convergence improved, the chance of trapping in local minima decreased and the competence of the selected features was guaranteed.

Table 10 lists the average CPU time measured in seconds for per run, taken by the original feature set and the selected feature set. In all cases, the CCOALFDO

Table 5 Accuracy and selected feature rate comparing result

No		WOA (Mafarja and Mirjalili 2018)		GOW (Emary et al. 2016; Mirjalili et al. 2014)		PSO (Eberchart and Kennedy 1995)		CSA (Sayed et al. 2019)		COA (Rajabioun 2011)		CCOALFDO	
	Dataset	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)	ACC (%)	SF (%)
1	D1	89.74	48.03	88.24	43.08	86.05	46.55	87.01	48.9	88.45	46.32	89.22	44.60
2	D2	83.28	36.91	87.52	27.73	82.30	30.27	87.39	32.85	82.69	34.95	91.66	27.51
3	D3	85.33	40.29	85.97	38.67	83.69	37.52	87.88	33.06	82.55	37.91	91.76	33.53
4	D4	74.77	30.62	78.03	27.30	74.85	32.07	75.80	30.76	80.29	35.38	82.31	27.62
5	D5	81.50	39.61	83.38	34.04	82.78	42.62	77.31	32.14	80.43	43.62	85.62	33.33
6	D6	89.12	35.94	78.76	26.89	88.31	30.95	77.04	31.82	86.25	32.84	93.67	26.12
7	D7	90.24	52.78	89.50	58.31	90.52	55.62	86.43	52.61	88.95	58.51	87.75	56.98
8	D8	82.25	28.27	89.93	20.27	88.21	24.91	85.60	23.95	83.79	25.25	91.26	21.22
9	D9	67.22	33.02	67.71	24.95	69.13	31.11	66.06	29.21	64.65	32.96	65.94	30.62
10	D10	90.61	50.07	96.79	47.63	95.42	49.2	94.45	50.35	88.71	47.07	95.34	47.62
11	D11	85.87	39.61	88.68	26.52	88.69	32.91	84.36	35.82	89.13	30.52	90.72	28.18
12	D12	76.31	28.79	89.21	25.01	85.29	28.35	86.32	29.48	87.75	30.91	90.36	24.35
13	D13	69.06	36.25	76.71	32.99	76.41	35.76	71.19	33.8	71.76	35.89	79.03	33.34
14	D14	81.76	28.71	76.06	25.87	82.41	30.2	82.38	29.97	84.25	26.47	82.25	27.27
15	D15	80.71	40.65	76.22	36.23	81.16	40.57	82.12	39.64	78.17	42.64	83.67	36.12
16	D16	83.02	26.52	85.32	24.01	83.85	29.64	81.11	27.62	87.15	27.87	90.22	24.32
17	D17	79.13	28.31	77.95	24.42	76.18	26.95	76.38	29.78	78.91	28.56	88.58	23.92
18	D18	76.72	28.67	77.35	27.00	82.43	29.63	78.72	31.88	81.08	33.74	85.85	27.25
19	D19	73.31	35.56	73.62	28.91	74.13	30.63	81.45	31.70	77.35	33.49	84.66	27.27
20	D20	75.36	32.77	74.59	27.13	83.25	34.72	78.09	29.11	74.83	31.94	84.25	26.95

algorithm deletes the redundant and noisy features, so the classifier can provide enhanced time performance, while keeping better classification performance.

Table 11 indicates the average CPU time measured in seconds for per run, taken by the CCOALFDO algorithm and five others evolutionary algorithm. This measure indicates the speed of the algorithms in selecting features from a given dataset. In this measure, the GOW is the fastest algorithm, while the proposed algorithm ranks fourth. Because of more searching steps in the CCOALFDO algorithm, it takes more time for computations but it is still reasonable in this case because of its high accuracy rate and good feature selection rate. Although the CCOALFDO algorithm needs more enhancement in the time complexity issue, it can be fixed by applying the opposition-based learning and disruption operator for only a small part of the cuckoo's or based on a specific condition. Figure 8 represents these results graphically.

6 Conclusion and future works

Feature selection is an ever evolving frontier in data mining and machine learning. The more the machine learning develops, the broader the scopes of feature selection research are. In this study, a new algorithm is presented for feature selection according to the cuckoo optimization algorithm. The proposed CCOALFDO algorithm utilizes the optimization feature selection which transferred the random variables to chaotic behavior. Moreover, the levy flight is used to handle uncertainty and update cuckoo's steps in high-dimensional space. In addition, the opposition-based learning technique and the disruption operator are used to improve the exploration ability and ensure the diversity of the population to speed up the convergence rate. The evaluation of the proposed method was done through applying 20 datasets. Finally, the results were compared with WOA, GOW, PSO, CSA, COA algorithms. The results revealed that the presented method was able to utilize the information structure of a large number of features and enhance the accuracy rate of the classification tasks accordingly. By controlling the random behavior of

