e-ISSN : 0976-5166
p-ISSN : 2231-3850

A NOVEL HYBRID ALGORITHM BASED ON CROW SEARCH ALGORITHM AND WHALE OPTIMIZATION ALGORITHM FOR HIGH-DIMENSIONAL OPTIMIZATION AND FEATURE SELECTION

[1]Artee Abudayor
[1]Computer Engineering Department, Engineering Faculty, Erciyes University
Talas, Kayseri, Turkey
[1]jamendat@gmail.com

[2]Özkan Ufuk Nalbantoğlu
[2]Computer Engineering Department, Engineering Faculty, Erciyes University
Talas, Kayseri, Turkey
[2]nalbantoglu@erciyes.edu.tr

Abstract

Crow search algorithm (CSA) is a meta-heuristic algorithm that mainly solves optimization problems. The weaknesses of the original CSA were its slow convergence speeds and inefficient exploitation capacity. Hence, this paper proposed a novel hybrid based on CSA and whale optimization algorithm (WOA), which is called HCSWOA for high-dimensional optimization problems. The main idea is to integrate two different algorithms' strengths into a proposed algorithm that utilizes the exploration ability of CSA with the exploitation and convergence abilities of WOA. To enhance the performance of the original WOA and CSA, this study employed an adaptive inertia weight strategy to improve exploitation and exploration capacities and convergence speed. The proposed algorithm has been compared against the original CSA, WOA, Grey Wolf Optimizer (GWO), Dragonfly Algorithm (DA), Particle Swarm Optimization (PSO), Sine Cosine Algorithm (SCA), Ant Lion Optimization algorithm (ALO), and Differential Evolution (DE) by using twenty-three standard benchmark functions and a real-world engineering problem as feature selection. The proposed algorithms have been examined on eighteen UCI standard and two DNA microarray datasets. The experimental results have revealed that HCSWOA has comprehensive superiority in solving global optimization and feature selection problems, which proves the capability of the proposed algorithm in solving real-world engineering problems.

Keywords: Crow search algorithm (CSA), Whale optimization algorithm (WOA), High-dimension, Optimization Problem, Feature selection, and Hybrid optimization.

1. Introduction

Recently, researchers have been interested in challenging optimization problems, such as global optimization, multi-objective optimization, and high-dimensional optimization problems. An interesting goal is to solve these problems with the optimal algorithm, which finds the optimal solution among all the available alternatives. The optimal algorithm can be divided into deterministic and stochastic optimization algorithms. The deterministic algorithms usually solve the problem both the same initialization set and steps are followed in order every time the algorithms run. This means that only one execution of each problem instance is needed. In contrast, stochastic optimization algorithms utilize random search to reach the optimal solution, which means the algorithm should repeat the executions under different conditions. However, these algorithms might end up finding the same final solutions in most cases. In addition, meta-heuristic algorithms (MAs) are stochastic optimization algorithms that are popular in solving real-world engineering problems due to their efficient performance and robustness, allowing them to find alternative solutions to the optimal ones for reasons of time, which have been successfully applied in different application domains, such as feature selection problems [1] , travel salesman problems [2], image processing [3], wireless sensor networks [4] etc.

Feature selection (FS) is a multi-objective optimization problem as a preprocessing step of the datasets, which is a dimensionality reduction technique for prediction [1] or classification by removing redundant and irrelevant features. Moreover, they also help to reduce computational time and increase classification accuracy and are successful in solving feature selection problems for classification tasks in different domains, such as bioinformatics [5], image processing [6], text mining [7], finance [8], and so on. According to the advantages of FS, many researchers emphasize employing MAs to solve feature selection problems, such as Particle swarm optimization(PSO) [9], Equilibrium Optimizer (EO) [10], Artificial bee colony algorithm(ABC) [11], and Ant colony optimizer (ACO) [12], Altruistic Whale Optimization Algorithm (AltWOA) [13], Sine cosine algorithm (SCA) [14], Seagull optimization algorithm (SOA) [15]. Feature selection can be divided into three categories: filter, wrapper, and hybrid methods [16]. To begin with, filter methods involve independently learning algorithms with only one iteration. Filter methods can be divided into two categories: univariate and multi-variate. Both of them usually give a score for each feature or group of features. Therefore, it is easy to rank features and select the best features between them or remove some features below a threshold, such as in DEFRS, a differential evolution (DE) algorithm that combines with fuzzy rough set theory [17]. To validate the performance of DEFRS using fourteen datasets from popular repositories, such as ionosphere, wbcd, sonar, hill, colon, and etc. A modified Binary Ant System (BAS) named FSCBAS was proposed [18]. The aim of FSCBAS was to avoid falling into local optima by estimating the correlation between the selected feature subsets and a new set.

Wrapper methods evaluate the importance of selected feature subsets using dependently learning algorithms. In addition, the wrapper methods iteratively produce different candidate feature subsets in some strategies and use a classifier algorithm to calculate the corresponding classification accuracy. An example of this method is Binary Cuckoo Search [19] and IBPSO [20], which had utilized search agents and convert them into a binary vector in each dimension by utilizing S-shaped transfer functions to select the significant feature subsets. Whale Optimization Algorithm (WOA) with tournament and roulette wheel selection mechanisms in the search [21] is called WOA-T and WOA-R, respectively. WOA-T and WOA-R algorithms had been tested on two-DNA microarrays and eighteen UCI standard datasets. Then, both of them compared three meta-heuristic algorithms, namely, PSO, GA, and ALO, and five-filter feature selection methods. The modified binary BA with a k-means clustering algorithm was proposed in [22]. To validate the selected features by the proposed method, classification algorithms like decision tree induction, support vector machines, and Naïve Bayesian classifiers were used and tested on eight different datasets, which are publicly available in the UCI machine learning repository.

Lastly, the combination of filter and wrapper methods is called the hybrid method. MOFOA with Fisher score, proposed mechanism the repository criterion for searching the feature subsets for separate populations, crowding-distance, and binary tournament selection, which are called GSMOFOA [23]. The GSMOFOA evaluated six DNA microarray datasets and compared the proposed algorithm with four hybrid multi-objective methods, such as MOBBBO, MOPSO, NSGA-II, and MOBAT. PSO with a correlation coefficient [24] was tested on the lymphoma, MLL, and SRBCT datasets. The experiment results demonstrated applying to comparison performance six classifier algorithms, such as ELM, J48, random forest, random tree, decision stump, and genetic programming. As a result, the ELM classifier algorithm achieved the highest accuracy on the SRBCT, MLL, and lymphoma datasets, with 93.7%, 85.6%, and 96.8%, respectively. In [25], a proposed GWO with IG was tested on two datasets, the breast and colon datasets. The results showed that the breast and colon datasets had the best possible classification accuracy of 94.87% and 95.935%, respectively. RFACO-GS is the name given to ACO combined with ReliefF in [26], which describes the experimental results on six DNA microarray datasets and comparisons with other algorithms such as the Fisher score, MIMAGA, and so on. Moreover, the results demonstrate that the RFACO-GS algorithm is very effective, with 94% and 99.5% for the colon and lung datasets, respectively.

Besides, many researchers have used CSA algorithm to solve feature selection problems for classification tasks in different domains, such as diseases, documents, and drugs. Furthermore, mostly modified CSA in wrapper-based feature selection is lacking for high-dimensional data, such as DNA microarray technology [27], which provides the expression profile of thousands of genes, resulting in the curse of dimensionality, and high computational execution is needed. An example of modified CSA in wrapped-based feature selection: CSA with a V-shaped transfer function helps select a feature subset and is called BCSA [28]. BCSA achieved results in terms of classification accuracy and selected features or subsets with a small number of features at a low computational cost.BCSA with an opposition-based learning strategy and an improved flight length (fl) parameter of CSA is called OBL-BCSA [29]. The performance of the OBL-BCSA was evaluated on drug and biodegradable datasets. CSA can be combined with other strategies to aid in the selection of significant feature subsets, such as adaptive awareness probability (AP), dynamic local neighborhood search, and a global search strategy known as ECSA for a wrapper feature selection method [30]. The results of ECSA were evaluated on 16 UCI standard datasets and revealed that the algorithm could achieve a better convergence speed and a better-quality solution. In [31], it was proposed to use the CSA with K-Nearest Neighbor (KNN) for document classification on the Reuters-21578, Webb, and Cade 12 datasets.

In this study, we focus on crow search algorithm (CSA) as a new population-based meta-heuristic optimization method, and it was developed by Askazadeh et al. in 2016 [32]. The algorithm imitates the crows' behavior, in which a crow individual endeavors to hide the place for storing their food from other crows, who could follow them to steal the food. In addition, the advantages of CSA are such as easy implementation and a few control parameters. However, the weakness of the original CSA has slow convergence speeds and is inefficient on the exploitation capacity. So, many researchers attempt to improve the performance of CSA, coming up with modifications such as A modified crow search algorithm (MCSA) [33], which improved the exploitation capacity of CSA by adaptive adjustment of the flight length (fl). ICSA improved its exploration and exploitation capacities and enhanced convergence speed with adaptive inertia weight parameter and roulette wheel selection for multi-dimensional, linear, and nonlinear problems [34]. The Sine Cosine Crow Search Algorithm (SCCA) was proposed to balance between exploration and exploitation capacities [35], [36]. CSA with Grey wolf optimization algorithm (GWOCSA) was reported to achieve global optima efficiently [37]. CSA with non-inferior neighborhood search is called NICSA in which is used for balancing exploration and exploration capacities [38]. The results of NICSA had shown that it attains a good search capacity, convergence rate, and robustness. An improved CSA (ICSA) with adaptive adjustment operator and Levy΄ flight distribution uses the position update mechanism of crows, and it also balances the exploration and exploitation capabilities of CSA [39]. According to literature review, many researchers attempted to modify the CSA in various ways in order to improve their performance with scale problems on 30-dimensional variables and a lack of handling with high-dimensional data. To address the problem of CSA, this study is to fill a research gap for proposed algorithms for hybridization based on stochastic population-based meta-heuristic algorithms that can manipulate high-dimensional data.

hybridization techniques can aid in improving performance and attaining efficient results in optimization problems of MAs [40]. Talbi et al. [41] defined a hybridization technique combining several algorithms categorized as high-level or low-level hybrid methods. This study aims to propose an improved CSA with a combination of WOA by hybridization techniques because the main weakness of CSA is its exploitation capacity, which can be enhanced by introducing WOA components with the low-level hybrid method that embeds in CSA and the algorithms work together. Moreover, WOA has a stable exploitation capacity and it also convergences quickly towards the optimum [42].

