Future Computing and Informatics Journal

Volume 7
Issue 2 (2022) *Volume 7 Issue 2*

Article 1

2022

Crow search algorithm with time varying flight length strategies for feature selection

Mohammed Abdullahi
Ahmadu Bello University, Zaria, Nigeria, abdullahilwafu@abu.edu.ng

Abdulhameed Adamu
College of Education, Akwanga, Nigeria, abdulhameedadamu@gmail.com

Ibrahim Hayatu Hassan
Ahmadu Bello University, Zaria, Nigeria, ihhassan@abu.edu.ng

Crow search algorithm with time varying flight length strategies for feature selection

Cover Page Footnote

Conflict of Interest: On behalf of all authors, the corresponding author states that there is no conflict of interest.

Future Computing and Informatics Journal

Homepage: https://digitalcommons.aaru.edu.jo/fcij/

doi: **http://Doi.org/10.54623/fue.fcij.7.2.1**

CROW SEARCH ALGORITHM WITH TIME VARYING FLIGHT LENGTH STRATEGIES FOR FEATURE SELECTION

Mohammed Abdullahi[1, *, a], Abdulhameed Adamu[2, b], Ibrahim Hayatu Hassan[1, c], Abdulrazaq Abdulrahim[1, d]

[1] Department of Computer Science, Ahmadu Bello University, Zaria, Nigeria
[2] Department of Computer Science, College of Education, Akwanga, Nigeria

[a] abdullahilwafu@abu.edu.ng, [b] abdulhameedadamu@gmail.com, [c] ihhassan@abu.edu.ng, [d] aabdulrahim@abu.edu.ng
[*]Corresponding Author

ABSTRACT

Feature Selection (FS) is an efficient technique use to get rid of irrelevant, redundant and noisy attributes in high dimensional datasets while increasing the efficacy of machine learning classification. The CSA is a modest and efficient metaheuristic algorithm which has been used to overcome several FS issues. The flight length (fl) parameter in CSA governs crows' search ability. In CSA, fl is set to a fixed value. As a result, the CSA is plagued by the problem of being hoodwinked in local minimum. This article suggests a remedy to this issue by bringing five new concepts of time dependent fl in CSA for feature selection methods including linearly decreasing flight length, sigmoid decreasing flight length, chaotic decreasing flight length, simulated annealing decreasing flight length, and logarithm decreasing flight length. The proposed approaches' performance is assessed using 13 standard UCI datasets. The simulation result portrays that the suggested feature selection approaches overtake the original CSA, with the chaotic-CSA approach beating the original CSA and the other four proposed approaches for the FS task.

Keywords: flight length; feature selection; wrapper; crow search; global search; local search; optimization

Published by Arab Journals Platform, 2022

1 INTRODUCTION

The volume of data is constantly increasing as science and technology advance and dealing with this growing big data has become a huge challenge. When datasets become complex and large, it is critical to refine the information and limit it to only the relevant and useful features using Feature Selection (FS) methods. This preprocessing technique is particularly important in data analysis as it helps to reduce computational time and cost (Ghosh et, al. 2020). Data preprocessing, which includes data cleaning, integration, transformation, reduction, discretization, normalization, and feature selection, is an important and necessary step in machine learning and data mining, and it has become an important and necessary step in classification and regression applications (Chen, 1998). The kinds and number of big data applications are changing dynamically, and their behavior is extremely uncertain. The most common way to use feature selection (FS) is to reduce cost by reducing finishing time to reduce the number of features under some set of criteria and gain a good performance (Alweshah et. al, 2020). As a result, feature selection is a vital data preprocessing approach prior to any model development.

Filter and wrapper approaches are the two most common FS methods. However, the fundamental problem with the filter technique is that the features are picked individually, without direct interaction with the machine learning classification algorithm (Faris et al., 2020). The wrapper method, on the other hand, selects features using a search method and interacts directly with the classifier. Faris et al., 2020; Arora and Anand, 2019). Because learning algorithms are used to evaluate a subset of features, the wrapper technique has a greater computational overhead. In terms of accuracy, however, the wrapper method can exceed the filter approach (Das et. al, 2020).

When attempting to discover the optimal set of features using conventional FS strategies, we may confront numerous disadvantages, like the nesting impact in the sequential search technique, in which a feature that will either be chosen in the forward selection or dropped in the backward selection may not be able to take part in the next step of the algorithm, and the exhaustive search, which guarantees that no ideal subset is neglected, when the problem is extremely large. There should be a total of 2^n subsets in a dataset with *N* features (Abdel-Basset, et. al, 2020). Metaheuristic algorithms may be the best option since they produce good results in a judicious amount of time. They can be a useful substitute to an exhaustive search, which has a large processing cost and might be time-consuming.