Table 6 Accuracy rate and standard deviation comparing result

No	ACC(%) dataset	WOA (Mafarja and Mirjalili 2018)	GOW (Emary et al. 2016; Mirjalili et al. 2014)	PSO (Eberchart and Kennedy 1995)	CSA Sayed et al. 2019)	COA (Rajabioun 2011)	CCOALFDO
1	D1	**89.74 ± 0.02 (1)**	88.24 ± 0.08 (4)	86.05 ± 0.14 (6)	87.01 ± 0.13 (5)	88.45 ± 0.13 (3)	89.22 ± 0.06 (2)
2	D2	83.28 ± 0.06 (4)	87.52 ± 0.03 (2)	82.30 ± 0.10 (6)	87.39 ± 0.03 (3)	82.69 ± 0.03 (5)	**91.66 ± 0.02 (1)**
3	D3	85.33 ± 0.09 (4)	85.97 ± 0.08 (3)	83.69 ± 0.13 (5)	87.88 ± 0.07 (2)	82.55 ± 0.12 (6)	**91.76 ± 0.06 (1)**
4	D4	74.77 ± 0.13 (6)	78.03 ± 0.06 (3)	74.85 ± 0.07 (5)	75.80 ± 0.09 (4)	80.29 ± 0.02 (2)	**82.31 ± 0.11 (1)**
5	D5	81.50 ± 0.16 (4)	83.38 ± 0.02 (2)	82.78 ± 0.03 (3)	77.31 ± 0.09 (6)	80.43 ± 1.03 (5)	**85.62 ± 0.02 (1)**
6	D6	89.12 ± 0.01 (2)	78.76 ± 0.03 (5)	88.31 ± 0.08 (3)	77.04 ± 0.05 (6)	86.25 ± 0.08 (4)	**93.67 ± 0.00 (1)**
7	D7	90.24 ± 0.01 (2)	89.50 ± 0.05 (3)	**90.52 ± 0.15 (1)**	86.43 ± 0.16 (5)	88.95 ± 0.15 (4)	87.75 ± 0.13 (6)
8	D8	82.25 ± 0.07 (6)	89.93 ± 0.09 (2)	88.21 ± 0.02 (3)	85.60 ± 0.07 (4)	83.79 ± 0.09 (5)	**91.26 ± 0.02 (1)**
9	D9	67.22 ± 0.12 (3)	67.71 ± 0.13 (2)	**69.13 ± 0.03 (1)**	66.06 ± 0.15 (4)	64.65 ± 0.21 (6)	65.94 ± 0.05 (5)
10	D10	90.61 ± 0.15 (5)	**96.79 ± 0.02 (1)**	95.42 ± 0.10 (2)	94.45 ± 0.07 (4)	88.71 ± 0.10 (6)	95.34 ± 0.08 (3)
11	D11	85.87 ± 0.04 (5)	88.68 ± 0.15 (4)	88.69 ± 0.11 (3)	84.36 ± 0.10 (6)	89.13 ± 0.22 (2)	**90.72 ± 0.03 (1)**
12	D12	76.31 ± 0.02 (6)	89.21 ± 0.08 (2)	85.29 ± 0.03 (5)	86.32 ± 0.06 (3)	87.75 ± 0.07 (4)	**90.36 ± 0.04 (1)**
13	D13	69.06 ± 0.06 (6)	76.71 ± 0.05 (2)	76.41 ± 0.05 (3)	71.19 ± 0.19 (5)	71.76 ± 0.18 (4)	**79.03 ± 0.02 (1)**
14	D14	81.76 ± 0.16 (5)	76.06 ± 0.10 (6)	82.41 ± 0.03 (2)	82.38 ± 0.16 (4)	**84.25 ± 0.02 (1)**	82.25 ± 0.16 (3)
15	D15	80.71 ± 0.08 (4)	76.22 ± 0.16 (6)	81.16 ± 1.00 (3)	82.12 ± 0.17 (2)	78.17 ± 0.12 (5)	**83.67 ± 0.11 (1)**
16	D16	83.02 ± 0.17 (5)	85.32 ± 0.12 (3)	83.85 ± 0.15 (4)	81.11 ± 0.16 (6)	87.15 ± 0.18 (2)	**90.22 ± 0.16 (1)**
17	D17	79.13 ± 0.03 (2)	77.95 ± 0.09 (4)	76.18 ± 0.02 (6)	76.38 ± 0.02 (5)	78.91 ± 0.05 (3)	**88.58 ± 0.00 (1)**
18	D18	76.72 ± 0.03 (6)	77.35 ± 0.09 (5)	82.43 ± 0.15 (2)	78.72 ± 0.16 (4)	81.08 ± 0.08 (3)	**85.85 ± 0.01 (1)**
19	D19	73.31 ± 0.02 (6)	73.62 ± 0.11 (5)	74.13 ± 0.08 (4)	81.45 ± 0.02 (2)	77.35 ± 0.09 (3)	**84.66 ± 0.07 (1)**
20	D20	75.36 ± 0.07 (4)	74.59 ± 0.03 (6)	83.25 ± 0.11 (2)	78.09 ± 0.02 (3)	74.83 ± 0.06 (5)	**84.25 ± 0.05 (1)**
	Average rank	4.30 (6)	3.50 (3)	3.45 (2)	4.15 (5)	3.90 (4)	**1.70 (1)**

Note that rank of each method is enclosed in the parenthesis and the best results are given in bold

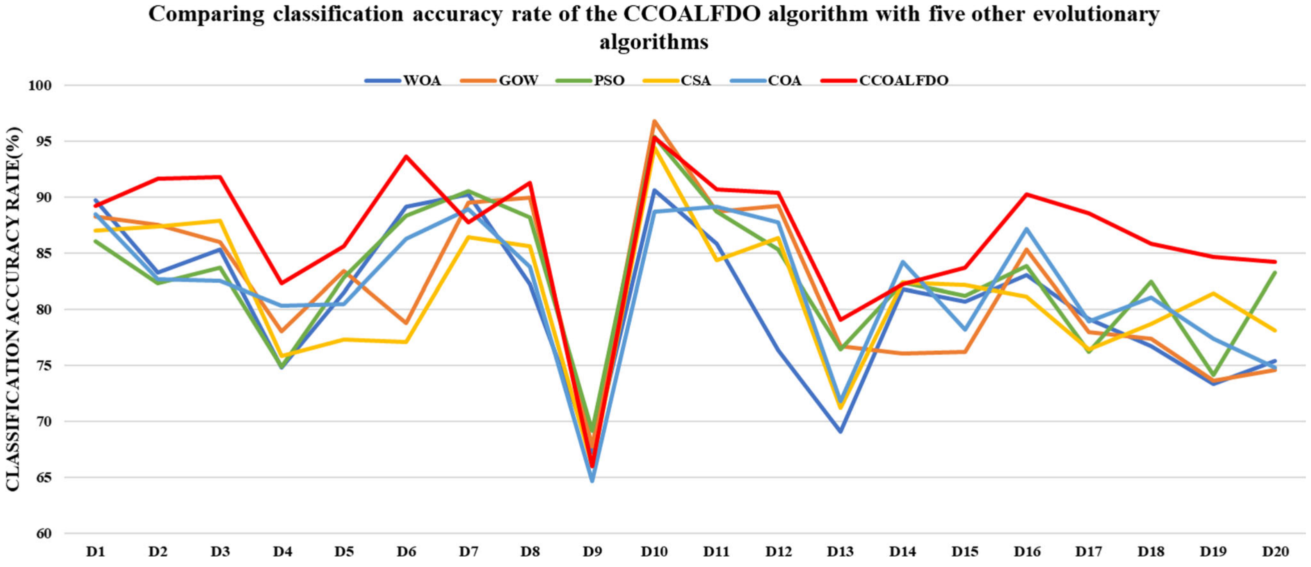


Fig. 5 Accuracy rate classification comparison between the CCOALFDO algorithm and five other evolutionary algorithms

Table 7 *p*-value of the Wilcoxon rank sum test for the classification accuracy rate based on CCOALFDO and five other evolutionary algorithms