Whale optimization algorithm (WOA) is a meta-heuristic optimization algorithm that was proposed by Mirjalili et al. in 2016. The algorithm mimics the social behavior of humpback whales, which is the bubble-net hunting technique. The performance results of WOA demonstrate that it outperforms PSO, GSA, and Fast Evolutionary Programming (FEP) in terms of both stable exploitation ability and rapid convergence to the optimum. An example of hybridization of WOA with other algorithms for enhancing the exploitation or/and exploration capacities, such as WOA with Mean Grey Wolf Optimizer (MGWO) for balancing exploitation and exploration abilities, avoiding both premature convergences and traps in local minima [43]. WOA with adaptive switching of the random walk to improve the exploration phase is called AWOA [44]. A hybrid WOA with modified differential evolution (MDE) is called MDE-WOA for solving global optimization problems [45]. The MDE-WOA algorithm was proposed to enhance local optimum avoidance ability and exploration capacity. WOA with the Fish Swarm Algorithm (AFSA) for improving robustness and convergence speed. To improve the premature convergence of WOA, WOA with a symbiotic organism search (SOS), named WOAmM, was introduced [46]. To avoid premature convergence and enhance the ability to explore the flight path of WOA, known as LWOA, which was proposed [47].

The aim of this study is to propose novel algorithms by considering the strengths of CSA and WOA, which are ideal for hybridization and can deal with high-dimensional data to accomplish more suitable exploitation and exploration capacities and offer significantly better results than the conventional original algorithms. Further, the proposed algorithms are able to be applied to other real-world engineering problems, such as feature selection to indicate the significant feature for classification tasks to confront the curse of dimensionality, which takes more than 2^n execution time to find all alternative solutions to optimal ones. As the rule of no free lunch (NFL) on meta-heuristic algorithms states [48], an algorithm may be successful in solving some problems, but not all of them. Therefore, this study attempts to modify an original CSA algorithm to encourage two differential problems, such as high-dimensional optimization and feature selection problems.

To conclude, we will summarize the steps of this research work as follows: 1) Novel hybridization algorithms based on CSA and WOA have been proposed. 2) The proposed algorithm is applied to 23 benchmark function optimization problems with different dimensional data. 3) The proposed hybrid algorithm is employed to solve the feature selection problem in the wrapper method, and the results are validated on eighteen data sets from UCI standard and DNA-microarray datasets. 4) The proposed approach's performance is compared to that of traditional CSA, WOA, and other meta-heuristic algorithms such as Grey Wolf Optimizer (GWO), Dragonfly Algorithm (DA), Particle Swarm Optimization (PSO), Sine Cosine Algorithm (SCA), Ant Lion Optimization algorithm (ALO), and Differential Evolution (DE) in different metrics of performance, such as precision, recall, F-score, classification accuracy, and so on.

The remainder of the paper is arranged as follows: Section 2 presents the related works. Section 3 presents the background information of CSA and WOA focussing on their inspiration and mathematical model. The proposed hybrid algorithm is presented in Section 4, whereas the experimental results on benchmark function optimization problems as well as feature selection problems are discussed in Section 5. Finally, in Section 6, we will indicate the conclusions and future work.

2. Methods

2.1 Crow search algorithm (CSA)

Crow search algorithm (CSA) was developed by Askazadeh et al. in 2016 [32]. The algorithm imitates crows' behavior in that a crow individual attempts to hide a place to store their food. Moreover, the crow takes precautions to protect their location from other crows, who could follow them to steal the food. Therefore, the motion of each crow individual to protect their food has been induced by two main situations: firstly, finding the hiding place by other crows, as shown in Eq. (1).

$$\vec{X}_p(iter+1) = \vec{X}_p(iter) + fl_p(iter) + rand() \times \left[\vec{M}_r(iter) - \vec{X}_p(iter)\right] \quad (1)$$

Where, p =1,2,..., N_C; N_C; iter =1,2 ...,$iter_{max}$; N_C is the flock size, that as the number of the crows; $iter_{max}$ is the number of maximum iterations. $\vec{X}_p(iter+1)$, $fl_p(iter)$ dedicate the current location and flight length of the p-th crow individual at the iter-th iteration, respectively. rand() is a random number by the range of [0,1]; Additionally, The food hiding place of crow r at iter-th iteration is represented by $\vec{M}_r(iter)$.

Secondly, The crow r may be aware that they are being followed by other crows as p. Therefore, the other crow r deceives crow p, and crow p chooses a position randomly, as shown in Eq. (2)

$$\vec{X}_p(iter+1) = a\ random\ position \quad (2)$$

According to descriptive two motions of the crow individuals for finding other crows' food hiding place, which can be expressed in Eq. (3):

$$\vec{X}_p(iter+1) = \begin{cases} \vec{X}_p(iter) + fl_p(iter) + rand(\ ) \times \left[\vec{M}_r(iter) - \vec{X}_p(iter)\right], & if\ r \geq AP_r(iter) \\ a\ random\ position & otherwise \end{cases} \quad (3)$$

Where, r dedicates a random number by the range of [0,1]; $AP_r(iter)$ is the awareness probability of the r-th crow at the iter-th iteration.

2.2 Whale optimization Algorithm

Whale optimization algorithm (WOA) is a meta-heuristic optimization algorithm that was proposed by Mirjalili et al. in 2016 [42]. The algorithm is inspired by the social behavior of humpback whales, which use a bubble-net hunting technique. The foraging behavior of humpback whales involves a special hunting method called bubble-net feeding method. In the step of hunting for prey, whales dive down, and then they start to create a bubble in a spiral shape around the prey and swim up toward the surface. Consequently, the process of WOA consists of three steps: shrinking the encircling prey, spiral bubble-net feeding maneuver, and searching for prey.

• Shrinking Encircling prey: The mathematically model for encircling prey by the whales, they can recognize the location of prey and then encircle them as presented in Eq. (4) and (5).

$$\vec{D} = \left|\vec{C}.\vec{X}^*(iter) - \vec{X}(iter)\right| \quad (4)$$

$$\vec{X}_p(iter+1) = \left|\vec{X}^*(iter) - \vec{A}.\vec{D}\right| \quad (5)$$

where $\vec{X}^*(iter)$, $\vec{X}(iter)$ are the best position vector and the current position vector at iter-th iteration, respectively. Additionally, $\vec{X}^*(iter)$, should be updated to the best position in each iteration if there is a better solution.

• Spiral Bubble-net attacking method: The mathematical model for the bubble-net behavior of humpback whales is in the exploitation phase of WOA. A spiral equation is created between the position of whale and prey mimicking the helix-shaped movement of humpback whales as follows in Eq. (6):

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.cos(2\pi l) + \vec{X}^*(iter) \quad (6)$$

Where b is a constant for defining the logarithmic spiral shape and l is a random number in [-1,1].
Additionally, to update the position of the whales in the exploitation phase of WOA, that can assume a probability (p) of 50% to choose between the shrinking encircling prey or the spiral bubble-net attacking mechanisms.

• Search for prey: The mathematical model of the search for prey is as exploration phase of WOA as shown in Eq. (7) and Eq. (8).

$$\vec{D} = \left|\vec{C}.\vec{X}_{rand}(iter) - \vec{X}(iter)\right| \quad (7)$$

$$\vec{X}_p(iter+1) = \vec{X}_{rand}(iter) - \vec{A}\vec{D} \quad (8)$$

Where $\vec{X}_{rand}(iter)$ is a random position vector chosen from the current population.

3. The Proposed Algorithm

In this study, we propose a novel hybrid based on CSA and WOA by employing the advantage of WOA for enhancing the local search (exploitation capacity) and convergence speeds in the original CSA. However, the CSA algorithm is efficient in global search (exploration capacity). Thus, the proposed algorithm could be a hybrid of two advantages and could properly balance exploration and exploitation capacities for solving real engineering problems.

The proposed algorithm consists of two steps: adaptive inertia weight ($\boldsymbol{in_\omega}$) strategy and a modified position update mechanism as follows:

3.1 Adaptive inertia weight (in_ω) strategy

To enhance the performance of both CSA and WOA algorithms, we utilize the inertia weight to improve the convergence speed and also control exploration and exploitation capacities in every iteration, as shown in Eq. (9).

$$in_\omega = (\omega_{max} - \omega_{min}) \times \left(\frac{iter_{max} + iter}{2 \times iter_{max}}\right) \quad (9)$$

Where: ω_{max} and ω_{min}dedicate 0.9 and 0.4 which provide excellent results, respectively.

3.2 Modified Position Update Mechanism

The proposed algorithm employs the low-level hybridization of two algorithms between CSA and WOA because the weakness of the CSA is its convergence speed and inefficient exploitation ability. Therefore, to increase the efficiency of the exploration phase of the CSA in our proposed algorithm HCSWOA, the update position is calculated based on the inertia weight, as shown in Eq. (9), obtained so far as calculated by Eq. (10).

$$\vec{X}_p(iter+1) = \left(in_\omega \times \vec{X}_p(iter)\right) + fl_p(iter) + rand() \times \left[\vec{M}_r(iter) - \vec{X}_p(iter)\right] \quad (10)$$

In the exploitation phase of HCSWOA, we proposed four different ways to find all alternative optimal solutions by employing inertia weighted (in_ω), which is used to control the movement in each iteration to improve WOA algorithm's position update. Then, to update the position equation, it is modified as per the calculated inertia weights, as shown in Eq. (11)-(14), which are called $HCSWOA_1$, $HCSWOA_2$, $HCSWOA_3$, and $HCSWOA_4$, respectively, as follows:

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.cos(2\pi l) + \left(\vec{X}^*(iter) \times in_\omega\right) \quad (11)$$

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.sin(2\pi l) + \left(\vec{X}^*(iter) \times in_\omega \times cos\left(7 \times \frac{iter}{iter_{max}} \times \pi\right)\right) \quad (12)$$

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.cos(2\pi l) + \left(\vec{X}^*(iter) \times in_\omega \times sin\left(7 \times \frac{iter}{iter_{max}} \times \pi\right)\right) \quad (13)$$

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.cos(2\pi l) + \left(\vec{X}^*(iter) \times in_\omega \times 0.5 \times sin\left(7 \times \frac{iter}{iter_{max}} \times \pi\right)\right) \quad (14)$$

In addition, to update the position of the our proposed algorithms HCSWOA, that can assume an awareness probability of the r-th crow (AP_r) of 80% to choose between the exploitation phase (WOA) or the exploration phase (CSA), that given the optimal solution. Therefore, we can conclude our proposed algorithms for high-dimensional data in pseudocode, as expressed in Algorithm 1.