Nature-inspired metaheuristic techniques have risen in popularity in the last two decades as the result of their effectiveness in resolving real-word difficult engineering issues. These techniques are capable of utilizing the population's useful information to discover the best solutions (Arora & Anand, 2018). Thus, metaheuristic such as PSO (Boubezoula and

https://digitalcommons.aaru.edu.jo/fcij/vol7/iss2/1
DOI: https://doi.org/10.54623/fue.fcij.7.2.1

Paris, 2012: Kumar and Bhati, 2019), CSA (Sayed et., al 2017: Marcedo et., al 2018: Alhakeem and Ali, 2019: Gupta et., al 2018), ant colony optimization (ACO) (Aghdam et., al 2009: Ghosh et., al 2019), genetic algorithm (GA) (Chatterjee and Bhattacherjee, 2011: Das et., al 2012) , harmony search (HS) (Ramos et., al 2011), differential evolution (DE) (Al-Ani et., al 2013), and quantum-inspired evolutionary algorithm (QEA) (Wang and Yu, 2009) have been used for feature selection. These methods can look for and provide viable solutions in complex spaces by looking for potential best solutions. However, some of these approaches have deficient performance, while others have excessive time complexity. CSA has the great search approach and the easiest implementations, with polynomial time complexity thst id proportional to the problem dimension. In addition, there are fewer parameters to tweak with CSA. As a result, it has been widely used in a variety of applications and has had great success.

The contributions of the paper ar as follows:

- Linear function based flight length for CSA
- sigmoid decreasing function based flight length for CSA
- Chaotic function based flight length for CSA
- simulated annealing function based flight length for CSA
- logarithmic decreasing function based flight length for CSA
- Performance evaluation of the five proposed time varying based flight length for CSA

The sections of this article that follow are structured as follows: Section 2 gives a summary of relevant studies as well as a short introduction to the CSA algorithm, whereas Section 3 discusses the feature selection problem. Section 4 introduces the proposed techniques, while Section 5 gives the simulation outcomes and discussions. Finally, the conclusion is provided in Section 6.

2 Related Works

This section discusses some related works in feature selection using crow search algorithms, as well as a brief overview of the CSA.

2.1 Overview of Related Works

In the field of feature selection, CSA has been used successfully. Arora et al. (2019) have combined the properties of GWO and CSA to resolve the feature selection issues and other unconstrained function optimization problems. In Sayed et al. (2019), the authors used chaos theory to enhance CSA and applied to handle the feature selection problem. A binary CSA was presented in De Souza et al., (2018) using transfer functions to handle the issue of feature selection. In the work of Ouadfel and Abd Elaziz (2020), adaptive dynamic local neighborhood search, awareness probability, and a novel global search strategy are used to improve CSA for handling feature selection related issues. Alhakeem and Ali (2019) utilized CSA to fix the feature selection issues of channel. Here, the efficacy of CSA was confirmed using classification accuracy and processing time,

Published by Arab Journals Platform, 2022

and the no-zero channels condition was used to overcome the zero variables problem. When compared to GA, the simulation results demonstrated the efficiency and robustness of CSA in obtaining the best offline channels. Gupta et al., (2018) used CSA for the Extraction Usability Feature from Hierarchical Model. In the presented study, the CSA was enhanced to achieve an effective result. In terms of accuracy, the validation outcomes prove that MCSA outperforms existing metaheuristics such as standard CSA, MWOA , and BBA. Chaudhuri and Sahu presented tvfl, a time-varying flight length variable, to stabilize the flight length variable of the CSA incorporated with eight diverse transfer functions, and contrasted with binary particle swarm optimization, binary dragonfly optimization, genetic algorithm, binary grey wolf optimization, binary grasshopper optimization algorithm, and binary crow search algorithm. The proposed algorithm produced a favorable result, but the comparison is based on datasets that are mostly low dimensional.

However, because the CSA relied heavily on awareness probability (AP) for a trade-off between the global and local search, most of these methods are trapped in local minima, and the constant flight length does not resemble crows' natural conduct. Crows have excellent spatial memory; in the autumn, they hide around 30000 food seeds in various locations, which they retrieve in the winter (Clayton & Emery, 2005).

2.2 Crow Search Algorithms

Crow search algorithm (CSA), a nature-inspired procedure was suggested by Askarzadeh (2016). This population-based evolutionary computation methods algorithm mimics crow bird behavior and social interaction. Crows are absolutely intelligent birds with a large brain for their size that live in groups (flocks) and conceal their food in hidden places that can be remembered and recovered even after a few months. Furthermore, they are self-conscious in the mirror test. They remember appearance and, if an unwelcoming one is met, they communicate with each other through crows in a difficult and complicated manner. Crows, like other social species, may commit robbery through visually examining other Crows' food hiding places and then attempting to steal their food.

The population includes *N* solutions and *d* dimensions. At iteration *t*, the position of each crow *i* is indicated by a vector $p_i^t = [p_{i1}^t, p_{i2}^t, .., p_{id}^t]$ $for\ i = 1,2,3, \dots N$, where p_i^t is the likely position solution in dimension *d* for crow *i*.

If a crow says *i* want to steal from another crow *j*, in this scenario, two situations could occur:

1. Crow *j* doesn't watch the crow *i* after it, crow *i* will discover the food's store of crow *j* and update its position as given in eq. (2.1).

$$p_i^{(t+1)} = p_i^{(t)} + r_i * fl_j^{(t)} * (m_j^{(t)} - p_i^{(t)}) . \quad 2.1$$

Where fl indicates the flight length. r_i been a random number ∈ [0, 1], p_i the

https://digitalcommons.aaru.edu.jo/fcij/vol7/iss2/1
DOI: https://doi.org/10.54623/fue.fcij.7.2.1

crow position while m_j is the memory of the food location.

2. When crow *j* understands that crow *i* follow her to discover her food's hiding place. The crow *j* moves randomly to fool crow *i* in this situation.

