An Efficient Moth Flame Optimization Algorithm using Chaotic Maps for Feature Selection in the Medical Applications

Ruba Abu Khurma, Ibrahim Aljarah and Ahmad Sharieh
King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan
RBA9150539@FGS.ju.edu.jo, {i.aljarah, sharieh}@ju.edu.jo

Keywords: Moth Flame Optimization Algorithm (MFO), Dimensionality Problem, Classification, Optimization, Feature Selection (FS), Chaotic Maps.

Abstract: In this paper, multiple variants of the Binary Moth Flame Optimization Algorithm (BMFO) based on chaotic maps are introduced and compared as search strategies in a wrapper feature selection framework. The main purpose of using chaotic maps is to enhance the initialization process of solutions in order to help the optimizer alleviate the local minima and globally converge towards the optimal solution. The proposed approaches are applied for the first time on FS problems. Dimensionality is a major problem that adversely impacts the learning process due to data-overfit and long learning time. Feature selection (FS) is a preprocessing stage in a data mining process to reduce the dimensionality of the dataset by eliminating the redundant and irrelevant noisy features. FS is formulated as an optimization problem. Thus, metaheuristic algorithms have been proposed to find promising near optimal solutions for this complex problem. MFO is one of the recent metaheuristic algorithms which has been efficiently used to solve various optimization problems in a wide range of applications. The proposed approaches have been tested on 23 medical datasets. The comparative results revealed that the chaotic BMFO (CBMFO) significantly increased the performance of the MFO algorithm and achieved competitive results when compared with other state-of-the-arts metaheuristic algorithms.

1 INTRODUCTION

In recent years, due to advances in data collection methods, vast amounts of data have been stored in data repositories. This is negatively reflected in the size of datasets either by increasing the number of instances and/or increasing the number of features.

Curse of dimensionality is a challenging problem that causes many negative consequences for the datamining tasks (i.e classification, clustering) (Khurma et al., 2020). It implies the existence of some features that are unuseful for the learning process such as the redundant and irrelevant features. Redundant features do not add any new information to the learning process because they can be inferred from other features. On the other hand, irrelative features are unrelated to the target class. These noisy features may mislead the learning algorithm when they are used in building the learning model. Furthermore, they adversely affect the learner's performance and generate poor quality models due to data-overfit. Increasing dimensionality also consumes more learning time and increases the demand for specialized hardware resources.

Feature selection (FS) is a primary preprocessing stage in a datamining process that has two conflicting objectives: producing a smaller version of the dataset by minimizing it's dimensions and simultaneously maximizing the learning performance (Faris et al., 2019; Al-Madi et al., 2018). This is accomplished by eliminating the noisy features (redundant or/and irrelevant) from original dataset without causing any loss of information. Formally speaking, for a dataset of N features, FS process selects n features from the original N features where $n \leq N$ without causing any degradation in the learner's performance. FS has the advantage that there is no generation of new feature combinations so the original meaning of the features is preserved. This is crucial for some fields which cares about the readability of a dataset such as bioinformatics and medicine.

The FS comprises four basic stages: subset generation, subset evaluation, checking a stopping criterion and validation stage (Dash and Liu, 1997). Subset generation is performed by a specific search technique (complete, heuristic) to generate candidate feature subsets. Subset evaluation determines the quality of a generated feature subset using a particular technique (filter or wrapper). FS is repeated until a specified condition is met (i.e maximum number of iterations). The last stage is to validate the feature subset by comparing it with the domain knowledge gathered from experts.

With regard to the evaluation approaches. Filters are considered rank based methods because they rely on a predefined threshold to evaluate a feature. They don't involve any learning algorithm but they use the intrinsic characteristics of the features. The absence of the learning process makes filters more time efficient. On the other hand, wrappers consider a learning algorithm to decide the quality of a feature subset. This contributes to better performance results but consumes more computational time.

FS search methods play a key role in controlling the complexity of the FS process. Brute force methods create the feature space by generating all the possible feature subsets from the original features set. Formally speaking, for n features, there are 2^n feature subsets that can be generated. FS process that involves a complete search procedure needs an exponential running time to exhaustively traverse all the generated feature subsets. This is computationally expensive and impractical with medium and large datasets making the FS an NP-Hard problem.

Metaheuristic algorithms are stochastic search methods that effectively generate promising solutions (near optimal) in a less time effort. Metaheuristic algorithms include a population based algorithms that initialize multiple solutions during the optimization process and update them in each iteration until the global solution is best approximated. Population based algorithms are further classified based on the source of inspiration into Evolutionary Algorithms (EA) and Swarm Intelligence algorithms (SI). SIs are inspired from the social intelligence that can be observed from the interactions between the groups of creatures such as flock of wolves, swarm of fish and colony of bee. A well known example for SI paradigm is the Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1997).