Algorithm 1: The proposed algorithm

Set the initial values of $N, AP, fl_k, t_{max}, , in_{\omega_{max}}$ *and* $in_{\omega_{min}}$
Initialize the crow position h randomly $\vec{X}_h$
Evaluate the fitness function of each crow $Fn(\vec{X})$.
Evaluate the best fitness function from $Fn(\vec{X})$ as $\vec{X}^*$
Initialize the memory of search crow $\vec{M}$
Set t: = 1. {counter initialization}.
While ($iter < iter_{max}$ number of iterations)
  Update $\vec{a}_{new}, in_\omega$, A, C and $\vec{X}^*(t)$
   for (p = 1: p ≤ N_C) do
    Randomly choose one of crows to follow q
    if rand $rand() \geq AP_r(iter)$ then // exploitation phase

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.cos(2\pi l) + \left(\vec{X}^*(iter) \times in_\omega\right) \text{ as called } HCSWOA_1$$

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.sin(2\pi l) + \left(\vec{X}^*(iter) \times in_\omega \times cos\left(7 \times \frac{iter}{iter_{max}} \times \pi\right)\right) \text{ as called } HCSWOA_2$$

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.cos(2\pi l) + \left(\vec{X}^*(iter) \times in_\omega \times sin\left(7 \times \frac{iter}{iter_{max}} \times \pi\right)\right) \text{ as called } HCSWOA_3$$

$$\vec{X}_p(iter+1) = \vec{D}'.e^{bl}.cos(2\pi l) + \left(\vec{X}^*(iter) \times in_\omega \times 0.5 \times sin\left(7 \times \frac{iter}{iter_{max}} \times \pi\right)\right) \text{ as called } HCSWOA_4$$

    Else // exploration phase

$$\vec{X}_h(iter+1) = \left(in_\omega.\vec{X}_k(iter)\right) + fl_h(iter).rand().\left[\vec{M}_q(iter) - \vec{X}_h(iter)\right]$$

    end If
   end for
  Check the feasibility of $\vec{X}(ite+1)$
  Evaluated the new position of crow $Fn(\vec{X}(ite+1))$
  Update the crow's memory $\vec{M}(ite+1)$
Until ($ite < iteMax$). {Termination criteria satisfied}.
Produce the best solution $\vec{M}$

4. Experimental Results

In this section, we choose a set of twenty-three benchmark functions for testing the ability of $HCSWOA_1$, $HCSWOA_2$, $HCSWOA_3$, and $HCSWOA_4$ in both exploitation and exploration capacities, as shown in [49]. Moreover, the benchmark functions can be divided into three groups: Functions UF1-UF7 are unimodal functions, MF1-MF6 are multimodal functions, and MFF1-MFF10 are fixed-dimension multimodal functions, respectively.

This study was coded in MATLAB R2018a with Intel HD Graphics 6000, 1536 MB, 8 GB of memory, 1600 MHz DDR3, 1,6 GHz Dual-Core Intel Core i5, macOS Big Sur, and 128 GB HDD.

Table 1 lists all parameter settings used in this study, such as the number of populations, the maximum number of iterations, and the other parameters, respectively. To examine the performance of $HCSWOA_1$, $HCSWOA_2$, $HCSWOA_3$, and $HCSWOA_4$; the experiment will proceed from the following aspects: The HCSWOA is compared with other meta-heuristic algorithms on 30 to 100-dimensional data with 500 iterations, such as CSA, PSO, GSA, WOA, GWO, SCA, ALO, DE, and DA [50]. These results were obtained by [42], [49], [51], and [52], respectively. In addition, all results are averaged over 30 independent runs in terms of average (avg.) and standard deviation (std) values.

Table 1. The planning and control components.

Algorithms	No. of population	No. of Iteration	Parameters
CSA	20, 30	500, 2000	AP = 0.1 and fl = 2
WOA	30	500	$\alpha \in [2,0]$
GWO	30	500	$\alpha \in [2,0]$
GSA	30	500	G0 = 100, alpha= 20 and final = 2%
BA	30	500	A=0.5 and r=0.5
SCA	30	500	A=2, $r_1 = r_2 = r_3 = r_4$ = rand()
ALO	30	500	k=500
DE	30	500	β_{max}=0.8, β_{min} = 0.8 and CR = 0.2
DA	30	500	s=a=c
PSO	30	500	$c_1 = c_2$=1.45 and w = [0.9,0.4]
ICSA	20, 30	500, 2000	ef =0.5, fl =2 and AP =0.1
ICSAGWO	20, 30	500, 2000	AP = 0.2, fl = 2, in_w_{max}=0.9, in_w_{min}=0.4, and a ∈ [2,0]
Ours proposed	20, 30	500, 2000	AP = 0.8, fl = , , in_w_{max}=0.9, in_w_{min}=0.4, μ=7, $\alpha \in [2,0]$

4.1 Experimental results: Comparison with our proposed algorithms and others Meta-heuristic algorithms

In this section, our objective is to present the best suitable optimal solution by comparing it with the proposed algorithms $HCSWOA_1$, $HCSWOA_2$, $HCSWOA_3$, and $HCSWOA_4$ and other meta-heuristic algorithms, such as CSA and WOA algorithms, on 30, 50, and 100-dimensional data with 500 iterations. Moreover, these results were run 30 times on each function and the best result will be highlighted in bold, as shown in Tables 2–5.

The UF1-UF7 unimodal benchmark functions are to test the performance of exploration ability. The results of the proposed algorithm, $HCSWOA_4$, can find optimal solutions for UF1–UF4 functions when the dimensions increase, as shown in the 30–100 dimensions range, meaning the performance results of $HCSWOA_4$ can find the optimal solution in terms of the exploitation phase. As shown in the table, the results of $HCSWOA_1$ are second best on UF6 and UF7 functions.

Table 2. The performance comparison of proposed algorithm (HCSWOA) with other algorithms on 30D with 500 iterations

Function	$HCSWOA_1$		$HCSWOA_2$		$HCSWOA_3$		$HCSWOA_4$		CSA		WOA	
	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
UF1	3.80E-134	2.08E-133	**8.81E-141**	4.82E-140	2.065E-142	1.02E-141	**7.22E-150**	**2.54E-149**	8.31E+00	3.63E+00	1.19E-73	6.44E-73
UF2	3.85E-73	1.13E-72	9.27E-72	3.81E-71	**2.1454E-76**	**9.32E-76**	1.79E-71	9.81E-71	3.23E+00	9.08E-01	3.40E-48	1.86E-47
UF3	1.09E-131	5.97E-131	1.01E-140	5.55E-140	2.964E-139	1.15E-138	**5.68E-142**	**3.10E-141**	3.36E+02	1.09E+02	4.80E+04	1.27E+04
UF4	1.82E-63	9.95E-63	6.93E-67	3.74E-66	3.5735E-71	1.82E-70	**1.92E-72**	**1.05E-71**	6.54E+00	1.35E+00	4.86E+01	2.58E+01
UF5	**2.86E+01**	1.11E-01	2.87E+01	**1.10E-02**	2.87E+01	3.99E-02	2.87E+01	1.18E-02	3.58E+02	1.49E+02	2.78E+01	3.66E-01
UF6	**1.44E-01**	4.03E-02	2.61E-01	9.13E-02	2.54E-01	8.09E-02	1.93E-01	6.58E-02	8.94E+00	3.52E+00	4.49E-01	2.05E-01
UF7	3.06E-04	2.01E-04	2.69E-04	2.40E-04	3.33E-04	3.02E-04	2.43E-04	1.91E-04	5.19E-02	2.37E-02	3.98E-03	4.74E-03
MF1	-2.27E+03	6.30E+02	-1.98E+03	5.74E+02	-2.03E+03	4.73E+02	-2.23E+03	5.91E+02	-6.52E+03	8.54E+02	**-1.03E+04**	**1.61E+03**
MF2	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.95E+01	8.90E+00	**0.00E+00**	**0.00E+00**
MF3	**8.88E-16**	**1.00E-31**	**8.88E-16**	4.01E-31	**8.88E-16**	4.01E-31	**8.88E-16**	4.01E-31	3.83E+00	7.10E-01	4.20E-15	2.46E-15
MF4	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.07E+00	2.47E-02	**0.00E+00**	**0.00E+00**
MF5	**5.45E-03**	1.72E-03	1.14E-02	4.44E-03	1.15E-02	6.39E-03	8.51E-03	2.94E-03	4.06E+00	1.50E+00	1.95E-02	9.87E-03
MF6	**2.13E-01**	5.02E-02	2.47E-01	6.63E-02	2.67E-01	7.99E-02	2.32E-01	7.18E-02	4.26E+00	6.39E+00	4.65E-01	2.49E-01

Table 3. The performance comparison of proposed algorithm (HCSWOA) with other algorithms on fixed-multimodal functions with 500 iterations

Function	$HCSWOA_1$		$HCSWOA_2$		$HCSWOA_3$		$HCSWOA_4$		CSA		WOA	
	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
MFF1	1.15E+01	2.84E+00	1.22E+01	1.62E+00	1.26E+01	1.69E-01	1.22E+01	1.61E+00	**1.05E+00**	**2.03E-01**	2.54E+00	2.62E+00
MFF2	3.97E-04	9.62E-05	3.74E-04	6.26E-05	3.85E-04	6.80E-05	3.73E-04	5.94E-05	4.48E-04	3.32E-04	**6.43E-04**	**3.86E-04**
MFF3	**-1.03E+00**	8.72E-03	-1.02E+00	1.29E-02	-1.02E+00	1.18E-02	-1.02E+00	1.06E-02	**-1.03E+00**	**6.78E-16**	**-1.03E+00**	**6.78E-16**
MFF4	**3.98E-01**	6.61E-04	3.99E-01	1.58E-03	4.00E-01	2.50E-03	3.99E-01	9.48E-04	**3.98E-01**	**1.69E-16**	**3.98E-01**	4.84E-06
MFF5	3.04E+00	3.52E-02	5.05E+00	6.80E+00	3.57E+00	9.46E-01	**3.32E+00**	4.71E-01	**3.00E+00**	**0.00E+00**	**3.00E+00**	7.85E-05
MFF6	**-3.86E+00**	7.64E-03	-3.84E+00	2.29E-02	-3.85E+00	1.12E-02	**-3.86E+00**	6.74E-03	**-3.86E+00**	**3.16E-15**	-3.85E+00	1.15E-02
MFF7	-3.26E+00	6.06E-02	-3.09E+00	1.93E-01	-3.12E+00	1.34E-01	-3.19E+00	1.19E-01	**-3.31E+00**	**4.13E-02**	-3.26E+00	9.32E-02
MFF8	-6.30E+00	1.75E+00	-5.00E+00	3.88E-01	-6.26E+00	1.50E+00	-6.70E+00	1.76E+00	**-9.07E+00**	**2.51E+00**	-8.28E+00	2.73E+00
MFF9	-6.00E+00	1.43E+00	-4.91E+00	3.58E-01	-6.06E+00	1.35E+00	-7.49E+00	1.92E+00	**-9.07E+00**	**2.51E+00**	-8.10E+00	2.88E+00
MFF10	-6.59E+00	1.64E+00	-5.05E+00	8.10E-02	-6.04E+00	1.21E+00	-7.53E+00	1.76E+00	**-1.02E+01**	**9.63E-01**	-7.10E+00	3.33E+00