The two cases can be combined mathematically as follow;

$$p_i^{(t+1)} = \begin{cases} p_i^{(t)} + r_i * fl_j^{(t)} * \left(m_j^{(t)} - p_i^{(t)}\right), r_j \geq AP_i^t \\ Choose\ a\ random\ position\ , \quad Otherwise \end{cases} \quad 2.2$$

where $r_i\ and\ r_j$ are a random number ∈ [0, 1] and AP_i^t is the awareness probability of crow *j* at *t iteration*. Crows' searching ability is influenced by the value of fl. High values of fl contribute to global search, while low values contribute to local search (Askarzadeh 2016).

During the algorithm's execution, each crow is evaluated using a well-defined fitness function. The crows then update their positions based on the fitness value. Each new position is validated as viable. The memories of the crows are updated in accordance with eq (2.3)

$$m_i^{(t+1)} = \begin{cases} p_i^{(t+1)} \quad if\ f\left(p_i^{(t+1)}\right)\ is\ better\ than\ f\left(p_i^{(t)}\right) \\ m_i^{(t)}, \quad otherwise \end{cases} \quad 2.3$$

The pseudocode of the standard CSA algorithm is presented in algorithm 1

Algorithm 1: The normal CSA operation

Crows location $p_i^t(1 = 1,2, \ldots m)$ get randomly initialize
The values of fl and AP are set
The fitness function $f\left(p_i^{(t)}\right)$ for each crow p_i^t are evaluated
The memory m_i^q for each crow p_i^t is set
Set t equal to zero, maximum number of iteration equal to max_iter and flock of crows equal to flc
while** (t less than max_iter) **do
  for** k = 1 to flc **do
    choose one of the crows to trail Randomly
    Outline an awareness probability
    If** $(rand \geq AP_k^t)$ **then
      Update crow position Using eq. (2.1)
    else
      Update crow position Using eq. (2.2)
    endif
  endfor
  Evaluate the viability of the new position $p_k^{(t+1)}$
  The new position of the crow $f\left(p_i^{(t+1)}\right)$ get evaluated
  Then the memory of the crow $m_i^{(t+1)}$ is updated using eq. (2.3)
end while
Produce the best solution

3. The proposed Approaches

The optimization technique in every population-based algorithm involves two phases: global search and local search. The technique can converge to the global best position in an acceptable period of time if the two phases are kept in excellent balance. Using the global search operator rather than the local search operator at the start of the search operation is the optimal technique for any algorithm. This permits the algorithm to look into more regions of the search space, which must then be carefully examined in order to find the global optimal solution. In general, one or more parameters maintain this equilibrium in every algorithm. The awareness probability (AP) and the flight length *fl* are the two parameters in the CSA

Published by Arab Journals Platform, 2022

algorithm that perform this role. Greater flight lengths indicate global search, whilst shorter flight lengths indicate local search. The parameter flight length is difficult to tune since small values of *fl* reduce solution exploration, resulting in the possibility of being stuck in local optima. Large values of *fl*, on the other hand, prevent the use of suitable solutions, resulting in delayed convergence or divergence. Instead of using a constant number, the value of flight length *fl* should be gradually reduced during the iteration to achieve this purpose. In the literature, different updating approaches for distinct behaviors were used. Five unique update methods, which were initially designed for continuous optimization issues, were used and presented in the proposed method of this article:

- Flight length with a linearly decreasing strategy: The strategy relies on Shi and Eberhart's (1999) technique, which suggests that global search is best at the beginning of the optimization process and local search is best at the end**.** Flight length *fl* is defined as in Eq. 3.1

$$fl(t) = f_{max} - (f_{max} - f_{min})\frac{t}{max\_iter} \quad (3.1)$$

f_{max} stands for the maximum flight length, f_{min} the minimum flight length, t the current iteration and max_iter the maximum iteration.

- Flight length with sigmoid decreasing strategy: Malik et al., (2007) suggested this approach that relies on the discovery that the sigmoid function aids in acquiring the global optimum objective functions, while linearly lowering the flight length factor aids in speedy convergence. Flight length *fl* is defined as in Eq. 3.2

$$fl(t) = \frac{f_{max} - f_{min}}{1 + e^{(u \times (t - n \times max\_iter))}} + f_{min} \quad (3.2)$$

where $u = 10^{(\log(max_{iter})-2)}$ and $n = 0.25$

f_{max} stands for the maximum flight length, f_{min} the minimum flight length, t the current iteration and max_iter the maximum iteration.

- Flight length with chaotic strategy: Feng et al., (2007) presented a chaotic inertia weight strategy using the advantages of chaotic optimization. In every step of its evolutionary process, this method alternates between rough and minute search phases. Flight length *fl* is defined as in Eq. 3.3

$$fl(t) = (f_{max} - f_{min}) \times \frac{max\_iter - t}{max\_iter} + f_{min} \times z \quad (3.3)$$

$z = 4 \times z \times (1 - z)$

where z is random uniform distributed in the interval of [0,1]

f_{max} stands for the maximum flight length, f_{min} the minimum flight length, t the current

https://digitalcommons.aaru.edu.jo/fcij/vol7/iss2/1
DOI: https://doi.org/10.54623/fue.fcij.7.2.1

iteration and max_iter the maximum iteration.