No	CCOALFDO vs.	WOA (Mafarja and Mirjalili 2018)	GOW (Emary et al. 2016; Mirjalili et al. 2014)	PSO (Eberchart and Kennedy 1995)	CSA (Sayed et al. 2019)	COA [(Rajabioun 2011)
1	D1	3.44E − 05	4.07E − 05	5.94E − 05	5.55E − 05	**3.41E − 01**
2	D2	1.12E − 04	2.30E − 03	**4.92E − 01**	5.62E − 03	1.98E − 08
3	D3	1.34E − 06	2.45E − 08	2.34E − 08	3.3E − 09	1.98E − 09
4	D4	1.72E − 07	1.33E − 08	2.01E − 08	3.40E − 07	1.74E − 09
5	D5	**3.12E − 01**	6.01E − 04	1.37E − 03	5.78E − 03	3.55E − 03
6	D6	8.63E − 03	8.98E − 13	9.23E − 12	7.67E − 12	**4.74E − 02**
7	D7	5.94E − 06	4.62E − 05	7.64E − 07	5.90E − 6	8.91E − 07
8	D8	1.75E − 03	1.32E − 03	**4.95E − 01**	2.68E − 3	3.75E − 05
9	D9	8.61E − 12	7.38E − 10	8.25E − 11	7.16E − 11	7.38E − 11
10	D10	5.62E − 04	**4.91E − 01**	3.12E − 03	2.94E − 03	6.51E − 03
11	D11	1.77-E02	2.50E − 04	4.66E − 06	2.57E − 06	5.84E − 07
12	D12	4.75E − 10	5.95E − 10	4.38E − 10	5.73E − 10	6.14E − 10
13	D13	3.92E − 11	2.35E − 07	2.71E − 11	3.94E − 10	3.01E − 11
14	D14	**7.67E − 01**	8.32E − 08	5.32E − 09	**3.13E − 02**	8.22E − 08
15	D15	2.53E − 07	3.65E − 07	**1.36E − 01**	2.64E − 06	1.25E − 08
16	D16	6.48E − 12	7.21E − 12	8.22E − 13	6.32E − 11	6.79E − 12
17	D17	4.62E − 10	4.26E − 08	8.36E − 10	7.41E − 09	7.41E − 10
18	D18	6.52E − 11	8.02E − 09	7.77E − 10	3.94E − 10	6.95E − 10
19	D19	7.26E − 11	8.66E − 11	8.00E − 11	5.62E − 10	7.65E − 11
20	D20	7.17E − 12	8.92E − 10	7.61E − 11	6.32E − 10	8.23E − 11

p-values >=0.5 are in bold

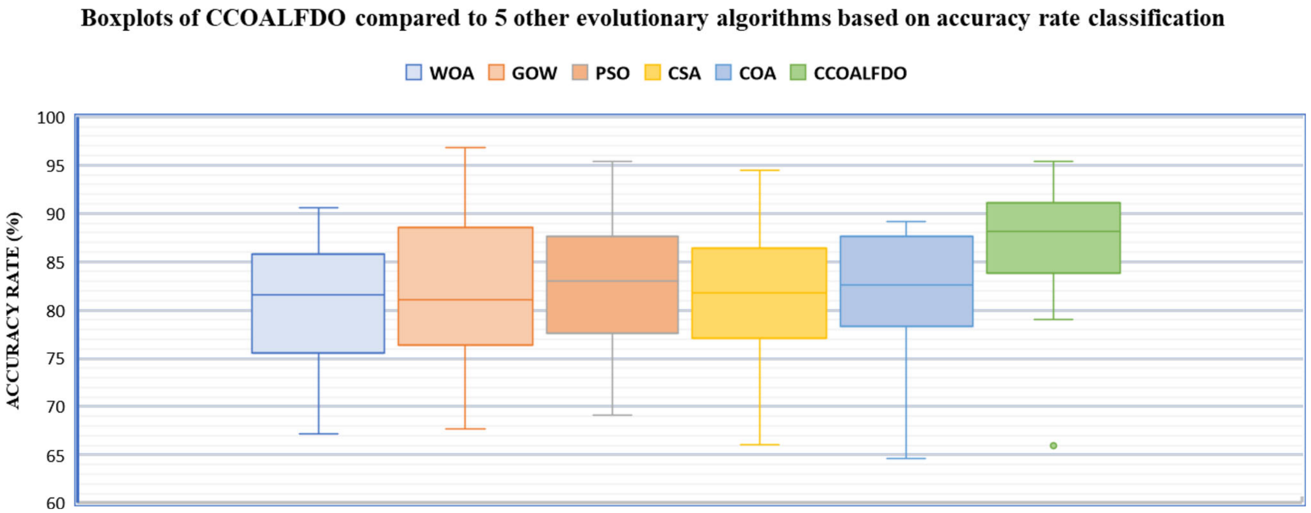


Fig. 6 Boxplots of CCOALFDO compared to 5 other evolutionary algorithms based on accuracy rate classification

this algorithm, a better feature set of datasets was selected. The rate of classification accuracy has improved, since some redundant features were eliminated. In other words, the CCOALFDO algorithm aims to find a compromise between two objectives maximizing the classification accuracy rate and reducing the number of features. The classification time was also reduced after decreasing the number of features and choosing a suitable feature set, while providing a more accurate classification. Unlike other evolutionary algorithms, applying levy flight, opposition-based learning and disrupt operator to the whole population leads to the solution diversity; however, the