There are many metaheuristic algorithms that have been adopted as search engines in a wrapper framework and proved their effectiveness to limit the complexity of FS problem and provide acceptable solutions within a bounded time frame. These include the well-regarded algorithms such as GA (Huang and Wang, 2006) and PSO (Jain et al., 2018) and the recent metaheuristic algorithms such as Whale Optimization Algorithm (WOA) (Sayed et al., 2018a), Multi-Verse Optimization algorithm (MVO) (Ewees et al., 2019) and Salp Swarm Algorithm (SSA) (Sayed et al., 2018b).

Moth Flame Optimization algorithm (MFO) is a recently developed SI algorithm that mimics the navigation method of moths at night. MFO proved it's effectiveness in optimizing various complex optimization problems in different fields (Mirjalili, 2015). Like any population based paradigm, the MFO optimizer has two conflicting milestones in the optimization process called exploration and exploitation. In the exploration phase, the search space is searched to identify promising regions where the best solution may exist while in the exploitation phase, the found solutions are further improved. The main target in the optimization process is to maintain a balance between exploration and exploitation and smoothly alternates between them. Too much exploration loses the optimal solution while too much exploitation causes stagnation in a local minima.

The MFO algorithm has many advantages that motivated us to select it as a search algorithm in wrapper frameworks: First, the search engine of the MFO methodology relies on a spiral position update procedure that can change the positions of moths in a manner that achieves a promising trade off between the exploration and exploitation and adaptively converges toward the optimal solution. Second, it had been used to solve many problems with unknown and constrained search spaces (Mirjalili, 2015). Third, MFO algorithm is equipped with adaptive parameters that increase exploration phase in the early stages of the optimization process and increase the exploitation in the final stages. Fourth, MFO always maintains the best solutions obtained and reduces their numbers in each iteration so that it gets one global best solution in the final stage. Despite the promising characteristics of the MFO algorithm, the search agents still have a chance of being entrapped in local minima.

Chaotic maps are commonly used operators that have been used to replace random components and support convergence of multiple metaheuristic algorithms. In 2007, Chaotic maps were hybridized with PSO algorithm to develop a feasible approach for FS and classification of the hyper spectral image data (YANG et al., 2007). In 2011, a hybrid model for FS and classification of large-dimensional microarray data sets was developed by using correlation-based FS (CFS) and the Taguchi Chaotic Binary PSO (TCBPSO) (Chuang et al., 2011). The same author, in the same year, designed a chaotic BPSO (CBPSO) based on two kinds of chaotic maps called logistic maps and tent maps that were integrated with BPSO to determine the inertia weight of the BPSO to enhance the FS process. In 2017, chaotic maps were integrated with the MVO algorithm in the the context of FS to cope with slow convergence and local minima problems. The used chaotic maps were Tent, Logistic, Singer, Sinusoidal and Piecewise (Ewees et al., 2019). An improved SSA was developed in 2018 to handle the FS problem. Chaos theory was integrated into the algorithm to replace the random variables with chaotic variables. The developed approach was able to efficiently mitigate the local minima and low convergence problems (Sayed et al., 2018b). In 2018, a new wrapper FS approach was developed based on WOA and chaotic theory named CWOA in the medical application (Sayed et al., 2018a).

According to No-free-Lunch theorem (Wolpert et al., 1997), there is no metaheuristic algorithm that has the same performance with all optimization problems. Thus, the doors are still opened to propose new modifications to enhance metaheuristic algorithms. In in this paper, chaotic maps have been proposed for the first time to enhance the MFO ability in the FS binary space. The main contribution is the development of four binary variants of the MFO algorithm through the use of four different chaotic maps. The main purpose is to improve the initialization strategy of the standard MFO algorithm by replacing the uniform random distribution with chaotic equations. The generated CBMFO variants are studied and compared to analyze the influence of the adopted operators on the BMFO performance while optimizing the feature space in the domain of disease diagnosis.

The paper is organized as follows: Section 2 gives an overview of the standard and binary MFO algorithm. Section 3 discusses the proposed approach. In Section 4, the experimental results are analyzed. Finally, in Section 5, conclusions and future works are outlined.

2 METHODOLOGY

2.1 Overview of MFO

Moth Flame Optimization (MFO) is one of the recent SI algorithms which was developed in (Mirjalili, 2015). The MFO methodology was inspired by the natural movements of moths at night. The moths are enabled to move long distances in straight line by maintaining the same angle with respect to moon light. This navigation method is called transfer orientation. However, transfer orientation has the shortcoming that nearby light sources such as candle fool the moths and force them to follow a spiral path until they eventually die.