- Flight length with simulated annealing strategy**:** For optimizing the flight length coefficient, Fayek et al., (2006) recommended utilizing simulated annealing. This method is far superior in terms of convergence speed and long-term stability. Flight length fl is defined as in Eq. 3.4

$$fl(t) = f_{min} + (f_{max} - f_{min}) \times \lambda^{(t-1)} \quad 3.4$$

$where\ \lambda = 0.95$

f_{max} stands for the maximum flight length, f_{min} the minimum flight length, t the current iteration and max_iter the maximum iteration.

- Flight length with a decreasing logarithm strategy: Gao et al. (2008) proposed this method, which is based on the logarithm reduction of the inertia weight coefficient. This mechanism has the potential to accelerate convergence. Flight length fl is defined as in Eq. 3.5

$$fl(t) = f_{max} + (f_{min} - f_{max}) \times log_{10}(a + \frac{10 \times t}{max\_iter}) \quad 3.5$$

where a = 1

f_{max} stands for the maximum flight length, f_{min} the minimum flight length, t the current iteration and max_iter the maximum iteration.

3. 1 Detailed explanation of the proposed method

The proposed approach is described in Algorithm 2. Each step of algorithm 2 is explained in detail here. The first step is to determine the number of features in the dataset, which is denoted by the letter d. A two-dimensional array of size N * d is created in the next step. The number of crows in the dataset is N, and the number of features in the dataset is d. This matrix's rows each represent a feature vector. The fitness of each crow is then assessed using the objective function. Each crow's memory is initialized following the fitness calculation. The initialization phase is finished at this point. The process now iterates for max iter times. Each iteration generates a new crow position with a new flight length, checks the viability of the generated position, calculates the fitness of the feasible position, and updates the crows' memory.

Algorithm 2: The pseudocode of the proposed algorithm

Crows location $p_i^t(1 = 1,2, \dots m)$ get randomly initialize
The values of fl_{max} , fl_{min} and AP are set
The fitness function $f(p_i^{(t)})$ for each crow p_i^t are evaluated
The memory m_i^q for each crow p_i^t is set
Set t equal to 1, maximum number of iteration equal to max_iter and flock of crows equal to N
while** (t less than max_iter) **do
  for** k = 1 to N **do
    choose one of the crows to trail Randomly
    Outline an awareness probability
    If** $(rand \geq AP_k^t)$ **then
      Update the flight length using eq. 3.1, 3.2 ,3.3, 3.4 and 3.5
      Update crow position Using eq. (2.1)
    else
      Update crow position Using eq. (2.2)
    endif
  endfor

16: *Evaluate the viability of the new position* $p_k^{(t+1)}$
17: *The new position of the crow* $f(p_i^{(t+1)})$ *get evaluated*
18: *Then the memory of the crow* $m_i^{(t+1)}$ *is updated using eq. (2.3)*
19: ***end while***
20: *Produce the best solution*

These CSA versions retain the benefits of the initial CSA, such as easiness, consistency, a better search approach, and ease of implementation. The CSA algorithm was first used to solve the continuous optimization problem. Because the feature selection is in the discrete domain, the individual position update equations must be modified to conform to the discrete domain in order to obtain the integer values of the position. Transfer functions are the most common methods used in the related FS literature that use the metaheuristic algorithm. As a result, the sigmoid transfer function (S-shape) is used to convert solution P's continuous search space to binary search space.

4 Implementation of proposed approaches

In this part, the proposed approach has been investigated. The experimental setup, opening CSA input variables, the datasets used, the results and discussion have been presented

4.1 Experimental setup

All of the experiments in this study were run on a PC with an Intel(R) Core (TM) i5-6500 3.20 GHz CPU and 8.0 GB of RAM. The algorithms were implement using python programming language.

4.2 *Input variables*

The mandatory CSA variables that need to be initialized at the start of the algorithm are given in Table 2.

4.3 Datasets

The suggested methods and the standard CSA are tested on 13 benchmark datasets in order to validate its effectiveness. These datasets are from UCI's machine learning repository. The detail of these datasets are depicted in Table 1

Table 1: Summary of Dataset

	Dataset	Number of Features	Number of Instances
1	Wine	13	178
2	Dermatology	34	366
3	Heart	13	270
4	Ionosphere	34	351
5	Lung cancer	56	32
6	Thoracic surgery	16	454
7	Hepatitis	19	155
8	Parkinson	22	194
9	Phishing website	30	2455
10	Qsar biodegradation	41	1054
11	Absenteeism at work	21	733
12	Divorce	54	169
13	Wpdc	34	192

https://digitalcommons.aaru.edu.jo/fcij/vol7/iss2/1
DOI: https://doi.org/10.54623/fue.fcij.7.2.1

Table 2: Initial parameters of the proposed approaches

Parameter	Value
Number of iterations	70
Population magnitude	20
Dimension(D)	Same as number of attributes in the dataset
Search domain	[0 ,1]
Number of runs (P)	20
f_{max}	2.5
f_{min}	2
AP	0.2
β	0.01
α	0.99

4.4 *Results and discussion*

Because meta-heuristic algorithms are stochastic optimization methods, they require at least ten independent runs to produce meaningful statistical results. Each method is tested in 20 independent runs, and the average results of each algorithm are presented. The metrics of the optimal solutions for all approaches in the most recent iteration are presented, including the average fitness value (Mean), standard deviation (SD), average classification accuracy (ACC), and maximum accuracy obtained. Tables 3, 4, 5, and 6 show the results of all approaches in the most recent iteration. The best values obtained are highlighted in bold.