Table 8 Selected features rate comparing results

No	SF(%) dataset	WOA (Mafarja and Mirjalili 2018)	GOW (Emary et al. 2016; Mirjalili et al. 2014)	PSO (Eberchart and Kennedy 1995)	CSA (Sayed et al. 2019)	COA (Rajabioun 2011)	CCOALFDO
1	D1	48.03 ± 0.68 (5)	**43.08 ± 0.02 (1)**	46.55 ± 0.05 (4)	48.90 ± 0.11 (6)	46.32 ± 0.52 (3)	44.60 ± 0.13 (2)
2	D2	36.91 ± 0.76 (6)	27.73 ± 0.13 (2)	30.27 ± 1.00 (3)	32.85 ± 0.07 (4)	34.95 ± 0.60 (5)	**27.51 ± 0.02 (1)**
3	D3	40.29 ± 0.51 (6)	38.67 ± 0.07 (5)	37.52 ± 0.07 (3)	**33.06 ± 0.02 (1)**	37.91 ± 0.09 (4)	33.53 ± 0.12 (2)
4	D4	30.62 ± 0.30 (3)	**27.30 ± 0.32 (1)**	32.07 ± 0.21 (5)	30.76 ± 0.12 (4)	35.38 ± 0.15 (6)	27.62 ± 0.14 (2)
5	D5	39.61 ± 0.75 (3)	34.04 ± 0.15 (2)	42.62 ± 0.22 (5)	**32.14 ± 0.01 (1)**	43.62 ± 0.18 (6)	33.33 ± 0.27 (4)
6	D6	35.94 ± 0.53 (6)	29.89 ± 0.36 (2)	30.95 ± 0.03 (3)	31.82 ± 0.63 (4)	32.84 ± 0.21 (5)	**26.12 ± 0.00 (1)**
7	D7	52.78 ± 0.40 (2)	58.31 ± 0.34 (5)	55.62 ± 0.74 (3)	**52.61 ± 0.57 (1)**	58.51 ± 0.84 (6)	56.98 ± 0.36 (4)
8	D8	28.27 ± 0.67 (6)	**20.27 ± 0.23 (1)**	24.91 ± 0.09 (4)	23.95 ± 0.03 (3)	25.25 ± 0.51 (5)	21.22 ± 0.01 (2)
9	D9	33.02 ± 0.90 (6)	**24.95 ± 0.01 (1)**	31.11 ± 0.08 (4)	29.21 ± 0.06 (2)	32.96 ± 0.73 (5)	30.62 ± 0.01 (3)
10	D10	50.07 ± 0.61 (5)	47.63 ± 0.07 (3)	49.20 ± 0.14 (4)	50.35 ± 0.27 (6)	**47.07 ± 0.68 (1)**	47.62 ± 0.06 (2)
11	D11	39.61 ± 1.01 (6)	**26.52 ± 0.74 (1)**	32.91 ± 0.18 (4)	35.82 ± 0.34 (5)	30.52 ± 0.03 (3)	28.18 ± 0.21 (2)
12	D12	28.79 ± 0.12 (4)	25.01 ± 0.30 (2)	28.35 ± 0.21 (3)	29.48 ± 0.95 (5)	30.91 ± 0.26 (6)	**24.35 ± 0.32 (1)**
13	D13	36.25 ± 0.67 (6)	**32.99 ± 0.02 (1)**	35.76 ± 0.25 (4)	33.80 ± 0.67 (3)	35.89 ± 0.75 (5)	33.34 ± 0.07 (2)
14	D14	28.71 ± 0.23 (4)	**25.87 ± 0.00 (1)**	30.20 ± 1.03 (6)	29.97 ± 0.32 (5)	26.47 ± 0.01 (2)	27.27 ± 0.14 (3)
15	D15	40.65 ± 0.34 (4)	36.23 ± 0.01 (2)	40.57 ± 0.75 (5)	39.64 ± 0.02 (3)	42.15 ± 0.65 (6)	**36.12 ± 0.25 (1)**
16	D16	26.52 ± 0.96 (3)	**24.01 ± 0.17 (1)**	29.64 ± 0.62 (6)	27.62 ± 0.17 (4)	27.87 ± 0.83 (5)	24.32 ± 0.41 (2)
17	D17	28.31 ± 0.61 (4)	24.42 ± 0.06 (2)	26.95 ± 0.47 (3)	29.78 ± 0.63 (6)	28.56 ± 0.25 (5)	**23.92 ± 0.00 (1)**
18	D18	28.67 ± 0.75 (3)	**27.00 ± 0.05 (1)**	29.63 ± 0.31 (4)	31.88 ± 0.07 (5)	33.74 ± 0.25 (6)	27.25 ± 0.32 (2)
19	D19	35.56 ± 0.42 (6)	28.91 ± 0.01 (2)	30.63 ± 0.54 (3)	31.70 ± 0.36 (4)	33.49 ± 0.15 (5)	**27.27 ± 0.40 (1)**
20	D20	32.77 ± 0.57 (5)	27.13 ± 0.00 (2)	34.72 ± 1.01 (6)	29.11 ± 0.36 (3)	31.94 ± 0.36 (4)	**26.95 ± 0.17 (1)**
	Average rank	4.65 (6)	**1.90 (1)**	4.10 (4)	3.75 (3)	4.65 (5)	1.95 (2)
	p-value	3.82E − 3	5.3E − 2	8.34E − 7	2.65E − 5	6.1E − 3	
	Wilcoxon test	✔	×	✔	✔	✔	

Note the rank of each method is enclosed in the parenthesis and the best results are given in bold

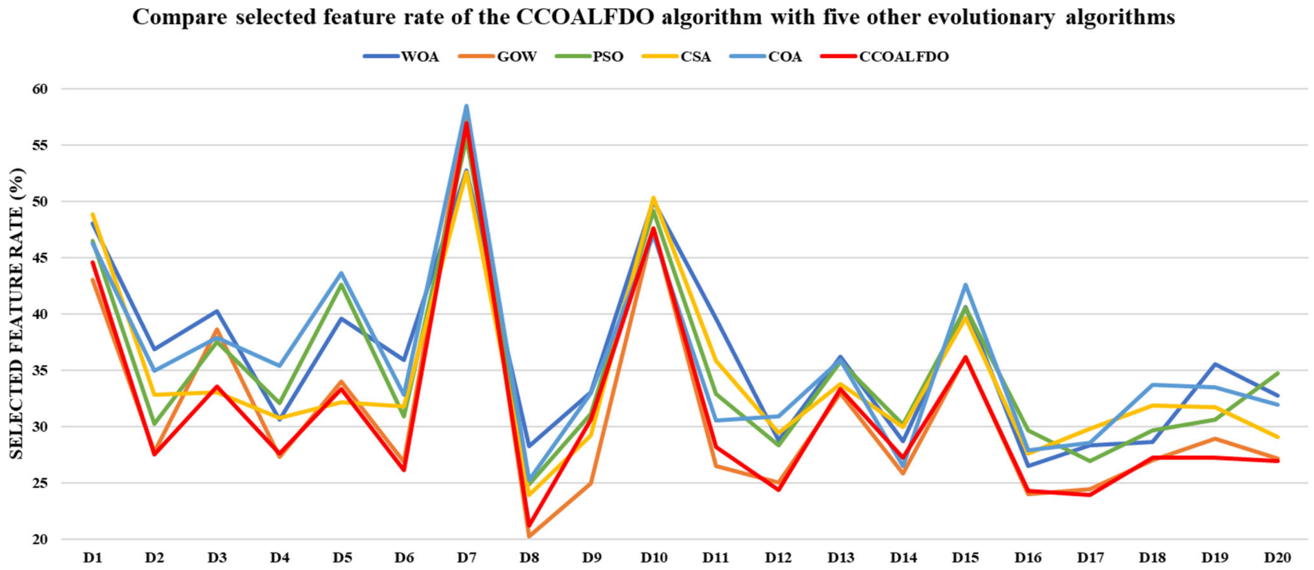


Fig. 7 Comparison selected feature rates of the CCOALFDO algorithm with five other evolutionary algorithms