Eq.1 describes mathematically the natural spiral motion of moths around a flame where M_i represents the i_{th} moth, Fj represents the j_{th} flame, and S is the spiral function. Eq.2 formulates the spiral motion using a standard logarithmic function where D_i is the distance between the i_{th} moth and the j_{th} flame as described in Eq.3, b is a constant value for determining the shape of the logarithmic spiral, and t is a random number in the range [-1, 1]. The parameter $t = -1$ indicates the closest position of a moth to a flame where $t = 1$ indicates the farthest position between a moth and a flame. To achieve more exploitation in the search space the t parameter is considered in the range [r,1] where r is linearly decreased over the course of iterations from -1 to -2. Eq.4 shows gradual decrements of the number of flames over the course of iterations where l is the current number of iteration, N is the maximum number of flames and T is the maximum number of iterations. Algorithm 1 shows the entire pseudo code of the MFO algorithm. The steps of the MFO optimization starts by initializing the positions of moths. Each moth updates it's position with respect to a flame based on a spiral equation. The t and r parameters are linearly decreased over iterations to emphasize exploitation. In each iteration, the flames list is updated and then sorted based on the fitness values of flames. Consequently, the moths update their positions with respect to their corresponding flames. To increase the chance of reaching to the global best solution, the number of flames is decreased with respect to the iteration number. Thus, a given moth updates it's position using only one of the flames.

$$Mi = S(Mi, Fj) \tag{1}$$

$$S(Mi, Fj) = Di.e^{bt}.cos(2\pi) + Fj \tag{2}$$

$$Di = |Mi - Fj| \tag{3}$$

$$FlameNo = round(N - l * (N-1)/T) \tag{4}$$

2.2 Binary MFO (BMFO)

The original MFO algorithm was developed to solve global optimization problems where the components of a solution are real values. All what is required is to check that the upper and lower bounds are not exceeded during the initialization and update procedures. In the binary optimization problems, the case is different because the solutions have only binary elements (i.e either "0" or "1"). This restriction should be not violated while the moths change their positions in the binary search space. For achieving this purpose, some operators have to be integrated with MFO algorithm to allow it optimize in the binary search space.

Algorithm 1: Pseudo-code of the MFO algorithm.

Input:$Max\_iteration$, n (number of moths), d (number of dimensions)
Output:Approximated global solution
Initialize the position of moths

```
while l ≤ Max_iteration do
  Update flame no using Eq.4
  OM = FitnessFunction(M);
  if l == 1 then
    F = sort(M);
    OF = sort(OM);
  else
    F = sort(M_{l-1}, M_l);
    OF = sort(OM_{l-1}, OM_l);
  end if
  for i = 1: n do
    for j = 1: d do
      Update r and t;
      Calculate D using Eq.3 with respect to the corresponding moth;
      Update M(i, j) using Eqs.1 and Eqs.2 with respect to the corresponding moth;
    end for
  end for
  l = l + 1;
end while
```

The most common binary operator used for converting continuous optimizers into binary is the transfer function (TF) (Mirjalili and Lewis, 2013). The main reason for using TFs is that they are easy to implement without impacting the merit of the algorithm. In this paper, the used TF is the sigmoid function which was used originally in (Kennedy and Eberhart, 1997) to generate the binary PSO (BPSO). In the MFO algorithm, the first term of Eq.2 represents the step vector which is redefined in Eq.5. The function of the sigmoid is to determine a probability value in the range [0,1] for each element of the solution. Eq.6 shows the formula of the sigmoid function. Each moth updates it's position based on Eq.7 which takes the output of Eq.6 as it's input.

$$\Delta M = Di \,.\, e^{bt} \,.\, cos(2\pi) \tag{5}$$

$$TF(\Delta M_t) = 1/(1 + e^{\Delta M_t}) \tag{6}$$

$$M_i^d(t+1) = \begin{cases} 0, & \text{if } rand < TF(\Delta M_{t+1}) \\ 1, & \text{if } rand \geqslant TF(\Delta M_{t+1}) \end{cases} \tag{7}$$

2.3 Feature Selection based on BMFO

The FS problem must be represented correctly in order to facilitate the optimizer task in the feature space. There are two key issues to achieve this: properly representing the solution and evaluating it using a specific fitness function. The solution to the FS problem is represented as a binary vector where the length of the vector is equal to the dimensions of the data set. Thus, each element of the solution represents a feature that takes two values, either "1" if the feature is selected or "0" if the feature is not selected. The evaluation for the solution in the FS problem depends on combining two main objectives of the FS problem in one formula. These objectives are maximizing the performance of the classifier and simultaneously minimizing the number of dimensions in the dataset. Eq.8 formulates the FS problem where $\alpha\gamma_R(D)$ is the error rate of the classification produced by a classifier, $|R|$ is the number of selected features in the reduced dataset, and $|C|$ is the number of features in the original dataset, and $\alpha \in [0,1]$, $\beta = (1-\alpha)$ are two parameters for representing the importance of classification performance and length of feature subset based on recommendations (Mafarja and Mirjalili, 2018).

$$Fitness = \alpha\gamma_R(D) + \beta\frac{|R|}{|C|} \tag{8}$$

3 THE PROPOSED APPROACH

In this section, the proposed chaotic BMFO (CBMFO) approaches are presented.