Table 4. The performance comparison of proposed algorithm (HCSWOA) with other algorithms on 50D with 500 iterations

Function	$HCSWOA_1$		$HCSWOA_2$		$HCSWOA_3$		$HCSWOA_4$		CSA		WOA	
	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
UF1	4.05E-134	2.21E-133	2.37E-141	1.20E-140	3.24E-142	1.77E-141	**7.71E-147**	**3.60E-146**	4.17E-02	2.03E-02	2.32E-72	7.48E-72
UF2	3.72E-71	1.65E-70	1.03E-72	5.55E-72	**1.28E-76**	**4.93E-76**	1.44E-71	7.90E-71	3.34E+00	9.24E-01	3.09E-51	7.55E-51
UF3	1.96E-129	1.07E-128	5.25E-137	2.85E-136	1.44E-133	7.88E-133	**1.53E-141**	**8.06E-141**	1.61E+02	3.16E+01	1.98E+05	4.10E+04
UF4	1.64E-67	7.74E-67	2.93E-74	7.94E-74	2.58E-70	1.38E-69	**1.85E-77**	**8.17E-77**	5.57E+00	1.27E+00	6.96E+01	2.18E+01
UF5	4.85E+01	3.20E-02	4.85E+01	7.21E-03	4.85E+01	8.07E-03	4.85E+01	8.84E-03	1.27E+02	8.49E+01	**4.81E+01**	**3.93E-01**
UF6	**2.99E-01**	**7.21E-02**	4.76E-01	1.32E-01	4.22E-01	9.54E-02	3.57E-01	7.28E-02	3.81E-02	1.23E-02	1.29E+00	4.48E-01
UF7	**1.80E-04**	**1.48E-04**	2.05E-04	1.66E-04	2.98E-04	2.41E-04	1.98E-04	2.08E-04	5.12E-02	1.76E-02	3.03E-03	3.44E-03
MF1	-2.84E+03	8.15E+02	-2.39E+03	6.98E+02	-2.56E+03	7.53E+02	-2.92E+03	6.46E+02	-1.09E+04	1.21E+03	**-1.73E+04**	**2.96E+03**
MF2	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	4.12E+01	1.28E+01	0.00E+00	0.00E+00
MF3	**8.88E-16**	**1.00E-31**	**8.88E-16**	4.01E-31	**8.88E-16**	4.01E-31	**8.88E-16**	4.01E-31	4.39E+00	9.60E-01	4.44E-15	2.28E-15
MF4	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.67E-01	4.40E-02	6.60E-03	3.61E-02
MF5	**6.45E-03**	**1.39E-03**	1.25E-02	5.87E-03	1.29E-02	6.40E-03	7.93E-03	2.42E-03	4.81E+00	1.48E+00	3.09E-02	1.35E-02
MF6	**4.51E-01**	**1.12E-01**	4.44E-01	9.26E-02	5.65E-01	1.19E-01	4.93E-01	1.39E-01	3.67E+01	2.88E+01	1.25E+00	5.19E-01

Table 5. The performance comparison of proposed algorithm (HCSWOA) with other algorithms on 100D with 500 iterations

Function	**HCSWOA$_1$**		**HCSWOA$_2$**		**HCSWOA$_3$**		**HCSWOA$_4$**		**CSA**		**WOA**	
	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
UF1	8.90E-133	4.79E-132	6.36E-141	3.48E-140	1.20E-139	5.08E-139	**7.08E-143**	**3.50E-142**	7.84E+00	1.96E+00	8.61E-72	4.42E-71
UF2	2.86E-71	6.92E-71	1.94E-70	1.06E-69	6.67E-73	3.53E-72	**2.42E-74**	**1.32E-73**	8.78E+00	1.57E+00	8.90E-51	3.60E-50
UF3	1.65E-132	8.87E-132	4.56E-136	2.46E-135	3.93E-131	2.15E-130	**3.25E-139**	**1.75E-138**	1.22E+03	1.50E+02	1.03E+06	2.73E+05
UF4	1.75E-67	9.42E-67	7.07E-71	3.82E-70	1.19E-72	4.66E-72	**6.89E-72**	**2.67E-71**	9.00E+00	1.07E+00	7.86E+01	2.36E+01
UF5	9.80E+01	1.41E-02	9.80E+01	1.61E-02	9.80E+01	9.52E-03	9.80E+01	1.44E-02	6.63E+02	1.30E+02	9.82E+01	2.77E-01
UF6	8.97E-01	2.06E-01	8.54E-01	1.62E-01	9.74E-01	2.26E-01	7.23E-01	1.66E-01	7.83E+00	2.06E+00	4.26E+00	1.26E+00
UF7	3.42E-04	3.04E-04	3.22E-04	2.57E-04	3.00E-04	2.27E-04	2.08E-04	1.81E-04	1.46E-01	1.83E-02	4.04E-03	6.91E-03
MF1	-3.66E+03	8.52E+02	-3.57E+03	1.03E+03	-3.79E+03	8.97E+02	-3.95E+03	8.66E+02	-1.94E+04	2.41E+03	-3.60E+04	5.99E+03
MF2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	0.00E+00	8.74E+01	2.05E+01	0.00E+00	0.00E+00
MF3	8.88E-16	3.94E-31	8.88E-16	4.01E-31	8.88E-16	4.01E-31	**8.88E-16**	4.01E-31	5.31E+00	9.31E-01	4.91E-15	2.42E-15
MF4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	0.00E+00	1.06E+00	2.63E-02	1.05E-02	5.76E-02
MF5	1.03E-02	2.66E-03	1.26E-02	6.37E-03	1.23E-02	5.19E-03	**8.60E-03**	3.01E-03	5.30E+00	9.95E-01	4.30E-02	1.61E-02
MF6	1.22E+00	2.23E-01	1.02E+00	2.00E-01	1.15E+00	2.58E-01	**1.17E+00**	2.97E-01	1.16E+02	1.77E+01	3.01E+00	9.18E-01

Table 6. The performance comparison of proposed algorithm (HCSWOA) with other algorithms on 30D with 500 iterations

Functions		**PSO**	**GSA**	**GWO**	**ABC**	**SCA**	**DA**	**ALO**	**DE**	**HCSWOA_4**
UF1	AVG	5.21E-06	4.41E-15	1.67E-27	5.33E+00	3.49E-11	1.65E+01	8.13E-09	5.43E-04	**7.22E-150**
	STD	2.42E-05	2.67E-15	3.14E-27	1.16E+00	1.85E-10	2.58E+01	3.91E-09	1.67E-04	**2.54E-149**
UF2	AVG	6.39E-02	2.31E-02	9.79E-17	2.89E+01	3.46E-09	1.74E+00	5.03E-01	2.53E-03	**1.79E-71**
	STD	1.25E-01	9.25E-02	5.97E-17	1.66E+01	8.23E-09	9.09E-01	9.16E-01	4.35E-04	**9.81E-71**
UF3	AVG	9.36E+01	9.48E+02	4.92E-06	5.37E+04	3.37E-02	3.22E+02	4.01E-02	2.95E+04	**5.68E-142**
	STD	1.23E+02	3.08E+02	9.46E-06	7.37E+03	1.68E-01	8.12E+02	1.02E-01	4.46E+03	**3.10E-141**
UF4	AVG	2.32E+00	6.32E+00	7.05E-07	5.20E+01	1.14E-03	2.91E+00	2.75E-03	1.29E+01	**1.92E-72**
	STD	1.25E+00	2.22E+00	7.34E-07	4.12E+00	4.00E-03	2.14E+00	4.23E-03	1.44E+00	**1.05E-71**
UF5	AVG	3.94E+01	9.34E+01	2.69E+01	1.22E+05	**7.48E+00**	1.67E+03	1.57E+02	1.63E+02	2.87E+01
	STD	2.96E+01	9.16E+01	7.08E-01	5.35E+04	3.28E-01	3.24E+03	3.83E+02	4.45E+01	**1.18E-02**
UF6	AVG	**1.30E-06**	1.37E+01	7.73E-01	4.89E+00	3.96E-01	1.04E+01	8.41E-09	5.58E-04	1.93E-01
	STD	**6.48E-06**	2.58E+01	4.80E-01	1.41E+00	1.50E-01	1.77E+01	4.80E-09	2.18E-04	6.58E-02
UF7	AVG	2.88E-02	8.94E-02	2.21E-03	2.14E-01	3.84E-03	3.15E-02	2.84E-02	5.25E-02	**2.43E-04**
	STD	1.15E-02	3.72E-02	1.04E-03	4.86E-02	4.98E-03	2.47E-02	1.68E-02	1.11E-02	**1.91E-04**
MF1	AVG	-6.53E+03	-2.67E+03	-6.15E+03	-4.17E+62	-2.11E+03	-2.73E+03	-2.39E+03	**-9.44E+03**	-2.23E+03
	STD	7.31E+02	5.84E+02	9.17E+02	1.02E+63	1.50E+02	3.20E+02	5.41E+02	**3.74E+02**	5.91E+02
MF2	AVG	5.87E+01	2.78E+01	3.70E+00	2.20E+02	3.59E-01	3.05E+01	2.06E+01	8.46E+01	**0.00E+00**
	STD	1.66E+01	6.34E+00	4.48E+00	1.35E+01	1.51E+00	1.22E+01	9.68E+00	7.75E+00	**0.00E+00**
MF3	AVG	1.66E+00	6.21E-02	1.06E-13	2.78E+00	4.29E-05	2.68E+00	3.90E-01	6.27E-03	**8.88E-16**
	STD	6.82E-01	2.36E-01	1.74E-14	3.37E-01	1.92E-04	1.37E+00	6.69E-01	1.30E-03	**4.01E-31**
MF4	AVG	3.07E-02	2.82E+01	2.58E-03	1.05E+00	6.56E-02	4.95E-01	2.19E-01	9.51E-03	**0.00E+00**
	STD	3.93E-02	6.09E+00	9.10E-03	1.04E-02	1.47E-01	2.96E-01	1.09E-01	1.04E-02	**0.00E+00**
MF5	AVG	8.99E-02	1.98E+00	6.27E-02	3.24E+04	9.86E-02	2.65E+00	2.87E+00	**7.98E-05**	8.51E-03
	STD	1.61E-01	1.05E+00	9.64E-02	4.82E+04	3.84E-02	2.31E+00	2.26E+00	**4.03E-05**	2.94E-03
MF6	AVG	2.50E-01	9.07E+00	5.07E-01	1.41E+05	3.56E-01	1.10E+00	3.30E-03	**3.54E-04**	2.32E-01
	STD	6.34E-01	6.52E+00	1.95E-01	9.52E+04	1.07E-01	2.04E+00	8.73E-03	**1.26E-04**	7.18E-02