Table 9 *p*-value of the Wilcoxon rank sum test for the selected features rate based on CCOALFDO and five other evolutionary algorithms

No	CCOALFDO vs.	WOA (Mafarja and Mirjalili 2018)	GOW (Emary et al. 2016; Mirjalili et al. 2014)	PSO (Eberchart and Kennedy 1995)	CSA (Sayed et al. 2019)	COA (Rajabioun 2011)
1	D1	1.62E − 07	**1.55E − 02**	3.64E − 07	2.54E − 07	3.72E − 03
2	D2	9.44E − 10	8.61E − 09	8.57E − 10	7.72E − 10	9.87E − 10
3	D3	8.42E − 08	8.56E − 08	7.00E − 07	**8.63E − 01**	7.14E − 07
4	D4	7.95E − 10	6.37E − 10	7.11E − 11	7.94E − 11	7.73E − 11
5	D5	5.84E − 03	5.67E − 03	6.94E − 03	5.62E − 03	5.88E − 03
6	D6	6.43E − 05	2.63E − 05	2.64E − 05	5.85E − 05	5.61E − 05
7	D7	**2.42E − 02**	8.51E − 10	7.96E − 10	**2.39E − 02**	3.66E − 10
8	D8	7.34E − 11	**3.65E − 01**	8.94E − 11	**4.79E − 02**	5.91E − 11
9	D9	8.32E − 10	**4.36E − 02**	7.62E − 10	5.74E − 02	7.85E − 10
10	D10	6.94E − 08	5.91E − 08	7.42E − 08	3.94E − 08	**3.84E − 01**
11	D11	3.85E − 10	**2.27E − 01**	1.64E − 10	6.54E − 10	8.64E − 09
12	D12	4.77E − 08	7.12E − 08	4.55E − 08	5.64E − 08	4.18E − 08
13	D13	7.94E − 08	8.94E − 02	5.28E − 07	1.31E − 08	1.65E − 08
14	D14	9.28E − 09	**7.95E − 01**	9.37E − 09	9.29E − 09	2.53E − 05
15	D15	5.96E − 07	4.48E − 06	3.20E − 08	5.93E − 07	6.29E − 08
16	D16	1.97E − 10	6.23E − 10	8.78E − 10	4.52E − 10	5.65E − 10
17	D17	4.95E − 09	3.32E − 09	3.41E − 09	5.11E − 09	7.94E − 09
18	D18	6.97E − 10	6.94E − 10	2.26E − 11	8.95E − 11	7.95E − 10
19	D19	8.46E − 10	7.94E − 10	8.22E − 10	8.23E − 10	3.67E − 11
20	D20	3.94E − 10	5.75E − 09	3.64E − 10	2.72E − 10	9.37E − 09

p-values >=0.5 are in bold

Table 10 Comparing runtime and accuracy rates between selected feature set and original feature set

		Result with original feature set			Result with selected feature set		
NO.	Dataset name	Feature number	Accuracy rate (%)	Time	Feature number	Accuracy rate (%)	Time
1	AD	1585	37.98	7.22	390	89.22	1.89
2	D2	20000	31.92	12.56	3502	91.66	3.54
3	D3	5000	39.19	15.32	1677	91.76	3.82
4	D4	6400	27.88	9.67	1128	82.31	2.81
5	D5	857	29.49	8.76	286	85.62	2.54
6	D6	20531	35.15	30.85	5363	93.67	8.85
7	D7	449	23.87	6.15	121	87.75	1.23
8	D8	10000	36.36	17.95	2122	91.26	4.37
9	D9	589	11.68	13.61	121	65.94	3.65
10	D10	265	37.98	8.50	47	95.34	2.32
11	D11	5625	28.28	13.83	1023	90.72	3.63
12	D12	100000	36.77	30.99	24350	90.36	8.75
13	D13	5409	12.97	14.36	722	79.03	4.40
14	D14	617	20.20	7.73	168	82.25	2.37
15	D15	1300	23.84	3.11	470	83.67	0.77
16	D16	5920	27.62	5.72	848	90.22	1.55
17	D17	5469	21.35	15.16	1308	88.58	4.34
18	D18	5327	22.62	26.78	919	85.85	6.28
19	D19	10509	31.05	14.02	2866	84.66	4.48
20	D20	2308	25.61	24.10	622	84.25	7.01

Table 11 Comparison between CCOALFDO and other metaheuristics based on time (s)

