Feature selection method based on chaotic maps and butterfly optimization algorithm

Asmaa Ahmed Awad[1], Ahmed Fouad Ali[1], and Tarek Gaber[1,2,3]

[1] Suez Canal University, Dept. of Computer Science, Faculty of Computers and Informatics, Ismailia, Egypt.
[2] School of Science, Engineering and Environment, University of Salford, UK.
[3] Scientific Research Group in Egypt, (SRGE), Cairo, Egypt

Abstract. Feature selection (FS) is a challenging problem that attracted the attention of many researchers. FS can be considered as an NP hard problem, If dataset contains N features then 2^N solutions are generated with each additional feature, the complexity doubles. To solve this problem, we reduce the dimensionality of the feature by extracting the most important features. In this paper we integrate the chaotic maps in the standard butterfly optimization algorithm to increase the diversity and avoid trapping in local minima in this algorithm.. The proposed algorithm is called Chaotic Butterfly Optimization Algorithm (CBOA).The performance of the proposed CBOA is investigated by applying it on 16 benchmark datasets and comparing it against six meta-heuristics algorithms. The results show that invoking the chaotic maps in the standard BOA can improve its performance with accuracy more than 95%.

Keywords: Butterfly optimization algorithm, chaotic maps, feature selection, dimensionality reduction

1 Introduction

Nowadays, Dimensionality reduction becomes most common pre-processing step to prepare the dataset for machine learning algorithms. Dimensionality reduction helps machine learning and data mining algorithms to be more faster and efficient. Because of high-dimensional data and large number of features, the construction of a suitable machine learning model can be extremely demanding and often almost fruitless [4]. To reduce high dimensionality of data there are two approaches. The first one is feature extraction, which includes creation a new subset of features from the original features with low dimensionality [18]. The second one is feature selection which is used as pre-processing step to eliminate the irrelevant, redundant and noisy data and find set of informative features [19].

Feature selection (FS) is a challenging problem that attracted the attention of many researchers, its research dates back to the 1960s [10]. FS is a process for finding set of M features from original set N where $M < N$ without loss any information lead to decrease performance of learning algorithm. If dataset contains N features then 2^N solutions are generated with each additional feature, the complexity doubles.

FS algorithms are classified into two main categories: filter-based algorithms and wrapper-based algorithms [11]. Filter-based algorithms depend on statistical method for calculation relation between features and finding optimal parameters, in another

hand, Wrapper-based algorithms based on machine learning algorithms for finding near optimal features. Despite Wrapper-based algorithms are computationally expensive, they obtain better results than filter-based algorithms in selecting features as show in the results in [13]. Meta-heuristic algorithms act as a source of concepts, mechanisms and principles for designing of artificial computing systems to deal with complex computational problems [7]. They divided into four categories (e.g. evolutionary algorithms, Physics-based algorithms, Swarm-based algorithms, Human-based algorithms). Swarm intelligence (SI) is a collection of intelligent systems inspired by the collective intelligence of a group. This collective intelligence is achieved through the direct or indirect interactions of agents that are homogeneous in nature, yet co-operate with each other in their local environment without being aware of global context or pattern [5].

There are many SI algorithms have been applied to solve feature selection problem such as: Grasshopper optimization algorithm [12], salp swarm algorithm [8], Whale optimization algorithm [15] and Dragon optimization algorithm [13]. Many Researchers add new features to swarm algorithms to get better results in feature selection such as using chaotic maps with Crow search algorithm in [16], adding binary version to Butterfly algorithms to select optimal features and achieve maximum accuracy [3], using sigmoid and v-shaped functions to design binary Grasshopper optimization algorithms [12], introducing chaotic version of salp algorithms by using four different chaotic maps to control the balance between exploration and exploitation [1], considering various chaotic maps are considered in whale optimization algorithms for tunning it's main parameters [9], proposing binary version of grey wolf optimization algorithm for maximization accuracy [6], implementing chaotic version of particle swarm algorithm for feature selection [17].