Table 3 also shows the accuracy of all variants over 20 independent runs. This table clearly shows that the chaotic-CSA algorithm outperforms all other algorithms in every case. In these cases, the other CSA-based approaches perform quite well as well. According to the average accuracy metric, the chaotic-CSA algorithm is ranked first, and the logarithm-CSA algorithm is ranked second. The average fitness of the competing approaches is shown in Table 4. According to this table, the chaotic-CSA algorithm has achieved the best fitness metric 46 percent of the time. The logarithm-CSA, on the other hand, has the best fitness metric 31% of the time. Table 5 shows the maximum accuracy achieved in 20 runs by each of the proposed approach's variants. The chaotic-CSA obtained the best outcome 53% of the time, whereas, simulated-CSA performs best 38% of the time. In terms of stability as shown in table 6, sigmoid-CSA performs better in this criterion, closely followed by chaotic-CSA and simulated-CSA. The graph of the metric average fitness, average accuracy and standard deviation between the variants of CSA and the normal CSA are shown in the figure 1-3. It can be seen from the graphs that the chaotic-CSA outperforms the other variants and the normal CSA. The outperformance of chaotic-CSA could be attributed to the non-repeatability property of chaotic sequence which introduces diversity into search space thereby improving the exploration capability of CSA.

Published by Arab Journals Platform, 2022

Table 3: Average Accuracy comparison of variants of CSA and the standard CSA

	CSA	Liner-CSA	Sigmoid-CSA	Chaotic-CSA	Simulated-CSA	Logarithm-CSA
Wine	0.9610	0.9664	0.9655	**0.9689**	0.9667	0.9647
Dermatology	0.9427	0.9430	0.9465	**0.9494**	0.9475	0.9483
Heart	0.8290	**0.8392**	0.8372	0.8359	0.8370	0.8353
Ionosophere	0.8863	0.8930	0.8911	**0.8954**	0.8906	0.8927
Lung cancer	0.9093	0.7870	0.7019	**0.9204**	0.9185	0.7907
Thoracic surgery	0.8512	0.8520	0.8520	0.8519	**0.8525**	0.8522
Hepatitis	0.9186	0.9330	0.9324	0.9349	0.9336	**0.9355**
Parkinson	0.8975	0.9042	0.9048	0.9067	0.9074	**0.9097**
Phishing website	0.9265	0.9324	0.8559	0.9343	0.9334	**0.9346**
Qsar biodegradation	0.8427	**0.8467**	0.8093	**0.8467**	0.8458	0.8449
Divorce	0.9854	0.9868	**0.9798**	0.9871	0.9877	0.9865
Absenteeism_at_work	0.4070	0.4111	0.2547	0.4116	**0.4165**	0.4143
Wpbc	0.8032	0.8094	0.7190	**0.8135**	0.8065	0.8047

Table 4: Average Fitness comparison of variants of CSA and the standard CSA

	CSA	Liner-CSA	Sigmoid-CSA	Chaotic-CSA	Simulated-CSA	Logarithm-CSA
Wine	0.0427	0.0377	0.0383	**0.0352**	0.0371	0.0394
Dermatology	0.0602	0.0600	0.0566	**0.0535**	0.0557	0.0547
Heart	0.1726	**0.1621**	0.1643	0.1655	0.1645	0.1662
Ionosophere	0.1145	0.1076	0.1096	**0.1053**	0.1102	0.1079
Lung cancer	0.0921	0.2294	0.3043	**0.0809**	0.0828	0.2342
Thoracic surgery	0.1491	0.1484	0.1483	0.1485	**0.1483**	**0.1483**
Hepatitis	0.0826	0.0686	0.0690	0.0668	0.0681	**0.0665**
Parkinson	0.1042	0.0970	0.0972	0.0951	0.0944	**0.0922**
Phishing website	0.0763	0.0703	0.0688	**0.0682**	0.0693	**0.0682**
Qsar biodegradation	0.1589	**0.1550**	0.1604	0.1562	0.1558	0.1568
Divorce	0.0163	0.0145	**0.0138**	0.0145	**0.0138**	0.0148
Absenteeism_at_work	0.5888	0.5848	0.5798	0.5841	**0.5791**	0.5813
Wpbc	0.1973	0.1899	**0.1879**	**0.1879**	0.1944	0.1944

https://digitalcommons.aaru.edu.jo/fcij/vol7/iss2/1
DOI: https://doi.org/10.54623/fue.fcij.7.2.1

Table 5: Maximum Accuracy Obtained by variants of CSA and the standard CSA

	CSA	Liner-CSA	Sigmoid-CSA	Chaotic-CSA	Simulated-CSA	Logarithm-CSA
Wine	0.9713	**0.9774**	**0.9774**	**0.9774**	**0.9774**	0.9718
Dermatology	0.9664	0.9552	0.9580	**0.9692**	0.9608	0.9634
Heart	0.8402	**0.8513**	**0.8513**	0.8402	0.8476	0.8402
Ionosophere	0.8944	0.9058	0.8973	0.9087	**0.9144**	0.9059
Lung cancer	0.9259	0.9259	0.8888	0.9259	**0.9630**	0.9259
Thoracic surgery	0.8524	0.8524	0.8524	**0.8547**	0.8546	0.8546
Hepatitis	**0.9625**	0.9620	**0.9620**	**0.9625**	**0.9625**	**0.9625**
Parkinson	0.9173	0.9173	0.9123	0.9173	0.9225	**0.9282**
Phishing website	0.9393	0.9442	0.9328	**0.9554**	0.9397	0.9470
Qsar biodegradation	0.8567	0.8586	0.8425	0.8558	0.8557	0**.8605**
Divorce	0.9883	**0.9942**	0.9883	0.9883	0.9883	0.9883
Absenteeism_at_work	**0.4297**	0.4229	0.3138	**0.4297**	**0.4297**	**0.4297**
Wpbc	0.8142	0.8283	0.7708	**0.8333**	0.8281	0.7812