The MF1–MF6 multimodal benchmark functions test the performance of exploration ability. The experimental results of HCSWOA$_1$ obtained the best optimum in MF2-MF6 benchmark functions. While the results of HCSWOA$_1$ show that it is better than HCSWOA$_4$ in terms of exploitation ability, in terms of exploitation capacity, HCSWOA$_4$ outperforms HCSWOA$_1$. Finally, MFF1–MFF10 fixed-dimensional benchmark functions also test exploration capacity. The experimental results of the proposed algorithm, HCSWOA$_4$, are superior to those of HCSWOA$_2$ and HCSWOA$_3$. It is clearly evident from the performance results of HCSWOA$_4$, which imply more robustness than CSA and WOA when confronting high-dimensional data.

Table 7. The performance comparison of proposed algorithm (HCSWOA) with other algorithms on fixed-multimodal functions

Functions		**PSO**	**GSA**	**GWO**	**ABC**	**SCA**	DA	ALO	DE	**HCSWOA_4**
MFF1	AVG	3.92E+00	5.42E+00	3.55E+00	**9.98E-01**	2.12E+00	1.16E+00	2.61E+00	**9.98E-01**	1.22E+01
	STD	2.94E+00	3.09E+00	3.18E+00	**4.52E-16**	1.00E+00	5.87E-01	1.96E+00	**4.52E-16**	1.61E+00
MFF2	AVG	1.18E-03	4.41E-03	3.85E-03	9.75E-04	1.10E-03	4.47E-03	2.30E-03	6.98E-04	**3.73E-04**
	STD	3.65E-03	3.00E-03	7.52E-03	6.77E-05	3.60E-04	6.64E-03	4.95E-03	1.03E-04	**5.94E-05**
MFF3	AVG	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	**-1.03E+00**	-1.02E+00
	STD	**6.78E-16**	**6.78E-16**	**6.78E-16**	**6.78E-16**	6.61E-05	**6.78E-16**	**6.78E-16**	**6.78E-16**	1.06E-02
MFF4	AVG	**3.98E-01**	**3.98E-01**	**3.98E-01**	**3.98E-01**	4.00E-01	**3.98E-01**	**3.98E-01**	**3.98E-01**	3.99E-01
	STD	**1.69E-16**	**1.69E-16**	1.99E-04	**1.69E-16**	2.12E-03	**1.69E-16**	**1.69E-16**	**1.69E-16**	9.48E-04
MFF5	AVG	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	3.32E+00
	STD	**0.00E+00**	**0.00E+00**	3.79E-05	0.00E+00	1.80E-04	1.83E-05	0.00E+00	0.00E+00	4.71E-01
MFF6	AVG	-3.84E+00	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	-3.85E+00	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**
	STD	1.41E-01	**3.16E-15**	2.69E-03	**3.16E-15**	2.23E-03	1.13E-03	**3.16E-15**	**3.16E-15**	6.74E-03
MFF7	AVG	-3.29E+00	-3.32E+00	-3.26E+00	**-3.32E+00**	-2.87E+00	-3.24E+00	-3.27E+00	**-3.32E+00**	-3.19E+00
	STD	5.54E-02	1.81E-15	8.03E-02	**1.81E-15**	4.15E-01	9.95E-02	6.05E-02	**4.20E-04**	1.19E-01
MFF8	AVG	-5.74E+00	-6.04E+00	-8.64E+00	**-1.02E+01**	-2.84E+00	-6.94E+00	-6.69E+00	**-1.01E+01**	-6.70E+00
	STD	3.51E+00	3.68E+00	2.62E+00	**0.00E+00**	2.10E+00	2.73E+00	2.77E+00	**1.67E-01**	1.76E+00
MFF9	AVG	-6.18E+00	-1.01E+01	-1.02E+01	**-1.04E+01**	-2.99E+00	-7.37E+00	-7.14E+00	**-1.04E+01**	-7.49E+00
	STD	3.79E+00	1.30E+00	9.63E-01	**0.00E+00**	1.95E+00	3.19E+00	3.45E+00	**3.17E-03**	1.92E+00
MFF10	AVG	-5.68E+00	-1.01E+01	-1.03E+01	**-1.05E+01**	-3.58E+00	-7.15E+00	-5.05E+00	**-1.05E+01**	-7.53E+00
	STD	3.80E+00	1.60E+00	1.48E+00	**0.00E+00**	1.78E+00	3.35E+00	3.20E+00	**1.83E-04**	1.76E+00

Table 8. The performance comparison of proposed algorithm (HCSWOA) with other algorithms on 50D with 500 iterations

Functions		PSO	GSA	GWO	ABC	SCA	DA	ALO	DE	HCSWOA_4
UF1	AVG	1.81E-06	3.62E+03	9.52E-20	5.09E+00	1.03E+03	8.05E+03	1.34E+01	5.59E-04	**7.71E-147**
	STD	9.19E-06	2.09E+03	1.08E-19	9.67E-01	1.35E+03	4.16E+03	2.77E+01	1.87E-04	**3.60E-146**
UF2	AVG	6.24E-02	2.02E+00	2.42E-12	3.12E+01	7.71E-01	3.18E+01	1.39E+02	2.66E-03	**1.44E-71**
	STD	8.28E-02	1.46E+00	1.41E-12	1.69E+01	8.06E-01	1.24E+01	7.08E+01	4.61E-04	**7.90E-71**
UF3	AVG	7.90E+01	3.32E+03	4.21E-01	5.37E+04	5.26E+04	6.06E+04	1.90E+04	3.17E+04	**1.53E-141**
	STD	9.85E+01	8.25E+02	1.04E+00	7.79E+03	1.81E+04	3.03E+04	8.13E+03	3.85E+03	**8.06E-141**
UF4	AVG	2.23E+00	1.28E+01	4.15E-04	5.16E+01	6.68E+01	4.43E+01	2.50E+01	1.33E+01	**1.85E-77**
	STD	9.69E-01	2.06E+00	3.78E-04	5.00E+00	8.74E+00	1.08E+01	4.20E+00	1.42E+00	**8.17E-77**
UF5	AVG	**3.77E+01**	1.02E+03	4.75E+01	1.29E+05	6.57E+06	4.48E+06	4.75E+03	1.55E+02	4.85E+01
	STD	**2.76E+01**	4.90E+02	7.17E-01	5.55E+04	8.30E+06	4.57E+06	5.77E+03	5.60E+01	8.84E-03
UF6	AVG	**5.58E-07**	6.03E+02	2.56E+00	4.84E+00	7.77E+02	7.88E+03	7.11E+00	5.98E-04	3.57E-01
	STD	**2.88E-06**	3.13E+02	7.18E-01	1.08E+00	9.54E+02	3.47E+03	5.53E+00	1.78E-04	7.28E-02
UF7	AVG	3.26E-02	4.09E-01	3.48E-03	2.13E-01	3.66E+00	3.63E+00	7.88E-01	5.17E-02	**1.98E-04**
	STD	1.54E-02	2.56E-01	1.55E-03	4.04E-02	3.88E+00	2.87E+00	2.24E-01	1.07E-02	**2.08E-04**
MF1	AVG	-6.51E+03	-3.36E+03	-9.17E+03	-5.25E+62	-4.83E+03	-7.26E+03	**-9.05E+03**	-9.51E+03	-2.92E+03
	STD	7.85E+02	4.96E+02	1.18E+03	1.42E+63	2.44E+02	8.88E+02	**6.41E+01**	3.72E+02	6.46E+02
MF2	AVG	6.01E+01	5.64E+01	4.06E+00	2.21E+02	8.76E+01	3.51E+02	1.36E+02	8.60E+01	**0.00E+00**
	STD	2.12E+01	1.38E+01	4.93E+00	1.63E+01	5.54E+01	5.37E+01	2.65E+01	8.68E+00	**0.00E+00**
MF3	AVG	1.67E+00	1.39E+00	3.90E-11	2.84E+00	1.72E+01	1.23E+01	1.02E+01	6.40E-03	**8.88E-16**
	STD	7.45E-01	7.23E-01	2.54E-11	3.00E-01	6.46E+00	1.94E+00	2.95E+00	1.17E-03	**4.01E-31**
MF4	AVG	4.47E-02	1.28E+02	4.99E-03	1.04E+00	8.61E+00	6.12E+01	1.04E+00	8.54E-03	**0.00E+00**
	STD	5.73E-02	1.49E+01	9.53E-03	8.70E-03	1.08E+01	3.46E+01	9.87E-02	8.38E-03	**0.00E+00**
MF5	AVG	1.39E-01	3.58E+00	1.24E-01	2.52E+04	1.27E+07	5.75E+05	2.51E+01	**7.12E-05**	7.93E-03
	STD	2.97E-01	1.08E+00	6.43E-02	2.92E+04	1.97E+07	8.99E+05	7.71E+00	**2.83E-05**	2.42E-03
MF6	AVG	1.25E-01	4.89E+01	2.17E+00	1.45E+05	2.33E+07	4.63E+06	1.13E+02	**3.67E-04**	4.93E-01
	STD	3.25E-01	1.22E+01	3.26E-01	9.87E+04	3.42E+07	4.76E+06	2.17E+01	**1.49E-04**	1.39E-01

Tables 6–9 demonstrated the best suitable optimal solution by comparing it with the proposed algorithm $HCSWOA_4$ and other meta-heuristic algorithms, such as CSA, PSO, GSA, WOA, ABC, GWO, SCA, ALO, DE, and DA algorithms on 30, 50, and 100-dimensional data with 500 iterations. As a result, our proposed algorithm outperformed other meta-heuristic algorithms in the UF1–UF4 and UF7 benchmark functions in terms of exploitation capacity on different dimension sizes (30D, 50D, and 100D). In terms of exploration ability, the experimental results of $HCSWOA_4$ obtained the best optimum in MF2-MF4. However, the performance results of DE show that it is better than $HCSWOA_4$ in terms of exploitation ability in MF1, MF5, and MF6 benchmark functions, but in terms of exploitation capacity, $HCSWOA_4$ outperforms DE. Finally, MFF1–MFF10 fixed-dimensional benchmark functions also test exploration capacity. The experimental results of the proposed algorithm are superior to SCA and ALO.