No.	Time(s) dataset	WOA (Mafarja and Mirjalili 2018)	GOW (Emary et al. 2016)	PSO (Eberchart and Kennedy 1995)	CSA (Sayed et al. 2019)	COA (Rajabioun 2011)	CCOALFDO
1	D1	1.97 (5)	2.13 (2)	1.25 (3)	**1.15 (1)**	2.05 (6)	1.89 (4)
2	D2	5.09 (5)	3.04 (2)	3.25 (3)	5.47 (6)	**2.04 (1)**	3.54 (4)
3	D3	4.91 (6)	3.86 (4)	3.82 (3)	3.84 (3)	**3.61 (1)**	4.81 (5)
4	D4	3.99 (6)	**3.28 (1)**	2.76 (2)	3.62 (5)	2.96 (4)	2.81 (3)
5	D5	0.94 (3)	1.09 (4)	0.62 (2)	2.64 (6)	**0.53 (1)**	2.54 (5)
6	D6	5.76 (2)	**4.37 (1)**	8.59 (4)	6.45 (3)	9.34 (6)	8.85 (5)
7	D7	3.44 (4)	2.33 (3)	2.48 (5)	1.91 (2)	3.15 (6)	**1.23 (1)**
8	D8	**3.39 (1)**	4.60 (4)	4.28 (2)	5.61 (6)	5.10 (5)	4.37 (3)
9	D9	4.18 (5)	4.12 (3)	4.17 (4)	6.56 (6)	**3.48 (1)**	3.65 (2)
10	D10	1.41 (4)	0.28 (1)	1.44 (5)	1.29 (3)	0.30 (2)	2.32 (6)
11	D11	3.83 (4)	3.73 (3)	4.27 (6)	4.01 (5)	**3.53 (1)**	3.63 (2)
12	D12	11.40 (4)	**6.20 (1)**	10.31 (3)	15.43 (6)	12.20 (5)	8.75 (2)
13	D13	5.99 (6)	4.39 (3)	5.65 (5)	4.33 (2)	**4.12 (1)**	4.40 (4)
14	D14	1.91 (3)	4.24 (6)	2.00 (4)	1.21 (2)	**1.18 (1)**	2.37 (5)
15	D15	**0.72 (1)**	1.84 (4)	1.86 (5)	0.99 (3)	2.77 (6)	0.77 (2)
16	D16	1.94 (3)	4.99 (6)	4.46 (5)	3.32 (4)	1.69 (2)	**1.55 (1)**
17	D17	4.77 (6)	6.75 (5)	**2.26 (1)**	3.04 (3)	2.47 (2)	4.34 (4)
18	D18	5.73 (5)	2.68 (2)	**3.21 (1)**	2.97 (3)	4.41 (4)	6.28 (6)
19	D19	6.39 (6)	**3.27 (1)**	4.33 (2)	5.83 (5)	4.74 (3)	4.48 (4)
20	D20	2.77 (3)	1.18 (2)	3.98 (5)	3.32 (4)	**1.06 (1)**	7.01 (6)
Average rank		4.10 (6)	**2.90 (1)**	3.50 (3)	3.90 (5)	2.95 (2)	3.65 (4)

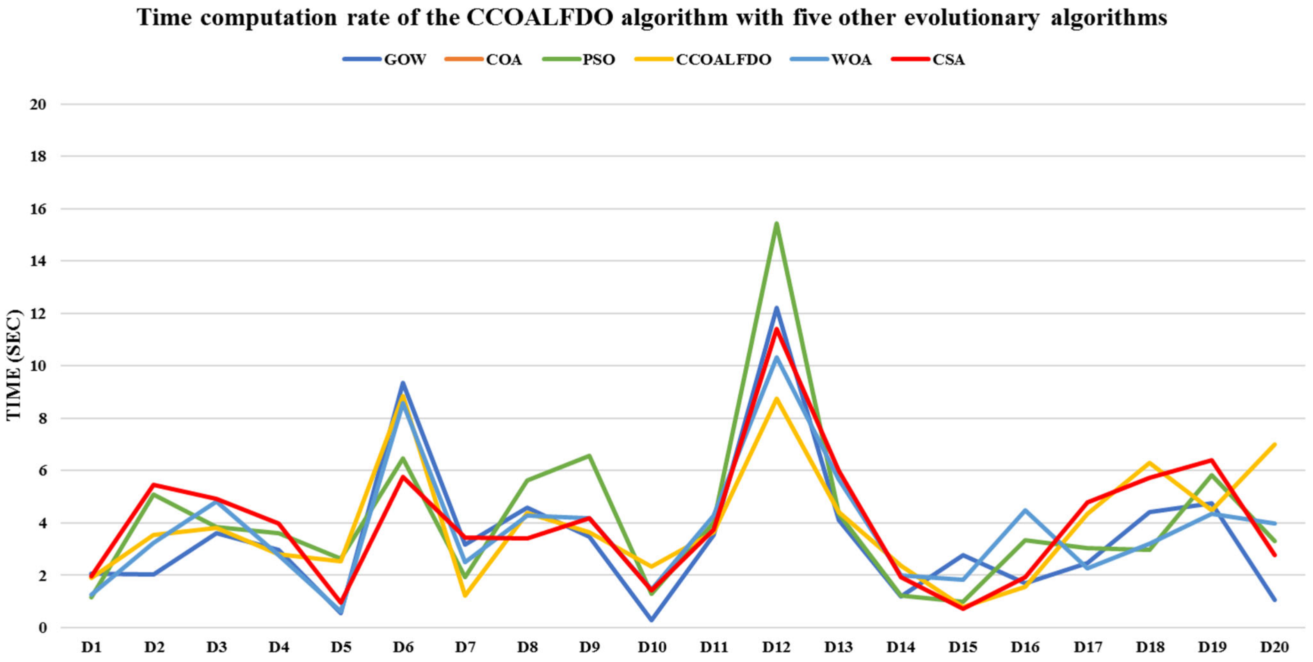


Fig. 8 comparing time computation rate of the CCOALFDO algorithm with five other evolutionary algorithms

computational time increases. This is an issue which can be handled in the future works. They can also focus on applying other rates, such as precision or G-mean and the use of CCOALFDO for feature selection issues in online applications. Moreover, the CCOALFDO can evaluate the real-world problems to ensure the stability.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with animals performed by any of the authors.

References

Aladeemy M et al (2020) New feature selection methods based on opposition-based learning and self-adaptive cohort intelligence for predicting patient no-shows. Appl Soft Comput 86:105866

Alatas B, Akin E, Ozer AB (2009) Chaos embedded particle swarm optimization algorithms. Chaos, Solitons Fractals 40(4):1715–1734

Ang JC et al (2015) Supervised, unsupervised, and semi-supervised feature selection: a review on gene selection. IEEE/ACM Trans Comput Biol Bioinf 13(5):971–989

Anter AM, Ali M (2020) Feature selection strategy based on hybrid crow search optimization algorithm integrated with chaos theory and fuzzy c-means algorithm for medical diagnosis problems. Soft Comput 24(3):1–20

Anter AM, Azar AT, Fouad KM (2019) Intelligent hybrid approach for feature selection. In: International conference on advanced machine learning technologies and applications. Springer

Arora S, Anand P (2019) Binary butterfly optimization approaches for feature selection. Expert Syst Appl 116:147–160

Bannigidad P, Gudada C (2019) Age-type identification and recognition of historical kannada handwritten document images using HOG feature descriptors. In: Iyer B, Nalbalwar S, Pathak N (eds) Computing, communication and signal processing. Springer, Berlin, pp 1001–1010

Beltramo T, Klocke M, Hitzmann B (2019) Prediction of the biogas production using GA and ACO input features selection method for ANN model. Inf Process Agric 6(3):349–356

Bhattacharya A, Chattopadhyay PK (2010) Hybrid differential evolution with biogeography-based optimization for solution of economic load dispatch. IEEE Trans Power Syst 25(4):1955–1964