3.1 Chaotic Maps

Chaotic maps are mathematical systems that describe a dynamic deterministic process which has a high sensitivity to initial conditions (dos Santos Coelho and Mariani, 2008). Even though the process is deterministic but the outcomes are unpredictable. Chaotic maps have proved their effectiveness in improving the performance of metaheuristic algorithms when they are integrated with them for solving a specific optimization problem. They have been applied to replace the random components of the metaheuristic algorithm to provide a higher convergence capability and alleviate the local minima problem by getting closer to the position of the optimal solution. In this paper, the impact of chaotic maps are studied on MFO optimizer in the FS search space. Four different chaotic maps have been selected called circle, logistic, piecewise and tent as formulated in equations Eq.9, Eq.10, Eq.11, Eq.12, respectively. Fig 1 visually presents these chaotic maps. The developed chaotic MFO variants are called CBMFO1, CBMFO2, CBMFO3 and CBMFO4 respectively. In these variants, the chaotic maps are used to initiate the positions of moths instead of using the uniform random distribution. The idea is to improve the initialization process of the MFO and reduce the uncertainty of the optimizer. This is done by replacing the initial random positions of moths generated by uniform random distribution with positions generated by chaotic maps.

$$Circle : x_i + 1 = mod(X_i + b \; - \; (a2\pi) \; sin(2\pi \, x_i), 1) \tag{9}$$

where: $a = 0.5, b = 0.2$.

$$Logistic : x_i + 1 = ax_i(1 \; - \; x_i) \tag{10}$$

where: $a = 4$.

$$Piecewise : x_{i+1} = \begin{cases} \frac{x_i}{p}, & \text{if } 0 \leq x_i < p \\ \frac{x_i - p}{0.5 - p}, & \text{if } p \leq x_i < 0.5 \\ \frac{1 - p - x_i}{0.5 - p}, & \text{if } 0.5 \leq x_i < 1 - p \\ \frac{1 - x_i}{p}, & \text{if } 1 - p \leq x_i < p \end{cases} \tag{11}$$

where: $p = 0.4$.

$$Tent : x_{i+1} = \begin{cases} \frac{x_i}{0.7}, & \text{if } x_i < 7 \\ \frac{10}{3}(1 - x_i), & \text{if } x_i \geqslant 0.7 \end{cases} \tag{12}$$

4 EXPERIMENTAL RESULTS

In this paper, 23 medical datasets were downloaded from UCI (Asuncion and Newman, 2007), Keel (Alcalá-Fdez et al., 2011) and Kaggle (Goldbloom et al., 2017) data repositories to evaluate the proposed wrapper approaches. Table 1 lists these datasets along with their number of features, instances and classes. All the datasets are characterized by balanced distribution of their classes. To validate the proposed approaches, three well known metaheuristic algorithms were used for comparison purposes: BGWO, BCS and BBA. Their parameter settings are as follows: the value of α parameter in GWO is in [2,0]. For BA, the Qmin Frequency minimum value is 0, Qmax Frequency maximum is 2, A Loudness value is 0.5 and r Pulse rate is 0.5. For CS, the *pa* value is 0.25 and the value of β parameter is 3/2.

All the experiments were executed on a personal machine with AMD Athlon Dual-Core QL-60 CPU

Table 1: Description of the used datasets.

NO	Dataset Name	No features	No instances	No classes
1	Breast Cancer Wisconsin (Diagnostic)	30	569	2
2	Breast Cancer Wisconsin (Original)	9	699	2
3	Breast Cancer Wisconsin (Prognostic)	33	194	2
4	Breast Cancer Coimbra	9	115	2
5	BreastEW	30	596	2
6	Diabetic Retinopathy Debrecen	19	1151	2
7	Dermatology	34	366	6
8	ILPD (Indian Liver Patient Dataset)	10	583	2
9	Lymphography	18	148	4
10	Parkinsons	22	194	2
11	Parkinson's Disease Classification	753	755	2
12	SPECT	22	267	2
13	Cleveland	13	297	5
14	HeartEW	13	270	2
15	Hepatitis	18	79	2
16	South African Heart (SA Heart)	9	461	2
17	SPECTF Heart	43	266	2
18	Thyroid Disease (thyroid 0387)	21	7200	3
19	Heart	13	302	5
20	Pima-indians-diabetes	9	768	2
21	Leukemia	7129	72	2
22	Colon	2000	62	2
23	Prostate_GE	5966	102	2

at 1.90 GHz and memory of 2 GB running Windows7 Ultimate 64 bit operating system. The optimization algorithms are all implemented in Python in the EvoloPy-FS framework (Khurma et al., 2020). The maximum number of iterations and the population size were set to 100 and 10 respectively. In this work, the K-NN classifier (where K = 5 (Mafarja and Mirjalili, 2018)) is used to evaluate individuals in the wrapper FS approach. Each dataset is randomly divided in two parts; 80% for training and 20% for testing. To obtain statistically significant results, this division was repeated 30 independent times. Therefore, the final statistical results were obtained over 30 independent runs. The α and β parameters in the fitness equation is set to 0.99 and 0.01, respectively (Emary et al., 2016). The used evaluation measures are fitness values, classification accuracy, number of selected features and CPU time.