Butterfly Optimization Algorithm (BOA) is novel natural inspired SI optimization algorithm that mimics the food foraging behavior of butterflies [2].BOA encounters two problems similar to other meta-heuristic algorithms; (1) entrapment in local optima and (2) slow convergence speed. In order to increase the diversity of the BOA and avoid trapping in local minima, we combine it with ten chaotic maps. The proposed algorithm is called Chaotic Butterfly Optimization Algorithm (CBOA). CBOA is tested on different dataset with different feature dimensions. CBOA is compared with the standard BOA and six meta-heurestics algorithms. The results show that CBOA outperforms the other algorithms and it can reduce the feature dimensionality with high accuracy.

The rest of the paper is organized as follow. An overview of the BOA and chaotic maps are given in Section 2. The proposed algorithm, CBOA is described in Section 3. The experimental results and discussion of the CBOA are reported in Section 4. The conclusion of the paper is shown in Section 5.

2 Background

In this section, we give an overview of the algorithms/techniques used in the proposed CBOA.

2.1 Butterfly optimization algorithm (BOA)

To explain the main idea of BOA algorithm [2], firstly we highlight the biological and natural behaviors of the butterfly insect. Then we describe the main steps of the algorithm and how it mimics the social life of the butterflies.

Biological and natural behaviors Butterflies are insects belong to Lepidoptera. They have five senses (smell, sight, taste, touch and hearing). They used these senses

for finding foods, searching for mating parcener, immigration from one place to another and escaping from enemies. Although these senses are very important for butterflies, smell sense considers the most important sense which help them for finding food. In the mating process, male butterfly can identify the female butterfly through her pheromone. When butterfly moves from one place to another, it generates a fragrance with intensity which is propagate over the distance. The other butterflies can sense this fragrance and attracted to the butterfly according to the intensity of its fragrance. When a butterfly senses the best butterfly's fragrance it moves toward it. This process is called global search, while when it fails to sense the fragrance of any butterfly, it moves randomly to a new position in the search space. This process is called local search [2].

Magnitude of fragrance Butterfly emits a fragrance with intensity when it moves. The other butterflies attracted to the butterfly according to its magnitude of fragrance. The fragrance of each butterfly can be defined as in Equation (1).

$$pf_i = cI^a \tag{1}$$

Where pf_i represents the perceived magnitude of fragrance,I is fragrance intensity. The parameters c, a are a power exponent which represents the degree of the fragrance absorbtion and the sensor modality, respectively.

Butterflies movement The movement of butterflies are based on three phases as follow.

- **Global search phase.** Each butterfly emits fragrance when it moves and the other butterflies attracted to it according to its magnitude of fragrance. This process is called a global search and can be defined as follow

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + (r^2 \times \boldsymbol{g}^* - \boldsymbol{x}_i^t) \times f_i \tag{2}$$

  Where $\boldsymbol{x}_i^t$ is a vector which represent the butterfly (solution) at iteration t, $\boldsymbol{g}^*$ is the overall best solution, r is a random number in $[0, 1]$ and f_i is a fragrance of $i_t h$ butterfly. **In this phase, g initial value is the position of the minimum fitness of all solutions and it calculated by assigning a fitness value for each solution and define the minimum fitness then update the g (position) according to the minimum fitness. Also, r value is not used to calculate fitness,it is controlled by p (switch probability) value and the intial value of p =0.8. The r value is compared with p value to control the butterfly while moving to the best solution with minimum filness in local search or global search.**
- **Local search phase.** When the butterflies fail to sense the fragrance of the other butterflies, they move randomly in the search space. The process is called local search and it can be defined as follow.

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + (r^2 \times \boldsymbol{x_j}^t - \boldsymbol{x}_k^t) \times f_i \tag{3}$$

  Where $\boldsymbol{x}_j^t, \boldsymbol{x_k}^t$ are two vectors that represent two different butterflies in the same population.
- **Solution evaluation.** The fragrance intensity of the butterfly represents its objective function. The butterfly attracts the other butterflies according to its magnitude of fragrance.