Bostani H, Sheikhan M (2017) Hybrid of binary gravitational search algorithm and mutual information for feature selection in intrusion detection systems. Soft Comput 21(9):2307–2324

Cheng C, Bao L, Bao C (2016) network intrusion detection with bat algorithm for synchronization of feature selection and support vector machines. In: International symposium on neural networks. 2016. Springer

da Silva DL, Seijas LM, Bastos-Filho CJ (2017) Artificial bee colony optimization for feature selection of traffic sign recognition. Int J Swarm Intell Res (IJSIR) 8(2):50–66

Dai Y et al. (2015) Feature selection of high-dimensional biomedical data using improved SFLA for disease diagnosis. In: International conference on bioinformatics and biomedicine (BIBM), IEEE

Dara S, Banka H, Annavarapu CSR (2017) A rough based hybrid binary PSO algorithm for flat feature selection and classification in gene expression data. Ann Data Sci 4(3):1–20

De Souza RCT et al (2018) A V-shaped binary crow search algorithm for feature selection. In: 2018 IEEE congress on evolutionary computation (CEC). 2018. IEEE

Ding C, Peng H (2005) Minimum redundancy feature selection from microarray gene expression data. J Bioinf Comput Biol 3(02):185–205

dos Santos CL, Mariani VC (2008) Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization. Expert Syst Appl 34(3):1905–1913

Du D, Simon D, Ergezer M (2009) Biogeography-based optimization combined with evolutionary strategy and immigration refusal. In: IEEE International conference onsystems, man and cybernetics, 2009. SMC IEEE

Eberchart R, Kennedy J (1995) Particle swarm optimization. In: IEEE international conference on neural networks, Perth, Australia. 1995

El Aziz MA, Hassanien AE (2016) Modified cuckoo search algorithm with rough sets for feature selection. Neural Comput Appl 29(4):1–10

Emary E, Zawbaa HM (2018) Feature selection via Lèvy Antlion optimization. Pattern Anal Appl 22(3):1–20

Emary E, Zawbaa HM, Hassanien AE (2016) Binary gray wolf optimization approaches for feature selection. Neurocomputing 172:371–381

Fong S, Yang XS, Deb S (2013) Swarm search for feature selection in classification. In: 2013 IEEE 16th international conference on computational science and engineering (CSE), 2013. IEEE

Fong S, Wong R, Vasilakos AV (2016) Accelerated PSO swarm search feature selection for data stream mining big data. IEEE Trans Serv Comput 9(1):33–45

Gandomi AH, Yang X-S (2014) Chaotic bat algorithm. J Comput Sci 5(2):224–232

Gandomi AH, Yang X-S, Alavi AH (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 29(1):17–35

Hamidzadeh J, Namaei N (2018) Belief-based chaotic algorithm for support vector data description. Soft Comput 23:1–26

Hamidzadeh J, Monsefi R, Yazdi HS (2015) IRAHC: instance reduction algorithm using hyperrectangle clustering. Pattern Recogn 48(5):1878–1889

Hamidzadeh J, Sadeghi R, Namaei N (2017) Weighted support vector data description based on chaotic bat algorithm. Appl Soft Comput 60:540–551

Harde S, Sahare V (2016) Design and implementation of ACO feature selection algorithm for data stream mining. In: International conference on automatic control and dynamic optimization techniques (ICACDOT), IEEE

Himabindu K, Jyothi S, Mamatha D (2019) GA-based feature selection for squid's classification. In: Wang J, Reddy G, Prasad V, Reddy V (eds) Soft computing and signal processing. Springer, Berlin, pp 29–36

Hossain MA, Jia X, Benediktsson JA (2016) One-class oriented feature selection and classification of heterogeneous remote sensing images. IEEE J Sel Top Appl Earth Observ Remote Sens 9(4):1606–1612

Hu B et al (2016) Feature selection for optimized high-dimensional biomedical data using the improved shuffled frog leaping algorithm. IEEE/ACM Trans Comput Biol Bioinf 15(6):1765–1773

Huang C-L, Dun J-F (2008) A distributed PSO–SVM hybrid system with feature selection and parameter optimization. Appl Soft Comput 8(4):1381–1391

Huang P et al (2018) Feature extraction based on graph discriminant embedding and its applications to face recognition. Soft Comput 23(16):1–14

Huda S et al (2016) A hybrid feature selection with ensemble classification for imbalanced healthcare data: a case study for brain tumor diagnosis. IEEE Access 4:9145–9154

Hussien AG et al (2019) S-shaped binary whale optimization algorithm for feature selection. In: Bhattacharyya S, Mukherjee A, Bhaumik H, Das S, Yoshida K (eds) Recent trends in signal and image processing. Springer, Berlin, pp 79–87

Jayabarathi T, Raghunathan T, Gandomi A (2018) The bat algorithm, variants and some practical engineering applications: a review. In: Yang XS (ed) Nature-inspired algorithms and applied optimization. Springer, Cham, pp 313–330

Jothiprakash V, Arunkumar R (2013) Optimization of hydropower reservoir using evolutionary algorithms coupled with chaos. Water Resour Manag 27(7):1963–1979

Kriegel H-P, Kröger P, Zimek A (2009) Clustering high-dimensional data: a survey on subspace clustering, pattern-based clustering, and correlation clustering. ACM Trans Knowl Discov Data (TKDD) 3(1):1

Kumar L, Bharti KK (2019) An improved BPSO algorithm for feature selection. In: Khare A, Tiwary U, Sethi I, Singh N (eds) Recent trends in communication, computing, and electronics. Springer, Berlin, pp 505–513

Lee J, Kim D-W (2016) Efficient multi-label feature selection using entropy-based label selection. Entropy 18(11):405

Li Y, Li T, Liu H (2017) Recent advances in feature selection and its applications. Knowl Inf Syst 53(3):551–577

Lichman M (2013) UCI machine learning repository. University of California, School of Information and Computer Science, Irvine, p 2017

Li-Jiang Y, Tian-Lun C (2002) Application of chaos in genetic algorithms. Commun Theor Phys 38(2):168

Luo T et al (2018) Semi-supervised feature selection via insensitive sparse regression with application to video semantic recognition. IEEE Trans Knowl Data Eng 30(10):1943–1956

Mafarja M, Mirjalili S (2018) Whale optimization approaches for wrapper feature selection. Appl Soft Comput 62:441–453