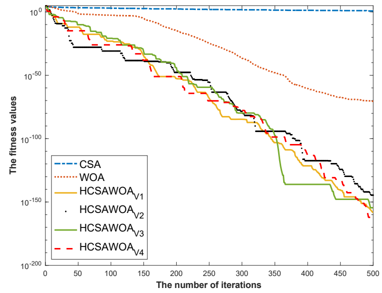
Table 9. The performance comparison of proposed algorithm (HCSWOA) with other algorithms on 100D with 500 iterations

Functions		PSO	GSA	GWO	ABC	SCA	DA	ALO	DE	HCSWOA_4
UF1	AVG	3.69E-07	3.89E+03	1.62E-12	5.24E+00	8.36E+03	1.74E+04	4.61E+03	6.13E-04	**7.08E-143**
	STD	1.52E-06	8.29E+02	1.79E-12	1.42E+00	7.39E+03	9.00E+03	1.56E+03	2.36E-04	**3.50E-142**
UF2	AVG	6.34E-02	1.75E+01	3.71E-08	3.18E+01	8.71E+00	8.98E+01	1.93E+18	2.57E-03	**2.42E-74**
	STD	1.05E-01	5.16E+00	1.18E-08	1.65E+01	6.69E+00	4.03E+01	1.06E+19	5.82E-04	**1.32E-73**
UF3	AVG	7.51E+01	1.67E+04	6.68E+02	5.35E+04	2.41E+05	2.16E+05	8.09E+04	3.12E+04	**3.25E-139**
	STD	5.67E+01	7.30E+03	8.17E+02	7.44E+03	5.61E+04	7.19E+04	2.52E+04	3.43E+03	**1.75E-138**
UF4	AVG	2.47E+00	1.87E+01	8.99E-01	5.22E+01	8.92E+01	4.90E+01	3.49E+01	1.28E+01	**6.89E-72**
	STD	1.35E+00	1.44E+00	9.98E-01	3.07E+00	3.28E+00	8.49E+00	5.60E+00	1.61E+00	**2.67E-71**
UF5	AVG	**5.18E+01**	8.97E+04	9.80E+01	1.46E+05	1.23E+08	1.33E+07	8.23E+05	1.53E+02	9.80E+01
	STD	**3.69E+01**	4.09E+04	5.76E-01	5.15E+04	5.20E+07	8.35E+06	7.50E+05	4.81E+01	1.44E-02
UF6	AVG	**2.23E-04**	5.28E+03	9.83E+00	5.21E+00	1.12E+04	1.72E+04	5.29E+03	5.67E-04	7.23E-01
	STD	**1.22E-03**	1.19E+03	1.09E+00	1.28E+00	7.44E+03	9.23E+03	2.43E+03	1.62E-04	1.66E-01
UF7	AVG	2.90E-02	3.92E+00	7.03E-03	2.04E-01	1.28E+02	1.46E+01	4.90E+00	5.18E-02	**2.08E-04**
	STD	1.12E-02	2.28E+00	3.11E-03	6.05E-02	7.06E+01	8.67E+00	1.40E+00	1.27E-02	**1.81E-04**
MF1	AVG	-6.37E+03	-4.80E+03	-1.62E+04	-7.60E+61	-6.76E+03	-1.05E+04	**-1.81E+04**	-9.35E+03	-3.95E+03
	STD	8.15E+02	8.99E+02	2.29E+03	1.41E+62	4.26E+02	1.33E+03	**0.00E+00**	3.04E+02	8.66E+02
MF2	AVG	4.93E+01	1.78E+02	1.10E+01	2.23E+02	2.50E+02	7.51E+02	3.71E+02	8.46E+01	**0.00E+00**
	STD	1.51E+01	2.92E+01	5.74E+00	1.57E+01	1.23E+02	1.24E+02	7.44E+01	8.12E+00	**0.00E+00**
MF3	AVG	1.60E+00	4.49E+00	1.27E-07	2.85E+00	1.98E+01	1.24E+01	1.38E+01	6.56E-03	**8.88E-16**
	STD	6.31E-01	7.88E-01	4.79E-08	3.78E-01	3.18E+00	2.26E+00	1.25E+00	1.09E-03	**4.01E-31**
MF4	AVG	3.59E-02	6.85E+02	4.10E-03	1.04E+00	9.39E+01	1.47E+02	3.27E+01	7.38E-03	**0.00E+00**
	STD	4.23E-02	3.74E+01	9.47E-03	1.00E-02	5.13E+01	8.23E+01	1.38E+01	7.66E-03	**0.00E+00**
MF5	AVG	1.04E-01	1.01E+01	2.98E-01	4.10E+04	3.39E+08	3.47E+06	4.10E+02	**8.24E-05**	8.60E-03
	STD	2.17E-01	2.95E+00	7.67E-02	4.22E+04	1.18E+08	2.45E+06	1.22E+03	**5.30E-05**	3.01E-03
MF6	AVG	4.86E-02	2.03E+03	6.75E+00	1.39E+05	5.28E+08	2.55E+07	1.02E+05	**3.27E-04**	1.17E+00
	STD	1.40E-01	2.56E+03	2.80E-01	1.17E+05	2.38E+08	2.48E+07	1.27E+05	**1.18E-04**	2.97E-01

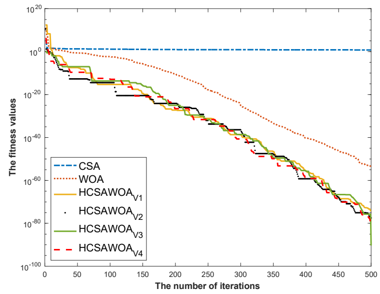
4.2 Experimental results: Convergence speed

In Fig. 1, we demonstrated the convergence speed curve of UF1-UF7 and MF1- MF6 benchmark functions by comparing our proposed algorithm with other meta-heuristic algorithms, such as CSA and WOA algorithms, on 30-

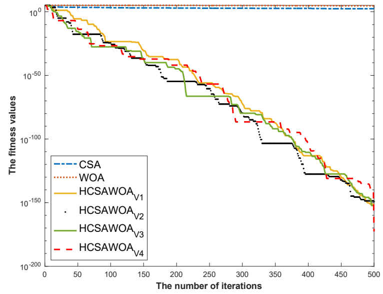
dimensional data with 500 iterations. The convergence speed curve reveals that our proposed algorithms can improve the weaknesses of the original CSA and also be quite stable to find the minimum solution to global optimization problems when compared with others.



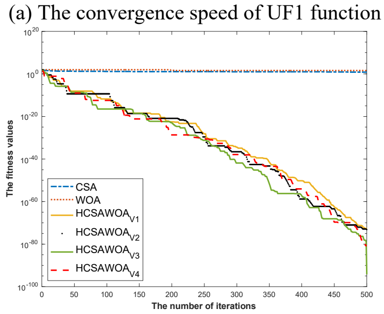
(a) The convergence speed of UF1 function



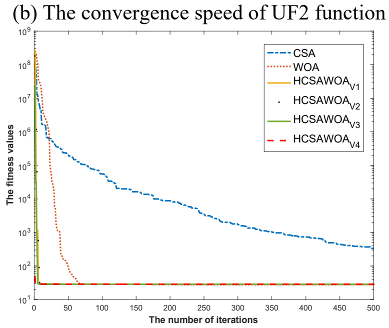
(b) The convergence speed of UF2 function



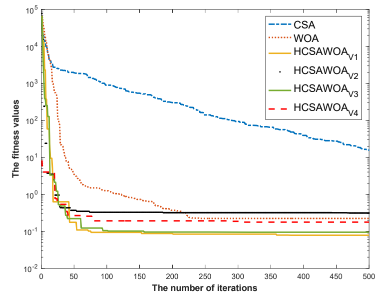
(c) The convergence speed of UF3 function



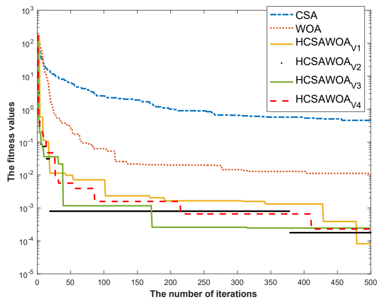
(d) The convergence speed of UF4 function



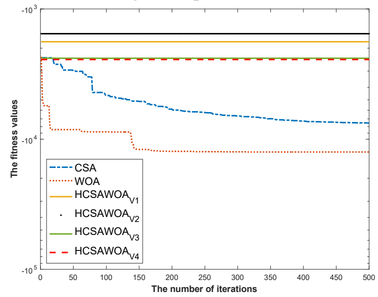
(e) The convergence speed of UF5 function



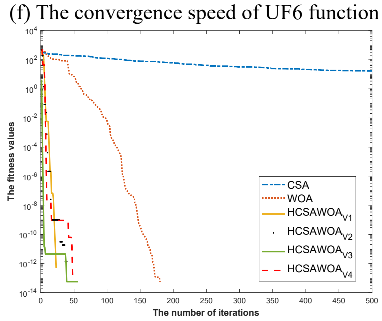
(f) The convergence speed of UF6 function