Algorithm 1 Butterfly optimization algorithm (BOA)

Set the initial values of the population size n (butterflies), parameters a (power exponent), c sensory modality, switch probability ρ, and the maximum number of iterations Max_{itr}.
Set $t := 0$. {Counter initialization}.
for $(i = 1 : i \leq n)$ **do**
    Generate an initial population (butterflies) $\boldsymbol{x_i^t}$ randomly.
    Evaluate the fitness function of each butterfly (solution) $f(\boldsymbol{x_i^t})$.
    Calculate the fragrance for $\boldsymbol{x_i^t}$ as shown in Equation 1.
    Assign the overall best butterfly (solution) $\boldsymbol{g}^*$.
end for
repeat
    Set $t = t + 1$.
    for $(i = 1 : i \leq n)$ **do**
        Generate random number r, $r \in [0, 1]$.
        if $(r < \rho)$ **then**
            Move butterflies towards the best butterfly $\boldsymbol{g}^*$ as shown in Equation 2. {Global search}.
        else
            Move butterflies randomly as shown in Equation 3. {Local search}.
        end if
        Evaluate the fitness function of each butterfly (solution) $f(\boldsymbol{x_i^t})$.
        Assign the overall best solution $\boldsymbol{g}^*$.
    end for
    Update the value of parameters $c = [0.01, 0.25]$.
until $(t > Max_{itr})$. {Termination criteria satisfied}.
Produce the best solution $\boldsymbol{g}^*$.

The BOA algorithm In this subsection, we present the main steps of the BOA as follow.

- **Parameter setting.** At the beginning, we initialize the algorithm's parameter values such as the population size n, parameters a (power exponent), c sensory modality, switch probability ρ, and the maximum number of iterations Max_{itr}.
- **Iteration initialization .** Set the initial value of the iteration counter t.
- **Initial population.** The initial population n is generated randomly $\boldsymbol{x_i^t}$.
- **Solutions evaluation.** Each butterfly $\boldsymbol{x_i^t}$ in the population is evaluated by calculating its fitness function $f(\boldsymbol{x_i^t})$.
- **Global best solution.** Assign the overall best butterfly (solution) $\boldsymbol{g}^*$ in the population.
- **The main loop.** The following steps are repeated until the termination criterion satisfied.
  - **iteration increasing.** The iteration counter is increasing, $t = t + 1$.
  - **Random number generation.** We generate random number number r, where $r \in [0, 1]$.
  - **Balancing between global and local search processes.** The global and local search processes are applied according to the parameters value of ρ and r.
  - **Global search process.** The butterflies update their position according to the position of the overall best solution $\boldsymbol{g}^*$ as shown in Equation 2.
  - **Local search process.** If the butterflies fail to sense the fragrance of any butterfly in the population, they move randomly as shown in Equation 3.
- **Solutions evaluation.** Each butterfly $\boldsymbol{x_i^t}$ in the population is evaluated by calculating its fitness function $f(\boldsymbol{x_i^t})$.
- **Global best solution.** Assign the overall best butterfly (solution) in the population $\boldsymbol{g}^*$.
- **Termination criteria satisfied.** The overall processes are repeated until termination criteria satisfied, which is reaching to the maximum number of iterations Max_{itr} in our case.
- **Produce the best solution.** Produce the best obtained butterfly (solution) so far $\boldsymbol{g}^*$.

2.2 Chaotic maps

We investigate the effect of integrating ten chaotic maps in the proposed CBOA to increase the random behavior (diversity) of it to arrive at a global minima and avoid getting stuck at a local minima which lead to faster convergence. The initial point for the ten chaotic maps is random number between [0,1]. We used the same chaotic maps and their parameters which are reported in [14].

3 The proposed chaotic butterfly optimization algorithm (CBOA)

The proposed CBOA algorithm depends on the integration of chaotic maps in the standard BOA. The main steps of the proposed CBOA are as follows.