Mafarja M et al (2019a) Whale optimisation algorithm for high-dimensional small-instance feature selection. Int J Parallel Emerg Distrib Syst. https://doi.org/10.1080/17445760.2019.1617866

Mafarja M et al (2019b) Binary grasshopper optimisation algorithm approaches for feature selection problems. Expert Syst Appl 117:267–286

Masud MM, et al (2010) Classification and novel class detection of data streams in a dynamic feature space. In: Joint European conference on machine learning and knowledge discovery in databases. 2010. Springer

Mirjalili S, Mirjalili SM, Lewis A (2014) Gray wolf optimizer. Adv Eng Softw 69:46–61

Mistry K et al (2017) A micro-GA embedded PSO feature selection approach to intelligent facial emotion recognition. IEEE Trans Cybern 47(6):1496–1509

Nayar N, Ahuja S, Jain S (2019) Swarm intelligence for feature selection: a review of literature and reflection on future challenges. In: Kolhe M, Trivedi M, Tiwari S, Singh V (eds) Advances in data and information sciences. Springer, Berlin, pp 211–221

Neggaz N, Houssein EH, Hussain K (2020a) An efficient henry gas solubility optimization for feature selection. Expert Syst Appl 152:113364

Neggaz N et al (2020b) Boosting salp swarm algorithm by sine cosine algorithm and disrupt operator for feature selection. Expert Syst Appl 145:113103

Oliva D, Abd Elaziz M (2020) An improved brainstorm optimization using chaotic opposite-based learning with disruption operator for global optimization and feature selection. Soft Comput 24:1–22

Peng H, Fan Y (2017) Feature selection by optimizing a lower bound of conditional mutual information. Inf Sci 418:652–667

Pes B, Dessì N, Angioni M (2017) Exploiting the ensemble paradigm for stable feature selection: a case study on high-dimensional genomic data. Inf Fusion 35:132–147

Qi C et al (2017) Feature selection and multiple kernel boosting framework based on PSO with mutation mechanism for hyper-spectral classification. Neurocomputing 220:181–190

Rajabioun R (2011) Cuckoo optimization algorithm. Appl Soft Comput 11(8):5508–5518

Ramírez-Gallego S et al (2017) A survey on data preprocessing for data stream mining: current status and future directions. Neurocomputing 239:39–57

Rao H et al (2019) Feature selection based on artificial bee colony and gradient boosting decision tree. Appl Soft Comput 74:634–642

Rodrigues D et al. (2014) A binary krill herd approach for feature selection. In: 2014 22nd international conference on pattern recognition (ICPR), 2014. IEEE

Saremi S, Mirjalili S, Lewis A (2014a) Biogeography-based optimisation with chaos. Neural Comput Appl 25(5):1077–1097

Saremi S, Mirjalili SM, Mirjalili S (2014b) Chaotic krill herd optimization algorithm. Procedia Technol 12:180–185

Sayed GI, Hassanien AE, Azar AT (2019) Feature selection via a novel chaotic crow search algorithm. Neural Comput Appl 31(1):1–18

Shunmugapriya P, Kanmani S (2017) A hybrid algorithm using ant and bee colony optimization for feature selection and classification (AC-ABC Hybrid). Swarm Evol Comput 36:27–36

Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713

Sivagaminathan RK, Ramakrishnan S (2007) A hybrid approach for feature subset selection using neural networks and ant colony optimization. Expert Syst Appl 33(1):49–60

Solorio-Fernández S, Carrasco-Ochoa JA, Martínez-Trinidad JF (2020) A review of unsupervised feature selection methods. Artif Intell Rev 53(2):907–948

Song J et al (2016) Deep and fast: deep learning hashing with semi-supervised graph construction. Image Vis Comput 55:101–108

Statnikov A et al (2005) A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. Bioinformatics 21(5):631–643

Sweetlin JD, Nehemiah HK, Kannan A (2017) Feature selection using ant colony optimization with tandem-run recruitment to diagnose bronchitis from CT scan images. Comput Methods Programs Biomed 145:115–125

Syberfeldt A (2014) Multi-objective optimization of a real-world manufacturing process using cuckoo search. In: Yang XS (ed) Cuckoo search and firefly algorithm. Springer, Cham, pp 179–193

Thaher T et al (2020) Binary Harris Hawks optimizer for high-dimensional, low sample size feature selection. In: Mirjalili S, Faris H, Aljarah I (eds) Evolutionary machine learning techniques. Springer, Berlin, pp 251–272

Tizhoosh HR (2005) Opposition-based learning: a new scheme for machine intelligence. In: International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, Web technologies and internet commerce (CIMCA-IAWTIC'06)

Viswanathan G, Raposo E, Da Luz M (2008) Lévy flights and superdiffusion in the context of biological encounters and random searches. Phys Life Rev 5(3):133–150

Wang N, Liu L, Liu L (2001) Genetic algorithm in chaos. Or Trans 5(3):1–10

Wang G-G et al (2014) Chaotic krill herd algorithm. Inf Sci 274:17–34

Wang G-G et al (2019) A comprehensive review of krill herd algorithm: variants, hybrids and applications. Artif Intell Rev 51(1):1–30

Xu S, Dai J, Shi H (2018) Semi-supervised feature selection based on least square regression with redundancy minimization. In: 2018 International joint conference on neural networks (IJCNN). 2018. IEEE

Yadav S, Ekbal A, Saha S (2018) Feature selection for entity extraction from multiple biomedical corpora: a PSO-based approach. Soft Comput 22(20):6881–6904

Yan C et al (2019) Hybrid binary coral reefs optimization algorithm with simulated annealing for feature selection in high-dimensional biomedical datasets. Chemom Intell Lab Syst 184:102–111

Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: Nature & biologically inspired computing, 2009. NaBIC 2009. World Congress on. 2009. IEEE

Yang X-K et al (2018) Semi-supervised minimum redundancy maximum relevance feature selection for audio classification. Multimed Tools Appl 77(1):713–739

Zare M, Eftekhari M, Aghamollaei G (2019) Supervised feature selection via matrix factorization based on singular value decomposition. Chemom Intell Lab Syst 185:105–113

Zhang M et al (2019) Multi-temporal SAR image classification of coastal plain wetlands using a new feature selection method and random forests. Remote Sens Lett 10(3):312–321

Zhenyu G et al (2006) Self-adaptive chaos differential evolution. In: Advances in natural computation. pp. 972–975

Zhong Z (2020) Adaptive graph learning and low-rank constraint for supervised spectral feature selection. Neural Comput Appl 32(11):1–10

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.