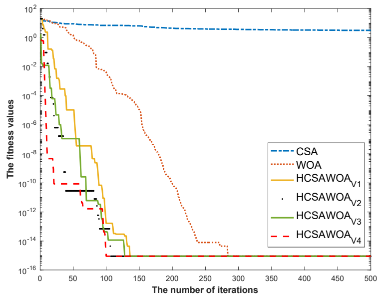
(g) The convergence speed of UF7 function



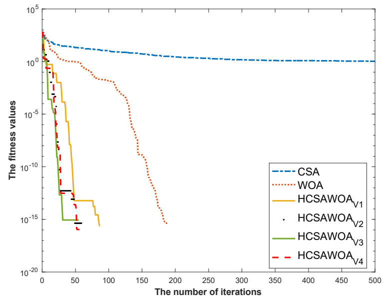
(h) The convergence speed of MF1 function



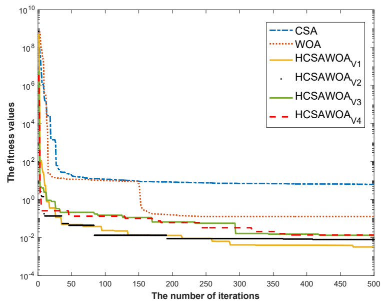
(i) The convergence speed of MF2 function



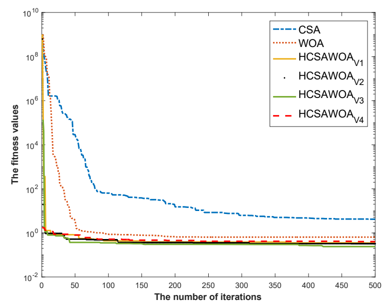
(j) The convergence speed of MF3 function



(k) The convergence speed of MF4 function



(l) The convergence speed of MF5 function



(m) The convergence speed of MF6 function

Fig. 1. The comparison of the convergence speed of the proposed algorithms with other algorithms

e-ISSN : 0976-5166
p-ISSN : 2231-3850

4.3 Experimental results: Statistical Analysis

In this study, we utilized the Wilcoxon rank-sum test, which is a statistical method based on nonparametric tests that employs the fitness value of our proposed algorithm, $HCSWOA_4$. The test compares two algorithms or repeats measurements using our proposed algorithm with 5% accuracy from a pair of samples. According to the test, we used a confidence level of 0.95 for statistical analysis, and p-values greater than or equal to 0.05 are shown in bold, as expressed in Table 10.

Table 10. The performance comparison of proposed algorithm on wilcoxon rank sum test 30D, 50D and 100D

	30D		50D		100D	
Functions	**CSA**	**WOA**	**CSA**	**WOA**	**CSA**	**WOA**
UF1	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
UF2	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
UF3	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
UF4	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
UF5	2.98E-11	5.51E-10	3.00E-11	3.98E-04	3.00E-11	2.06E-05
UF6	3.02E-11	1.73E-07	3.02E-11	3.02E-11	3.02E-11	3.69E-11
UF7	3.02E-11	1.07E-07	3.02E-11	9.26E-09	3.02E-11	5.19E-07
MF1	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
MF2	1.21E-12	**NaN**	1.21E-12	**NaN**	1.21E-12	**NaN**
MF3	1.21E-12	1.09E-08	1.21E-12	9.84E-10	1.21E-12	3.57E-10
MF4	1.21E-12	**NaN**	1.21E-12	3.34E-01	1.21E-12	3.34E-01
MF5	3.02E-11	8.20E-07	3.02E-11	6.07E-11	3.02E-11	3.02E-11
MF6	3.02E-11	6.36E-05	3.02E-11	2.23E-09	3.02E-11	3.82E-10

In the Table, the p-value scores obtained by Wilcoxon rank-sum test with 5% accuracy from a pair of samples for two algorithms of 30 independent runs to test the null hypothesis for benchmark functions are summarized for different problem dimension sizes (30D, 50D, and 100D). Looking at the results in these tables, p-values show that there are significant differences between the results obtained by the CSA, WOA and the proposed $HCSWOA_4$ for all benchmark problems. However, there is no significant difference between $HCSWOA_4$ and WOA for only 2 benchmarks in the 30D, p-value table results, only a benchmark in the 50D and 100D p-value table results.

4.4 Experimental results: Comparison with others CSA for High-Dimensional Data Problems

In this section, our objective is to present the best suitable optimal solution by comparing it with the proposed algorithm $HCSWOA_4$ and other meta-heuristic algorithms, such as CSA, ICSA, ICSA, ICSAGWO [53] algorithms on 50 to 500 dimensional data with 2000 iterations. Moreover, these results were run 30 times on each function, as shown in Table 11-12.

According to the performance of our proposed algorithm, it is able to explore the optimal results for high-dimensional data as the dimensions increase to 50 and 500 in both exploration and exploitation capacities. It means our proposed method has consistent performance with increasing dimensional data. Furthermore, the performance of $HCSWOA_4$ confirms that it achieves balance in two crucial search optimization problems and is robust when manipulating high dimensions.

Table 11. The experimental results for 50 to 200-dimensional data.

Function	50				100				200			
	CSA	**ICSA**	**ICSAGWO**	**HCSWOA**	**CSA**	**ICSA**	**ICSAGWO**	**HCSWOA**	**CSA**	**ICSA**	**ICSAGWO**	**HCSWOA**
UF-1	4.17E-02	1.22E-27	**0.00E+00**	**0.00E+00**	7.84E+00	2.39E-27	**0.00E+00**	**0.00E+00**	1.75E+02	6.71E-28	**0.00E+00**	**0.00E+00**
UF-2	3.34E+00	7.38E-19	**0.00E+00**	8.65E-282	8.78E+00	1.82E-25	**0.00E+00**	9.10E-288	2.47E+01	2.19E-43	**1.20E-316**	3.95E-289
UF-3	1.61E+02	1.74E-26	**0.00E+00**	**0.00E+00**	1.22E+03	5.40E-26	**0.00E+00**	**0.00E+00**	5.72E+03	5.58E-27	**0.00E+00**	**0.00E+00**
UF-4	5.57E+00	9.99E-15	**1.03E-306**	1.09E-292	9.00E+00	1.11E-14	**1.74E-296**	2.40E-289	1.13E+01	1.97E-14	**8.21E-286**	**4.14E-290**
UF-5	1.27E+02	4.85E+01	**4.78E+01**	4.85E+01	6.63E+02	9.82E+01	9.82E+01	**9.80E+01**	3.82E+03	1.98E+02	1.98E+02	**1.97E+02**
UF-6	3.81E-02	**0.00E+00**	5.78E+00	3.33E-01	7.83E+00	**0.00E+00**	1.76E+01	6.48E-01	1.76E+02	**0.00E+00**	4.31E+01	1.45E+00
UF-7	5.12E-02	9.87E-05	**4.61E-05**	5.72E-05	1.46E-01	1.35E-04	**2.75E-05**	4.86E-05	4.43E-01	6.41E-05	**3.16E-05**	4.42E-05
MF-2	4.12E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**	8.74E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**	3.23E+02	**0.00E+00**	**0.00E+00**	**0.00E+00**
MF-3	4.39E+00	4.91E-15	**8.88E-16**	**8.88E-16**	5.31E+00	1.84E-15	**8.88E-16**	**8.88E-16**	5.81E+00	**8.88E-16**	**8.88E-16**	**8.88E-16**
MF-4	1.67E-01	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.06E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.58E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**
MF-5	4.81E+00	3.30E-02	2.84E-01	**7.90E-03**	5.30E+00	7.77E-02	6.82E-01	**8.56E-03**	5.33E+00	1.17E-02	9.17E-01	**8.42E-03**
MF-6	3.67E+01	1.40E+00	3.84E+00	**3.95E-01**	1.16E+02	9.67E+00	9.34E+00	**1.07E+00**	2.25E+02	1.99E+01	1.96E+01	**2.40E+00**

Table 12. The experimental results for 500-dimensional data.

Function	CSA	ICSA	ICSAGWO	HCSWOA
UF-1	2.31E+02	5.83E-29	**0.00E+00**	**0.00E+00**
UF-2	4.62E+01	-	1.53E-310	2.64E-296
UF-3	1.49E+04	9.69E+00	**0.00E+00**	**0.00E+00**
UF-4	1.18E+01	-	7.12E-275	1.36E-295
UF-5	4.56E+03	3.00E+03	4.99E+02	**4.94E+02**
UF-6	2.43E+02	**7.60E-02**	1.18E+02	5.04E+00
UF-7	8.27E-01	-	**4.70E-05**	6.10E-05
MF-2	8.39E+02	-	**0.00E+00**	**0.00E+00**
MF-3	4.77E+00	5.72E+00	**8.88E-16**	**8.88E-16**
MF-4	3.10E+00	2.04E+01	**0.00E+00**	**0.00E+00**
MF-5	3.45E+00	-	1.09E+00	**1.07E-02**
MF-6	4.75E+02	3.74E+01	4.97E+01	**7.12E+00**

5. Feature Selection for HCSWOA

Generally, feature selection (FS) is a multi-objective optimization problem that deals with high-dimensional data. In addition, FS is a crucial problem in the pattern recognition and machine learning areas, which is a preprocessing step of the datasets as dimensionality reduction techniques for prediction or classification by removing redundant and irrelevant features. Moreover, they also help in reducing the computational load and increasing the classification accuracy. The goal of FS is to strike a balance between minimizing the number of selected features and maximizing classification accuracy.

In the search space of the FS problem, it is essential to convert the continuous search space (or the position of the crow individual) of $HCSWOA_4$ to a binary version as 0 or 1. In the FS, the value of the binary vector is equal to 1, which means the feature is selected, whereas the value is 0, meaning the corresponding feature is unselected. In this way, the number of features will be reduced without affecting the classification performance.

In this study, we proposed a way to achieve selecting a subset of significant features that can be divided into two strategies without modifying their search space, such as transfer functions and mutation operators, which are called $HCSWOA_4$-V and $HCSWOA_4$-M, respectively. In the case of the $HCSWOA_4$-S, it utilizes the positions of the search agents and converts them into a binary vector in each dimension by a V-shaped transfer function, as shown in Eq. (15).

$$V\left(X_{p,j}(iter)\right) = \frac{1}{\sqrt{1 + X_{p,j}(iter)^2}} \quad (15)$$

Where $X_{p,j}(iter)$ is the position of the p-th search agent (the crow individual) in the j-th dimension at the current iteration iter.