1. invoking the chaotic maps for updating butterfly positions instead of using random variables so this would improving the accuracy of CBOA. The Equations 2 and 3 will be modified by replacing r^2 by C_j as following:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + (C_j \times \boldsymbol{g}^* - \boldsymbol{x}_i^t) \times f_i \tag{4}$$

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + (C_j \times \boldsymbol{x_j}^t - \boldsymbol{x}_k^t) \times f_i \tag{5}$$

where C_j is the chaotic map and $j = 1, 2, \ldots, 10$. **Note that the C_j values are chaotic values, generated using 10 chaotic maps, are replaced with the r value to get better results in accuracy and minimum fitness than original algorithm use random values.**

2. transferring continues CBOA to binary CBOA: apply the binary CBOA will represent the search space in binary values [0, 1], so the binary CBOA can adaptively search the feature space for best feature combination and is expected to be much simpler than continuous version.binary CBOA represented in the following equations:

$$x_i^{t+1} = \begin{cases} 1 & if(s(x_i^{t+1})) \geq rand() \\ 0 & otherwise \end{cases} \tag{6}$$

where s is a transfer function, $rand()$ is a random number generated from uniform distribution [0,1] and x_i^{t+1} is the updated solution.

$$s(x_i^{t+1}) = \frac{1}{1 + \exp^{10(x_i^{t+1} - 0.5)}} \tag{7}$$

4 Results and Discussion

In this section, we present the parameter setting and the obtained results of the CBOA as shown in Table 1.

4.1 Parameter setting

The initial values of c, a are obtained from [2]: a is a constant and c is in the range [0.01,0.25].The other parameters values are chosen based on a number of experiments where their values were found to be the most efficient ones.

Table 1. Parameter setting

Parameters	Definitions	Values
p	switch probability	0.8
a	power exponent	0.1
c	sensory modality(c)	Min =0.01 , Max=0.25
n	Search Agents no	7
Max_{itr}	Maximum number of iteration	30
M	number of runs	20
K	cross validation	5

4.2 Fitness Function

Each solution is evaluated based on its fitness function, which employs the k-nearest neighbors (KNN) classifier as an evaluator and considers the number of selected features in the solution. The fitness function in Eq 8 is used to minimize the classification error in our proposed CBOA as follow.

$$Fitness = Minimize(\alpha \ \ Errrate + \beta \ \frac{|n|}{|N|}) \tag{8}$$

where *Errrate* represents the classification error rate,$|n|$ and $|N|$ are the number of selected features and the number of original features in the dataset respectively, α and β are the weights of the classification error rate and selection ratio, $\alpha \in [0,1]$ and $\beta = (1-\alpha)$.

4.3 Dataset Description

Sixteen dataset from UCI machine learning repository are used to prove the efficiency of our proposed algorithm. The dataset is reported in [3]. Two datasets (Breast-cancer, Congress) contain missing values. In this work, all the missing values are replaced by the median values of all known values of a feature given class as calculated in [16] .

4.4 Performance Analysis

In this subsection we analyze the performance of CBOA algorithm as follows.

The effect of integrating the chaotic maps in CBOA To verify the efficiency of invoking the chaotic maps in the proposed CBOA. We present the convergence curves for eight random datasets to show the performance of the standard BOA and the other ten modified CBOA as shown in Figure 1.

Comparison between CBOA and the state-of-art algorithms We investigate the performance of the proposed CBOA by comparing it against six algorithm. These algorithms are Antloin (ALO) , brain storm optimizer (BSO), genetic algorithm (GA), greywolf (GWO), particle swarm optimization (PSO), binary butterfly optimizer (BBOA). The results of these algorithm are reported in [3]. The results in Table 2 show that CBOA achieves better results in maximize classification accuracy than the other six algorithms, while the results in Table 3 show that the mean fitness value for CBOA obtain the best results in most cases.