Table 6: Standard deviation comparison of variants of CSA and the standard CSA

	CSA	Liner-CSA	Sigmoid-CSA	Chaotic-CSA	Simulated-CSA	Logarithm-CSA
Wine	0.0087	0.0065	**0.0064**	0.0068	0.0371	0.0069
Dermatology	0.0136	0.0090	0.0087	**0.0081**	0.0105	0.0102
Heart	0.0087	0.0062	0.0071	0.0060	**0.0050**	0.0075
Ionosophere	0.0059	0.0064	**0.0043**	0.0085	0.0079	0.0077
Lung cancer	0.0221	0.1083	0.0308	**0.0133**	0.0828	0.0985
Thoracic surgery	0.0009	0.0005	**0.0004**	**0.0004**	**0.0004**	0.0006
Hepatitis	0.0173	0.0118	0.0132	0.0141	0.0161	**0.0115**
Parkinson	0.0111	0.0085	0.0062	**0.0057**	0.0071	0.0091
Phishing website	0.0063	0.0045	0.0045	0.0056	**0.0030**	0.0046
Qsar biodegradation	0.0066	0.0057	**0.0034**	0.0044	0.0052	0.0049
Divorce	0.0027	0.0029	**0.0017**	0.0021	**0.0017**	0.0025
Absenteeism_at_work	0.0115	**0.0078**	0.0095	0.0102	0.0090	0.0106
Wpbc	0.0105	0.0081	0.0090	0.0083	0.0069	**0.0062**

Published by Arab Journals Platform, 2022

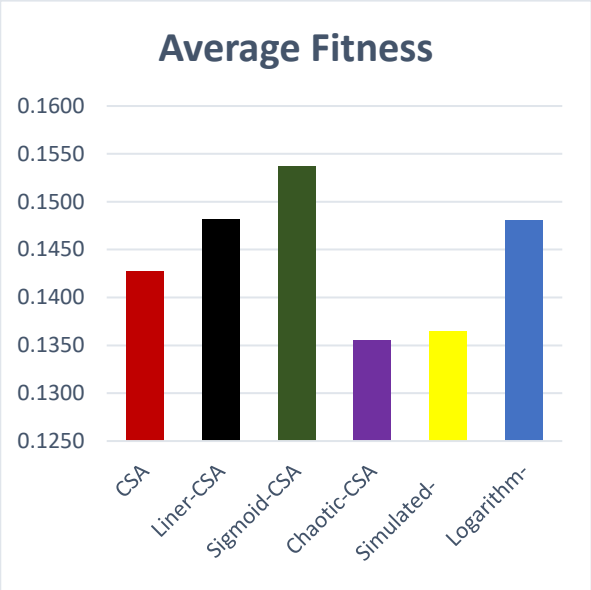

Figure 1: The average fitness comparison between variants of CSA and normal CSA

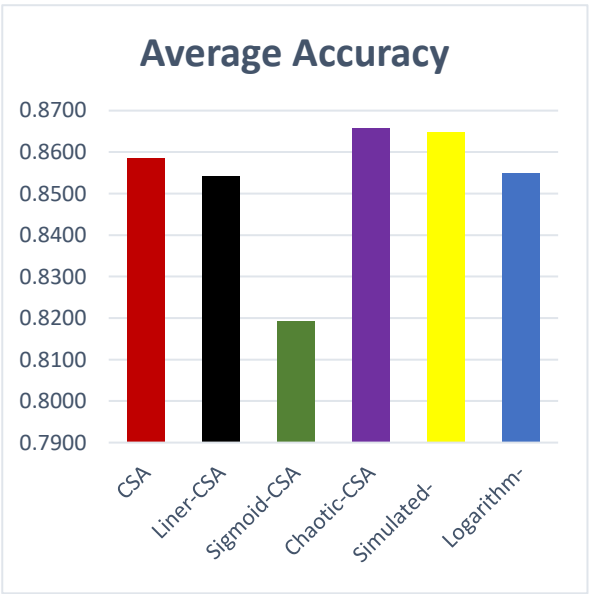

Figure 2: The average accuracy between the variants of CSA and normal CSA

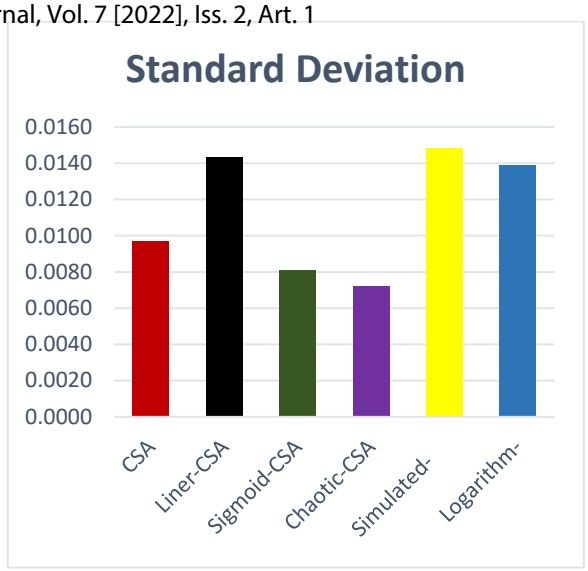

Figure 3: The standard deviation comparison between the variants of CSA and the normal CSA