Inspecting the results in Table 2, it seems clearly that the usage of chaotic operators have improved the performance of the BMFO algorithm in terms of the classification accuracy. By comparing CBMFO1, CBMFO2, CBMFO3 and CBMFO4 it appears that the CBMFO2 and CBMFO4 achieved promising results. Based on the ranking results, the Tent-based CBMFO4 achieved the highest classification performance in five out of 23 datasets then Logistic-based CBMFO2 which which was superior across four datasets. On the other hand, Circle-based CBMFO1 and Piecewise-based CBMFO3 outperformed other algorithms only across two datasets. By combining all the chaotic variants of the BMFO algorithm and comparing their results with the standard BMFO algorithm, it can be seen that the BMFO chaotic variants achieved an improvement over the standard BMFO equals 70%. Moreover, the standard BMFO was not better than any of the chaotic approaches on any

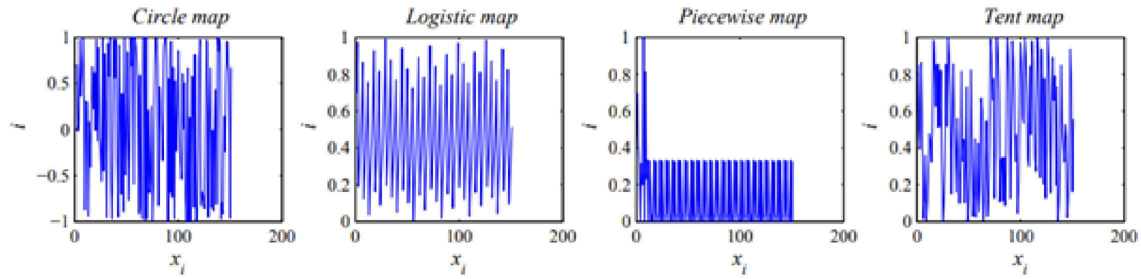

Figure 1: Visualized chaotic maps.

Table 2: Average Classification Accuracy from 30 Runs for All Approaches.

NO	Dataset Name	BMFO	CBMFO1	CBMFO2	CBMFO3	CBMFO4	BGWO	BCS	BBA
1	Breast Cancer Wisconsin (Diagnostic)	0.909	0.906	0.909	0.907	**0.910**	0.901	0.890	0.870
2	Breast Cancer Wisconsin (Original)	0.963	**0.970**	0.955	0.967	0.966	0.871	0.812	0.698
3	Breast Cancer Wisconsin (Prognostic)	0.583	0.588	**0.600**	0.596	0.575	0.570	0.533	0.500
4	Breast Cancer Coimbra	0.907	**0.908**	0.893	0.900	0.873	0.760	0.648	0.561
5	BreastEW	0.932	0.923	**0.954**	0.942	0.938	0.835	0.794	0.828
6	Diabetic Retinopathy Debrecen	0.542	0.538	**0.551**	0.536	0.548	0.527	0.500	0.500
7	Dermatology	0.820	0.787	0.813	0.822	0.804	0.829	**0.833**	0.753
8	ILPD (Indian Liver Patient Dataset)	0.714	0.714	0.714	0.714	0.714	**0.742**	0.733	0.661
9	Lymphography	0.772	0.744	0.761	**0.800**	0.750	0.723	0.691	0.612
10	Parkinsons	**0.754**	0.750	**0.754**	**0.754**	0.750	0.667	0.719	0.676
11	Parkinsonś Disease Classification	0.811	0.809	0.811	0.798	**0.816**	0.789	0.763	0.671
12	SPECT	0.658	0.652	**0.659**	0.642	0.642	0.591	0.590	0.502
13	Cleveland	0.536	0.533	0.528	0.533	0.536	0.508	**0.613**	0.529
14	HeartEW	0.942	0.936	0.930	0.936	**0.949**	0.870	0.855	0.832
15	Hepatitis	**0.750**	**0.750**	**0.750**	**0.750**	**0.750**	0.673	0.641	0.561
16	South African Heart (SA Heart)	0.684	0.683	0.693	**0.697**	0.684	0.630	0.616	0.593
17	SPECTF Heart	0.700	0.682	0.709	0.688	0.700	0.673	**0.743**	0.618
18	Thyroid Disease (thyroid0387)	**0.981**	0.980	**0.981**	**0.981**	**0.981**	0.943	0.934	0.902
19	Heart	0.752	0.767	0.767	0.758	**0.776**	0.727	0.724	0.719
20	Pima-indians-diabetes	0.807	0.807	0.807	0.807	0.807	**0.813**	0.800	0.790
21	Leukemia	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.987	0.950	0.885
22	Colon	0.656	0.644	**0.667**	0.656	**0.667**	0.630	0.603	0.620
23	Prostate_GE	0.500	0.501	0.511	0.503	**0.519**	0.503	0.504	0.500
	Ranking ($W\|T\|L$)	0|4|19	2|2|19	4|5|14	2|4|17	**5|4|14**	2|0|21	3|0|20	0|0|23

Table 3: Average Fitness Values from 30 Runs for All Approaches.