Table 2. Comparison between CBOA and other algorithms based on Average Accuracy:

Dataset	ALO	BSO	GA	GWO	PSO	BBOA	CBOA
Breast-cancer	0.959	0.920	0.959	0.960	0.960	0.969	0.997
Bresat EW	0.939	0.902	0.949	0.938	0.938	0.970	0.985
Clean1	0.847	0.826	0.870	0.858	0.855	0.883	0.955
Congress	0.937	0.855	0.941	0.933	0.924	0.959	0.997
HeartEW	0.777	0.695	0.787	0.777	0.779	0.824	0.879
IonosphereEW	0.860	0.854	0.894	0.868	0.880	0.907	0.977
KrVsKp	0.901	0.760	0.922	0.914	0.920	0.966	0.957
Lymphography	0.786	0.693	0.816	0.763	0.791	0.868	0.960
PenglungEW	0.807	0.768	0.672	0.834	0.814	0.878	0.912
Semeion	0.971	0.960	0.976	0.967	0.968	0.982	0.979
SonarEW	0.849	0.794	0.875	0.862	0.784	0.846	0.942
SpectEW	0.788	0.751	0.810	0.785	0.784	0.846	0.829
Tic-Tac-Toe	0.759	0.660	0.761	0.754	0.751	0.798	0.837
WaveformEW	0.707	0.615	0.692	0.709	0.719	0.743	0.803
WineEW	0.954	0.865	0.954	0.948	0.952	0.984	0.989
Zoo	0.922	0.813	0.929	0.953	0.945	0.978	1