5 **Conclusion**

This article presents a CSA-based feature selection strategy with different time-varying flight length mechanisms. The impact of using five different flight length updating strategies is studied. The proposed approaches and standard CSA algorithms are implemented to FS tasks for evaluation purposes. The experimental results show that the chaotic-CSA performed better in the Wine, Dermatology, Ionosphere, Lung Cancer, and wpbc datasets, the linear CSA performed best in the Heart dataset, and the Linear CSA performed best in the Hepatitis, Parkinson, and Phishing websites. The Sigmoid CSA and simulated CSA has the best performance in Divorce and Absenteeism_at_work datasets respectively.

In terms of Fitness value, the chaotic CSA has the best performance in seven of the datasets, the Simulated CSA and Logarithm CSA has the best performance in three of the datasets each, while the

https://digitalcommons.aaru.edu.jo/fcij/vol7/iss2/1
DOI: https://doi.org/10.54623/fue.fcij.7.2.1

Linear and Sigmoid has the best performance in two of the datasets each.

Moreover, in terms of the standard deviation, the Sigmoid CSA has the best performance in five of the datasets, the chaotic and simulated CSA in four of the datasets, while the linear and logarithm performed better in one two of the datasets respectively. In average, the chaotic-CSA approach achieves the superior performance for the FS task compared to original CSA and other four approaches. The outperformance of chaotic-CSA could be attributed to the non-repeatability property of chaotic sequence which introduces diversity into search space thereby improving the exploration capability of CSA. As a result, when dealing with dimensionality reduction problems, chaotic-CSA should be viewed as a candidate strategy. This proposed approach's application in other areas can be investigated as future work.

REFERECES

[1] Abdel-Basset, M., El-Shahat, D., El-henawy, I., de Albuquerque, V., and Mirjalili, S. (2020). A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection. Expert Systems With Applications 139 (2020) 112824.

[2] Aghdam, M. H, Ghasem-Aghaee, N. and Basiri, M.E. (2009) *Text feature selection using ant colony optimization*, Expert Syst. Appl. 36 (2009), pp. 6843–6853.

[3] Al-Ani, A. Alsukker, and R.N. Khushaba, (2013). *Feature subset selection using differential evolution and a wheel based search strategy*, Swarm Evol. Comput. 9 , pp. 15–26.

[4] Alweshah, M., Alkhalaileh, S., Albashish, D., Mafarja, M., Bsoul, Q., and Dorgham, O., (2020). A hybrid mine blast algorithm for feature selection problems. Soft Computing, 10.1007/s00500-020-05164-4

[5] Arora, S., & Anand, P. (2018). Chaotic grasshopper optimization algorithm for global optimization. *Neural Comput. Appl.*, 1–21.

[6] Arora, S., & Anand, P. (2019). Binary butterfly optimization approaches for feature selection. *Expert Syst. Appl., 116*, 147–160.

[7] Arora, S., Sharma, M., & Anand, P. (2020). A novel chaotic interior search algorithm for global optimization and feature selection. *Appl. Artif. Intell., 34*(4), 292–328.

[8] Askazadeh, A. (2016). A novel metaheuristic method for solving constraint engineering. O*ptimization problems:* crow search algorithm. *Computer Structure*, 160, 1-12.

[9] Dash, M. and Luo, H. (1997). Feature selection for classification. . *Intelligent Data Analysis*, 1(4), 131-156, doi: 10.1016/S1088-467X(97)00008-5.

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

[11] Ghosh, M., Guha, R., Sarkar, R. and Abraham, A. (2019). A wrapper-filter feature selection technique based on ant colony optimization. *Neural Computing and Applications*, 32, 7839–7857.

[12] Gupta, D., Sundaram, S., Khanna, A., Hassanien, A. E. and Victor, H. A. (2018). Improved diagnosis of Parkinson's disease using optimize crow search algorithm. . *Computer and Engineering*, 68, 412-424. *Proceeding of IEEE International Conference*, 23, 1942-1948.

Published by Arab Journals Platform, 2022

[13] Kumar, L. and Bharti, K. K. (2019). An improved BPSO algorithm for feature selection. *In Recent trends in communication, computing, and electronics*, 524, 505-513.

[14] Mafarja, M. Mirjalili, S. (2018). Whale optimization approaches for wrapper feature selection. Applied Soft computing. 62, 441-453.

[15] Al-Ani, A., Alsukker, A. and Khushaba, R. N. (2013). Feature subset selection using differential evolution and a wheel based search strategy. *Swarm Evol. Comput*, 9, 15-26.