NO	Dataset Name	BMFO	CBMFO1	CBMFO2	CBMFO3	CBMFO4	BGWO	BCS	BBA
1	Breast Cancer Wisconsin (Diagnostic)	0.095	0.098	0.095	0.096	**0.094**	0.102	0.112	0.107
2	Breast Cancer Wisconsin (Original)	0.043	**0.036**	0.051	0.040	0.041	0.130	0.188	0.190
3	Breast Cancer Wisconsin (Prognostic)	0.415	0.411	**0.399**	0.403	0.424	0.433	0.468	0.499
4	Breast Cancer Coimbra	0.097	**0.096**	0.110	0.103	0.130	0.240	0.350	0.367
5	BreastEW	0.072	0.081	**0.051**	0.062	0.067	0.163	0.204	0.128
6	Diabetic Retinopathy Debrecen	0.457	0.462	**0.448**	0.464	0.452	0.474	0.498	0.495
7	Dermatology	0.184	0.216	0.190	0.181	0.199	0.172	**0.168**	0.180
8	ILPD (Indian Liver Patient Dataset)	0.290	0.290	0.290	0.290	0.290	**0.263**	0.271	0.321
9	Lymphography	0.231	0.258	0.242	**0.204**	0.253	0.209	0.222	0.220
10	Parkinsons	**0.248**	0.252	**0.248**	**0.248**	0.252	0.332	0.280	0.273
11	Parkinsonś Disease Classification	0.192	0.194	0.192	0.203	**0.187**	0.223	0.247	0.267
12	SPECT	0.344	0.351	**0.343**	0.359	0.359	0.411	0.411	0.417
13	Cleveland	0.462	0.465	0.470	0.464	0.462	0.489	**0.386**	0.427
14	HeartEW	0.063	0.068	0.075	0.069	**0.057**	0.082	0.080	0.079
15	Hepatitis	0.253	**0.252**	**0.252**	0.253	**0.252**	0.329	0.359	0.364
16	South African Heart (SA Heart)	0.317	0.318	0.308	**0.304**	0.316	0.371	0.384	0.367
17	SPECTF Heart	0.302	0.319	0.293	0.314	0.302	0.326	**0.257**	0.340
18	Thyroid Disease (thyroid 0387)	**0.019**	0.020	**0.019**	**0.019**	**0.019**	0.059	0.066	0.048
19	Heart	0.251	0.236	0.236	0.245	**0.227**	0.273	0.276	0.264
20	Pima-indians-diabetes	0.200	0.200	0.200	0.200	0.200	**0.195**	0.206	0.199
21	Leukemia	**0.005**	**0.005**	**0.005**	**0.005**	**0.005**	0.033	0.066	0.042
22	Colon	0.344	0.356	0.333	0.344	0.333	0.352	0.377	**0.329**
23	Prostate_GE	**0.450**	0.499	0.489	0.498	0.481	0.497	0.496	0.500
	Ranking ($W\|T\|L$)	1|3|19	2|2|19	**4|4|15**	2|3|18	4|3|16	2|0|21	3|0|20	1|0|22

dataset. By comparing all BMFO-based approaches with other metaheuristic wrapper approaches, the BMFO approach was superior in 78% of the data sets. The improvement in the classification results of the CBMFO can be explained that the chaotic operators have effectively improved the initialization procedure by replacing the uniform random distribution with chaotic functions that have a greater sensitivity to initial conditions. This led to a greater exploration for the search space and better alleviating for the local minima problem. Furthermore, the optimizer was able to achieve a better trade-off between the two conflicting milestones: exploration and exploitation. This was realized by an improvement in the convergence behaviour of the MFO in the binary feature space.

The overall performance of the proposed approaches can be better realized when analyzing the fitness values results that combine both classification accuracy and the selection ratio. From the results in Table 3, it can be seen that CBMFO approaches got better results than other methods in twelve out of 23 datasets that are close to half of the datasets.

Table 4: Average Number of Selected Features from 30 Runs for All Approaches.