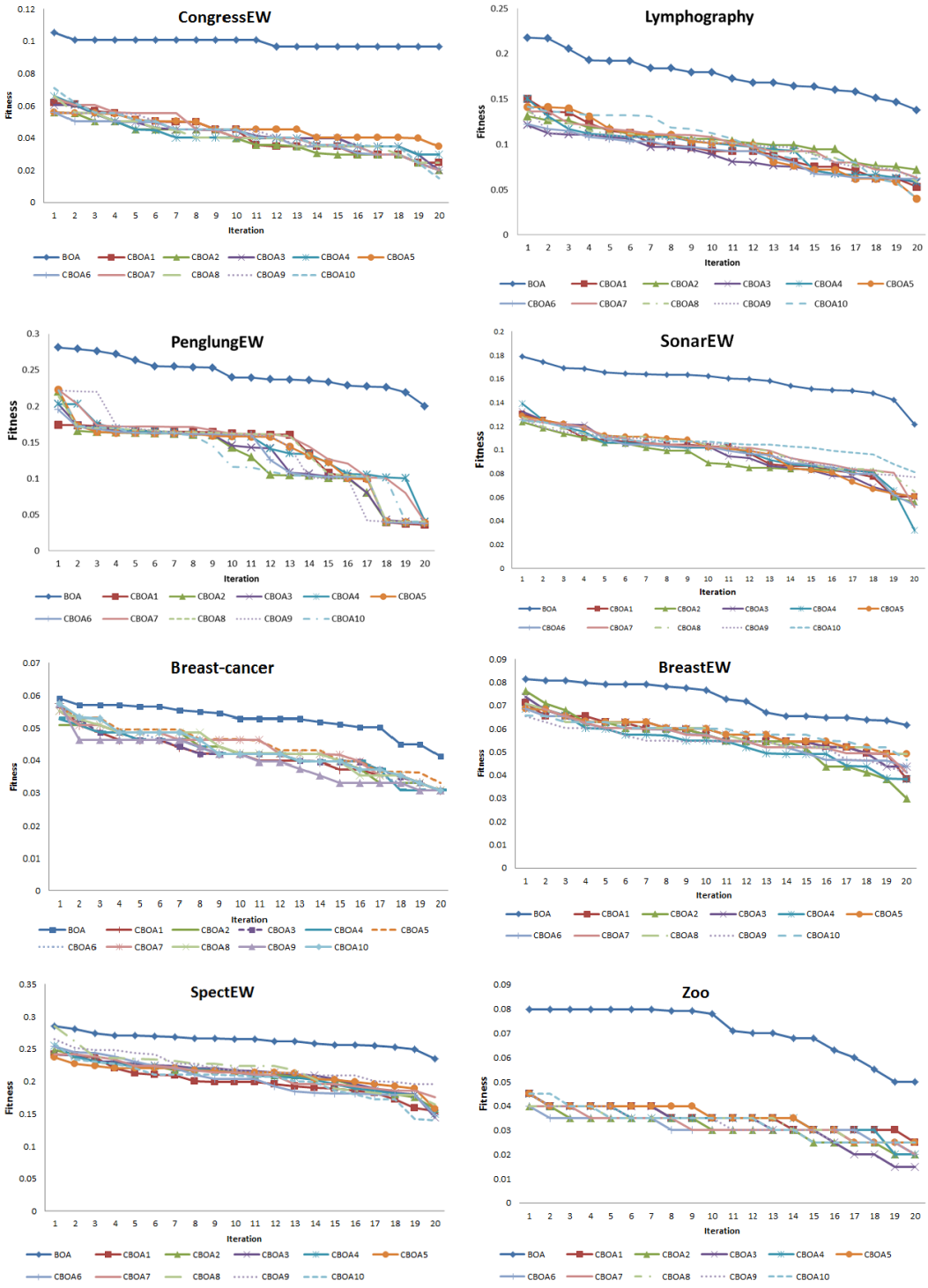

Fig. 1. The convergence curves of the standard BOA and the other modified CBOA

Table 3. Comparision between CBOA and other algorithms based on mean of fitness:

Dataset	ALO	BSO	GA	GWO	PSO	BBOA	CBOA
Breast-cancer	0.048	0.084	0.046	0.047	0.045	0.040	0.040
Breast EW	0.068	0.102	0.055	0.068	0.068	0.042	0.054
Clean1	0.160	0.177	0.134	0.147	0.150	0.113	0.082
Congress	0.069	0.149	0.063	0.073	0.082	0.045	0.039
HeartEW	0.228	0.307	0.216	0.228	0.226	0.180	0.150
IonosphereEW	0.145	0.149	0.109	0.136	0.124	0.096	0.059
KrvskpEW	0.108	0.242	0.083	0.094	0.086	0.054	0.511
Lymphography	0.219	0.309	0.187	0.241	0.214	0.139	0.087
PenglungEW	0.196	0.235	0.129	0.169	0.190	0.118	0.125
Semeion	0.035	0.044	0.029	0.040	0.039	0.026	0.126
SonarEW	0.158	0.209	0.128	0.143	0.138	0.086	0.086
SpectEW	0.216	0.252	0.192	0.219	0.219	0.160	0.199
Tic-Tac-Toe	0.249	0.342	0.243	0.252	0.253	0.205	0.223
WaveformEW	0.300	0.386	0.310	0.297	0.287	0.265	0.234
WineEW	0.054	0.139	0.051	0.060	0.055	0.023	0.045
Zoo	0.085	0.190	0.073	0.055	0.062	0.034	0.031

From the results showed above, we can draw the following remarks. Firstly, it can be noticed that the results of BBOA are similar to CBOA ones. This is because some of dataset, such as breast EW, have large difference between its numerical values while other datasets have string values which give less fitness values than other datasets when using KNN classifier. Secondly, the results shows that the use of Chaotic maps increased the random behavior (diversity) that helped to reach to a global minima and avoid getting stuck at a local minima which lead to faster convergence. Thirdly, the results of the proposed method proved that irrelevant and redundant features are ignored while selecting the important features only so the dimension of dataset is reduced.

5 Conclusion

In this paper, the feature selection problem is discussed and how chaotic maps and bio-inspired optimization algorithms can be used to improve a feature selection algorithm. A new chaotic-based butterfly optimization algorithm, CBOA, is proposed. In this algorithm 10 chaotic maps are investigated to improve the performance of the standard butterfly optimization algorithm (BOA). Sixteen dataset from UCI machine learning repository are used to evaluate the proposed CBOA algorithm. Also its results are compared with the BOA and other optimization and feature selection algorithms. The results showed that the CBOA be used to maximize classification accuracy while minimizing the classification error. The results also showed that integrating chaotic maps into the standard BOA can reduce the dimension of a dataset.