[16] Ghosh, M., Guha, R., Sarkar, R. and Abraham, A (2020). A wrapper-filter feature selection technique based on ant colony optimization. Neural Computing and Applications 10.1007/s00521-019-04171-3

[17] Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research 3, 1157–1182.

[18] Mafarja, M. and Abdullah, S. (2014). Fuzzy modified great deluge algorithm for attribute reduction. In Recent Advances on Soft Computing and Data Mining. Springer, Cham, 195–203.

[19] Faris, H., Aljarah, I. and Al-Shboul, B., (2016). A Hybrid Approach Based on Particle Swarm Optimization and Random Forests for E-Mail Spam Filtering. Springer International Publishing, Cham, 498–508.

[20] Faris, H., Mirjalili, S., Aljarah, I., Mafarja, M., & Heidari, A. A. (2020). Salp swarm algorithm: theory, literature review, and application in extreme learning machines Nature-Inspired Optimizers (pp. 185–199): Springer.

[21] Faris, H., Hassonah, M. A., Al-Zoubi, A. M., Mirjalili, S. and Aljarah, I., (2017). A multi-verse optimizer approach for feature selection and optimizing SVM parameters based on a robust system architecture. Neural Computing and Applications (02 Jan 2017).

[22] Mafarja, M. and Abdullah, S. (2013). Investigating memetic algorithm in solving rough set attribute reduction. International Journal of Computer Applications in Technology 48, 3, 195–202.

[23] Mafarja, M. and Mirjalili, S. (2017). Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. Neurocomputing.

[24] Chatterjee, S. and Bhattacherjee, A. (2011). *Genetic algorithms for feature selection of image analysis-based quality monitoring model: An application to an iron mine*, Eng. Appl. Artif. Intell. 24, pp. 786–795.

[25] Chen, B.L., Chen, L and Chen, Y.X. (2013). *Efficient ant colony optimization for image feature selection*, Signal Process. 93, pp. 1566–1576.

[26] Das, N., Sarkar, S. Basu, S., Kundu, Nasipuri, M. and D.K. Basu, (2012)*A genetic algorithm based region sampling for selection of local features in handwritten digit recognition application,*Appl. Soft Comput. 12, pp. 1592–1606.

[27] Ramos, C.C.O. , Souza, A.N. and Papa, J.P. (2011). *A novel algorithm for feature selection using Harmony search and its application for non-technical losses detection*, Comput. Electr. Eng. 37 (2011), pp. 886–894.

[28] Wang, L. Niu, Q. and Fei, M. (2008). *A novel quantum ant colony optimization algorithm and its application to fault diagnosis*, Trans. Inst. Meas. Control. 30 (2008), pp. 313–329.

[29] Wang, L., Wang, X.T. and Yu, J.S., (2009). *Naphtha cracking furnace fault diagnosis based on adaptive quantum ant colony algorithm*, J. Chem. Ind. Eng. 60, pp. 401–408.

[30] Boubezoula, A. and S. Paris, S. (2012). *Application of global optimization methods to model and feature selection,* Pattern Recognit. 45, pp. 3676–3686.

https://digitalcommons.aaru.edu.jo/fcij/vol7/iss2/1
DOI: https://doi.org/10.54623/fue.fcij.7.2.1

[31] Chatterjee, S. and A. Bhattacherjee, A., (2011). *Genetic algorithms for feature selection of image analysis-based quality monitoring model: An application to an iron mine*, Eng. Appl. Artif. Intell. 24, pp. 786–795.

[32] Chen, B.L., Chen, L, and Chen, Y.X. (2011). *Efficient ant colony optimization for image feature selection*, Signal Process. 93, pp. 1566–1576.

[33] Chuang, L.Y., Chang, H.W. , Tu, C.J. and Yang, C.H., (2008). *Improved binary PSO for feature selection using gene expression data*, Comput. Biol. Chem. 32 (2008), pp. 29–38.

[34] Chuang, L.Y. , Yang, C.H. and Li, J.C., (2011). *Chaotic maps based on binary particle swarm optimization for feature selection*, Appl. Soft Comput. 11 (2011), pp. 239–248.

[35] Chuang, L., Tsai, S. and Yang, C., (2011). *Improved binary particle swarm optimization using catfish effect for feature selection*, Expert Syst. Appl. 38 (2011), pp. 12699–12707.

[36] Shi, Y., and Eberhart, R.C. (1999). Empirical study of particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation—CEC99 (Cat. No. 99TH8406), vol. 3, pp. 1945–1950. IEEE

[37] Malik, R.F., Rahman, T.A., Hashim, S.Z.M., Ngah, R. (2007). New particle swarm optimizer with sigmoid increasing inertia weight. Int. J. Comput. Sci. Secur. 1(2), 35–44

[38] Feng, Y., Teng, G.F., Wang, A.X., Yao, Y.M. (2007). Chaotic inertia weight in particle swarm optimization. In: Second International Conference on Innovative Computing, Information and Control (ICICIC), pp. 475–475. IEEE

[39] Al-Hassan, W., Fayek, M., Shaheen, S. (2006). PSOSA: an optimized particle swarm technique for solving the urban planning problem. In: 2006 International Conference on Computer Engineering and Systems, pp. 401–405. IEEE

[40] Gao, Y.L., An, X.H., Liu, J.M. (2008). A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation. In: 2008 International Conference on Computational Intelligence and Security, vol. 1, pp. 61–65. IEEE