NO	Dataset Name	BMFO	CBMFO1	CBMFO2	CBMFO3	CBMFO4	BGWO	BCS	BBA
1	Breast Cancer Wisconsin (Diagnostic)	14.233	14.867	14.233	**13.500**	14.500	14.010	16.177	14.010
2	Breast Cancer Wisconsin (Original)	**5.500**	5.800	5.967	5.867	5.867	7.533	7.000	6.467
3	Breast Cancer Wisconsin (Prognostic)	15.967	15.867	16.033	16.100	16.367	18.233	16.400	**15.400**
4	Breast Cancer Coimbra	3.667	**3.367**	3.633	3.467	3.833	6.077	6.844	4.810
5	BreastEW	13.833	14.100	14.900	14.300	14.433	17.338	13.505	**11.505**
6	Diabetic Retinopathy Debrecen	7.567	7.667	7.267	7.467	7.533	10.563	8.290	**6.763**
7	Dermatology	18.300	17.733	17.567	18.233	18.200	21.000	18.699	**14.401**
8	ILPD (Indian Liver Patient Dataset)	**4.000**	**4.000**	**4.000**	**4.000**	**4.000**	6.567	7.333	5.300
9	Lymphography	9.433	9.733	10.167	10.133	9.233	10.230	9.300	**7.330**
10	Parkinsons	10.633	9.700	10.733	10.167	10.000	11.258	10.687	**7.854**
11	Parkinsonś Disease Classification	373.733	345.967	346.600	**225.833**	347.000	421.211	383.508	363.711
12	SPECT	11.433	12.167	11.700	11.500	11.367	13.106	10.779	**8.210**
13	Cleveland	6.200	6.400	5.967	**5.633**	5.833	8.900	7.632	6.346
14	HeartEW	7.400	7.067	7.633	7.567	7.400	6.700	5.867	**4.700**
15	Hepatitis	9.100	**8.167**	8.667	9.400	8.233	13.496	11.123	9.511
16	South African Heart (SA Heart)	3.700	3.467	3.400	**3.067**	3.333	6.333	4.221	5.100
17	SPECTF Heart	20.600	19.000	20.400	20.567	20.033	26.355	18.941	**14.891**
18	Thyroid Disease (thyroid 0387)	8.523	8.022	8.100	9.1061	8.087	11.490	9.990	**6.990**
19	Heart	6.367	6.100	6.200	6.167	**6.033**	8.600	7.367	6.067
20	Pima-indians-diabetes	6.500	6.333	6.400	6.367	6.700	6.946	8.059	**6.067**
21	Leukemia	3562.692	3536.442	3822.169	3671.113	3652.963	4052.986	3944.353	**3249.220**
22	Colon	994.157	989.529	1000.569	998.457	992.417	1147.978	1101.712	**933.378**
23	Prostate_GE	2988.113	2960.663	2998.510	2991.336	**2958.447**	2994.691	2986.400	2985.631
	Ranking ($W\|T\|L$)	1|1|21	2|1|20	0|1|22	4|1|18	2|1|20	0|0|23	0|0|23	**13|0|10**

Table 5: Average Computational Time from 30 Runs for All Approaches.

NO	Dataset Name	BMFO	CBMFO1	CBMFO2	CBMFO3	CBMFO4	BGWO	BCS	BBA
1	Breast Cancer Wisconsin (Diagnostic)	123.147	159.580	161.999	162.096	114.157	98.772	68.757	**52.269**
2	Breast Cancer Wisconsin (Original)	121.202	108.772	122.246	117.109	112.431	83.596	52.370	**45.798**
3	Breast Cancer Wisconsin (Prognostic)	85.989	87.877	85.364	95.580	55.849	49.632	27.458	**15.896**
4	Breast Cancer Coimbra	62.392	43.926	39.175	46.251	64.872	31.715	25.790	**18.896**
5	BreastEW	165.641	161.151	150.501	163.598	153.825	103.559	**45.789**	46.796
6	Diabetic Retinopathy Debrecen	268.268	291.092	241.833	280.222	271.357	211.789	169.524	**144.891**
7	Dermatology	86.433	70.400	66.805	111.632	107.998	80.632	53.969	**41.598**
8	ILPD (Indian Liver Patient Dataset)	103.550	76.286	58.309	80.301	59.457	84.633	62.753	**55.789**
9	Lymphography	68.208	40.978	47.599	53.355	41.869	65.890	34.529	**32.741**
10	Parkinsons	82.594	65.517	50.102	50.137	47.086	81.522	43.960	**41.875**
11	Parkinsonś Disease Classification	956.432	734.329	768.159	946.304	914.159	884.631	436.590	**420.637**
12	SPECT	99.688	57.469	68.748	74.292	58.382	75.457	28.693	**27.551**
13	Cleveland	55.173	66.702	54.722	54.976	75.738	61.896	22.963	**31.893**
14	HeartEW	87.035	85.627	78.856	83.678	83.966	88.460	**33.569**	37.560
15	Hepatitis	34.603	48.176	36.794	38.536	50.800	39.772	26.896	**13.598**
16	South African Heart (SA Heart)	82.592	76.325	86.452	88.409	81.023	95.569	**37.510**	39.632
17	SPECTF Heart	93.671	97.836	84.029	91.410	104.362	91.559	**38.569**	49.110
18	Thyroid Disease (thyroid 0387)	5021.887	4335.116	4896.563	5781.663	4891.263	3789.224	1598.789	**1269.633**
19	Heart	56.492	51.446	67.667	73.856	59.964	71.583	**35.094**	43.669
20	Pima-indians-diabetes	179.240	118.969	137.138	166.997	141.529	145.225	33.789	**31.115**
21	Leukemia	1416.448	1458.963	1215.662	1589.445	1460.781	1490.236	849.740	**800.559**
22	Colon	583.960	5689.021	561.896	5981.024	5713.693	371.777	220.631	**197.855**
23	Prostate_GE	1257.102	1236.896	1234.112	1456.932	1008.963	1149.115	988.763	**950.111**
	Ranking ($W\|T\|L$)	0|0|23	0|0|23	0|0|23	0|0|23	0|0|23	0|0|23	5|0|18	**18|0|5**

According to Table 4, it seems clear that the BBA algorithm outperformed other wrapper approaches across 57% of datasets. By comparing the CBMFO approaches to each other in terms of selection ratio, the Circle-based CBMFO1 outperformed other chaotic approaches across 52% of datasets and then came the Piecewise-based CBMFO3 and Tent-based CBMFO4 that outperformed other chaotic variants evenly over 17% of datasets and finally came Logistic-based CBMFO2 that outperformed other chaotic variants on 1% of datasets.