References

1. S. Ahmed, M. Mafarja, H. Faris and I. Aljarah. Feature selection using salp swarm algorithm with chaos. In Proceedings of the 2nd International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence (pp. 65-69), 2018.

2. S. Arora and S. Singh. Butterfly optimization algorithm: a novel approach for global optimization. Soft Computing, 23(3), 715-734, 2019.
3. S. Arora and P. Anand. Binary butterfly optimization approaches for feature selection. Expert Systems with Applications, 116, 147-160, 2019.
4. L. Brezocnik, I. Fister and V. Podgorelec. Swarm intelligence algorithms for feature selection: a review. Applied Sciences, 8(9), 1521, 2018.
5. A. Deshpande and M. Kumar. Artificial Intelligence for Big Data: Complete Guide to Automating Big Data Solutions Using Artificial Intelligence Techniques. Packt Publishing Ltd, 2018.
6. E. Emary, H. M. Zawbaa and A. E. Hassanien. Binary grey wolf optimization approaches for feature selection. Neurocomputing, 172, 371-381, 2016.
7. S. Harifi, M. Khalilian, J. Mohammadzadeh and S. Ebrahimnejad. Emperor Penguins Colony: a new metaheuristic algorithm for optimization. Evolutionary Intelligence, 12(2), 211-226, 2019.
8. A. E. Hegazy, M. A. Makhlouf and G. S. El-Tawel. Improved salp swarm algorithm for feature selection. Journal of King Saud University-Computer and Information Sciences, 2018.
9. G. Kaur and S. Arora. Chaotic whale optimization algorithm. Journal of Computational Design and Engineering, 5(3), 275-284, 2018.
10. Y. Li, T. Li and H. Liu. Recent advances in feature selection and its applications. Knowledge and Information Systems, 53(3), 551-577, 2017.
11. H. Liu, and L. Yu. Toward integrating feature selection algorithms for classification and clustering. IEEE Transactions on Knowledge & Data Engineering, (4), 491-502, 2005.
12. M. Mafarja, I. Aljarah, H. Faris, A. I. Hammouri, A. Z. Ala'M and S. Mirjalili. Binary grasshopper optimisation algorithm approaches for feature selection problems. Expert Systems with Applications, 117, 267-286, 2019.
13. M. M. Mafarja, D. Eleyan, I. Jaber, A. Hammouri and S. Mirjalili. Binary dragonfly algorithm for feature selection. In 2017 International Conference on New Trends in Computing Sciences (ICTCS) (pp. 12-17), 2017.
14. S. Mirjalili and A. H. Gandomi. Chaotic gravitational constants for the gravitational search algorithm. Applied soft computing, 53, 407-419, 2017.
15. H. Nematzadeh, R. Enayatifar, M. Mahmud and E. Akbari. Frequency based feature selection method using whale algorithm. Genomics, 2019.
16. G. I. Sayed, A. E. Hassanien and A. T. Azar. Feature selection via a novel chaotic crow search algorithm. Neural Computing and Applications, 31(1), 171-188, 2019.
17. X. Xu, H. Rong, M. Trovati, M. Liptrott and N. Bessis. CS-PSO: chaotic particle swarm optimization algorithm for solving combinatorial optimization problems. Soft Computing, 22(3), 783-795, 2018.
18. Tharwat, A., Gaber, T. and Hassanien, A.E., 2017. One-dimensional vs. two-dimensional based features: Plant identification approach. Journal of Applied Logic, 24, pp.15-31.
19. Tharwat, A., Gaber, T., Ibrahim, A. and Hassanien, A.E., 2017. Linear discriminant analysis: A detailed tutorial. AI communications, 30(2), pp.169-190.