By looking at Table 5, it seems clear from the ranking results that the BBA algorithm outperformed other wrapper approaches across 78% of datasets while the BCS algorithm outperformed others across 22% of the datasets. It is seen that the BMFO approaches were not able to outperform these approaches on any of the datasets. By comparing the CBMFO approaches to each other in terms of running time, the Logistic-based CBMFO2 and Circle-based CBMFO1 achieved better results compared with other chaotic variants. The CBMFO2 consumed the shortest optimization time to find the near optimal feature subset across ten datasets and the CBMFO1 consumed the shortest optimization time across eight datasets. However, the BBA algorithm had much superior performance in general. On the other hand, CBMFO4 based on Tent map achieved the shortest CPU time for optimization across four datasets. For the CBMFO3, it is clear that this approach didn't outperform any other approaches in terms of running time.

5 CONCLUSIONS AND FUTURE WORK

In this paper, multiple binary versions based on MFO algorithm have been proposed to address the FS problem. The chaotic maps have been adopted to enhance the BMFO performance in the feature space. Specifically, the chaotic maps were used to enhance the initialization of moths and promote the convergence behaviour of the MFO algorithm. Therefore, the MFO can alleviate stagnation in local minima and reach to a closer place near the global optima. To evaluate the proposed approaches, 23 medical datasets were used from well regarded data repositories including UCI, Keel and Kaggle. The comparative results showed that the chaotic operators have enhanced the performance of the standard BMFO when used to optimize the feature search space. For the future, the research line of metaheuristic based wrapper methods can be continued by proposing new modification strategies and adopting other metahueristic algorithms to examine feature space.

REFERENCES

Al-Madi, N., Faris, H., and Abukhurma, R. (2018). Cost-sensitive genetic programming for churn prediction and identification of the influencing factors in telecommunication market. *International Journal of Advanced Science and Technology*, pages 13–28.

Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., and Herrera, F. (2011). Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17.

Asuncion, A. and Newman, D. (2007). Uci machine learning repository.

Chuang, L.-Y., Yang, C.-S., Wu, K.-C., and Yang, C.-H. (2011). Gene selection and classification using taguchi chaotic binary particle swarm optimization. *Expert Systems with Applications*, 38(10):13367–13377.

Dash, M. and Liu, H. (1997). Feature selection for classification. *Intelligent data analysis*, 1(1-4):131–156.

dos Santos Coelho, L. and Mariani, V. C. (2008). Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization. *Expert Systems with Applications*, 34(3):1905–1913.

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

Ewees, A. A., El Aziz, M. A., and Hassanien, A. E. (2019). Chaotic multi-verse optimizer-based feature selection. *Neural Computing and Applications*, 31(4):991–1006.

Faris, H., Abukhurma, R., Almanaseer, W., Saadeh, M., Mora, A. M., Castillo, P. A., and Aljarah, I. (2019). Improving financial bankruptcy prediction in a highly imbalanced class distribution using oversampling and ensemble learning: a case from the spanish market. *Progress in Artificial Intelligence*, pages 1–23.

Goldbloom, A., Hamner, B., Moser, J., and Cukierski, M. (2017). Kaggle: your homr for data science.

Huang, C.-L. and Wang, C.-J. (2006). A ga-based feature selection and parameters optimizationfor support vector machines. *Expert Systems with applications*, 31(2):231–240.

Jain, I., Jain, V. K., and Jain, R. (2018). Correlation feature selection based improved-binary particle swarm optimization for gene selection and cancer classification. *Applied Soft Computing*, 62:203–215.

Kennedy, J. and Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. In *1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation*, volume 5, pages 4104–4108. IEEE.

Khurma, R. A., Aljarah, I., Sharieh, A., and Mirjalili, S. (2020). Evolopy-fs: An open-source nature-inspired optimization framework in python for feature selection. In *Evolutionary Machine Learning Techniques*, pages 131–173. Springer.

Mafarja, M. and Mirjalili, S. (2018). Whale optimization approaches for wrapper feature selection. *Applied Soft Computing*, 62:441–453.

Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, 89:228–249.

Mirjalili, S. and Lewis, A. (2013). S-shaped versus v-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, 9:1–14.

Sayed, G. I., Darwish, A., and Hassanien, A. E. (2018a). A new chaotic whale optimization algorithm for features selection. *Journal of classification*, 35(2):300–344.

Sayed, G. I., Khoriba, G., and Haggag, M. H. (2018b). A novel chaotic salp swarm algorithm for global optimization and feature selection. *Applied Intelligence*, 48(10):3462–3481.

Wolpert, D. H., Macready, W. G., et al. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

YANG, H.-c., ZHANG, S.-b., DENG, K.-z., and DU, P.-j. (2007). Research into a feature selection method for hyperspectral imagery using pso and svm. *Journal of China University of Mining and Technology*, 17(4):473–478.