After transformation to binary search space, the set of the best solutions or features will be selected or unselected, which can be calculated, as expressed in Eq. (16).

$$X_{p,j}(iter) = \begin{cases} 0 & if\ rand\ < V\left(X_{p,j}(iter)\right) \\ 1 & if\ rand \geq V\left(X_{p,j}(iter)\right) \end{cases} \quad (16)$$

Where rand ∈ [0,1] which indicates a random threshold. if $X_{p,j}(iter)$ = 1 represents the value element is selected as a relevant attribute while where $X_{p,j}(iter)$= 0 indicates the j-th corresponding element is ignored. For example, if $X_{p,j}(iter)$= [0.4, 0.6, 0.2, 0.7], $S\left(X_{p,j}(iter)\right)$= [0.5987, 0.6457, 0.5498, 0.6682], and rand = 0.63, the output of Eq. (16) is $X_{p,j}(iter)$ = [0,1,0,1], that means the first and third features will be discarded.

Then, the $HCSWOA_4$-M employs the search agents and converts them into the binary vector in each dimension by Eq. (16) to select the feature subsets. Next, the mutation operator will start as follows:

• The selected feature value represents 0 in the binary vector; in the mutation operator, it should be inverted to 1;
• and, the selected feature value represents 1; therefore, the operator inverts it to 0.

Table 13. The characterize details of 18 UCI standard and two-DNA microarray datasets

Dataset No.	Name	No. of features	No. of Samples
D1	Breastcancer	9	699
D2	BreastEW	30	569
D3	CongressEW	16	435
D4	Exactly	13	1000
D5	Exactly2	13	1000
D6	HeartEW	13	270
D7	IonosphereEW	34	351
D8	KrvskpEW	36	3196
D9	Lymphography	18	148
D10	M-of-n	13	1000
D11	PenglungEW	325	73
D12	SonarEW	60	208
D13	SpectEW	22	267
D14	Tic-tac-toe	9	958
D15	Vote	16	300
D16	WaveformEW	40	5000
D17	WineEW	13	178
D18	Zoo	16	101
MD1	Colon	2000	62
MD2	Leukemia	7129	72

We chose eighteen datasets from the UCI standard datasets [54] and two DNA-microarray datasets to evaluate the performance of our proposed method. The detailed distributions of name, the number of samples, and the number of features for each dataset are outlined in Table 13. In this study, we apply our proposed algorithm to wrapper-based feature selection. It utilizes a 10-fold cross-validation method, which is used to divide the dataset into training and testing sets [55]. Our experiments utilize the K-Nearest Neighbor (KNN) classifier to evaluate the significant subset of features, and the best choice of K was at 5, which was selected as the best performing experimental result on all the datasets [20].

5.1 Fitness Function

The fitness function is used to evaluate each solution (or position) as X. In the case of global optimization problems, the best solution is evaluated using the minimum fitness value, or min(f(X)) function. On the other hand, the feature selection problem utilizes fitness functions by combining two objectives into one by setting a weight factor for balancing between maximizing the classification accuracy and minimizing the number of selected features, as shown in Eq. (17) [21].

$$fitness = \alpha\gamma_R(D) + \beta\frac{|SF|}{|NF|} \quad (17)$$

where $\gamma_R(D)$ is the error rate of the classification accuracy of the KNN classifier. Furthermore, $|SF|$represents the cardinality of the selected feature subset and $|NF|$represents the total number of features in the original dataset, α and β are two parameters corresponding to the importance of classification quality and selected feature subset size, α = 0.99 and β = 0.01, respectively.

Table 14. Average Acc, SF and Time by different feature selection algorithms on 18 UCI standard datasets

DB.	Fitness			Classification Accuracy			No. of Selected Features			Time (minus)		
	BCSA	Ours-V	Ours-M	BCSA	Ours-V	Ours-M	BCSA	Ours-V	Ours-M	BCSA	Ours-V	Ours-M
D1	3.51E-02	3.33E-02	**3.08E-02**	0.970	0.971	**0.975**	5.65	**4.30**	5.75	2.26	**0.51**	0.82
D2	5.45E-02	5.31E-02	**4.77E-02**	0.949	0.948	**0.954**	11.65	**3.65**	5.30	2.25	**0.49**	0.82
D3	4.17E-02	4.30E-02	**3.78E-02**	0.961	0.958	**0.965**	5.00	**2.30**	4.60	2.06	**0.61**	0.79
D4	1.92E-01	2.44E-01	**8.00E-02**	0.814	0.757	**0.924**	10.85	**4.30**	6.05	2.39	**0.50**	0.89
D5	2.41E-01	**2.40E-01**	**2.40E-01**	**0.758**	**0.758**	**0.758**	1.50	**1.00**	**1.00**	2.51	**0.40**	0.87
D6	1.90E-01	1.78E-01	**1.51E-01**	0.811	0.823	**0.851**	4.85	**4.20**	4.70	2.01	**0.45**	0.75
D7	1.21E-01	9.44E-02	**7.64E-02**	0.881	0.906	**0.924**	10.95	**2.95**	3.65	2.00	**0.43**	0.76
D8	4.18E-02	6.22E-02	**3.23E-02**	0.967	0.941	**0.973**	34.15	**13.20**	20.90	6.29	**1.15**	1.64
D9	1.65E-01	1.70E-01	**1.44E-01**	0.838	0.832	**0.859**	8.35	**6.05**	8.50	2.79	**0.63**	1.01
D10	9.96E-02	1.18E-01	**9.16E-03**	0.908	0.886	**0.996**	11.55	**6.25**	6.70	2.45	**0.62**	0.89
D11	1.05E-01	7.97E-02	**8.39E-02**	0.898	0.921	**0.918**	143.35	**54.60**	82.65	2.84	**0.91**	0.95
D12	1.51E-01	1.35E-01	**1.28E-01**	0.852	0.867	**0.875**	23.90	**19.90**	25.95	1.95	**0.59**	0.74
D13	1.90E-01	1.88E-01	**1.69E-01**	0.814	0.813	**0.833**	13.90	**6.45**	9.40	2.13	**0.44**	0.78
D14	1.67E-01	2.04E-01	**1.72E-01**	0.841	0.800	**0.835**	9.00	**5.70**	7.25	2.81	**0.55**	0.84
D15	5.49E-02	4.96E-02	**4.47E-02**	0.948	0.951	**0.957**	4.95	**1.20**	2.85	2.18	**0.39**	0.77
D16	1.89E-01	1.91E-01	**1.75E-01**	0.818	0.812	**0.830**	37.80	**20.50**	24.65	14.33	**2.12**	2.62
D17	6.14E-02	5.16E-02	**4.54E-02**	0.942	0.951	**0.959**	5.65	**4.60**	6.60	2.11	**0.45**	0.74
D18	6.14E-02	6.72E-02	**4.97E-02**	0.943	0.936	**0.954**	8.10	**6.40**	7.45	2.20	**0.64**	0.90
AVG	1.20E-01	1.22E-01	**9.54E-02**	0.884	0.879	**0.908**	19.51	**9.31**	13.00	3.20	**0.66**	0.98
Rank	2	3	1	2	3	1	3	1	2	3	1	2

Note* **Ours-V:** HCSWOA$_4$-V **Ours-M:** HCSWOA$_4$-M

In this study, the population size is fixed to 10, whereas the number of maximum iterations is set to 100, and the results are averaged over 20 independent runs to achieve statistically average results. The performance of the proposed $HCSWOA_4$-V and $HCSWOA_4$-M is evaluated and compared with other meta-heuristics in terms of the average classification accuracy (Acc), the number of selected features (SF), fitness values (Fit), and CPU computational time in minutes (Time).

To verify the performance of our proposed algorithms, the experiment will be conducted under the following three aspects: (1) $HCSWOA_4$-V and $HCSWOA_4$-M algorithms are compared to CSA algorithms.These results of CSA were obtained from . In the $HCSWOA_4$-M, mutation ratio and mutation percentage are set as 0.1 and 0.3, respectively. (2) The HCSWOA-M is compared with the other meta-heuristic algorithms on 18 UCI standard datasets, such as BDA [56], ALO [57], GSA, SCA, PSO [58], GA [59], WOA, GWO [59], PIL-BOA [28], GWOCSA [37], and WOA-CM [21], BCSA [28]. These results were obtained by [55], [21], [37], [57], [60], and [61], respectively. 3) Moreover, The $HCSWOA_4$-M is compared with the other meta-heuristic algorithms on two DNA-Microarray datasets (eq. colon and leukemia datasets), such as BCS, BBA, BPSO, BDE, BGA, WOA, CBMFSO, EBGWO, and BGWO2. These results were obtained by [21], [61]–[65], respectively. The population size is fixed to 10, whereas the number of maximum iterations is set to 100, and the results are averaged over 20 independent runs to achieve statistically average results.

Table 14 demonstrates the results of BCSA and the proposed algorithms $HCSWOA_4$-V and $HCSWOA_4$-M in terms of fitness values, classification accuracy, selected feature size, and CPU computational time on 18 UCI standard datasets. The best results in the table are highlighted in bold. The experiment results of $HCSWOA_4$-V show that it performs superior to other algorithms in terms of CPU computational time. In terms of classification accuracy, $HCSWOA_4$-M can obtain the highest classification accuracy on 17 datasets except for D14. In addition, the fitness values of the $HCSWOA_4$-M achieve minimum fitness values over 17 datasets, except for D14. Moreover, $HCSWOA_4$-V demonstrates superior performance by having fewer selected feature sizes than others. Overall, the average classification accuracy of the $HCSWOA_4$-M is superior performance, which can prove the competency of the proposed algorithm efficiently to find the optima in the search space.

Table 15. Average Acc, SF and Time by different feature selection algorithms on 18 UCI standard datasets

DB.	Precision			Recall			F-Score		
	BCSA	**Ours-V**	**Ours-M**	**BCSA**	**Ours-V**	**Ours-M**	**BCSA**	**Ours-V**	**Ours-M**
D1	0.971	0.971	**0.975**	0.964	0.965	**0.970**	0.967	0.968	**0.972**
D2	0.941	0.940	**0.947**	0.949	0.947	**0.954**	0.945	0.944	**0.950**
D3	0.961	0.960	**0.964**	0.957	0.953	**0.962**	0.959	0.956	**0.963**
D4	0.752	0.644	**0.886**	0.789	0.807	**0.945**	0.770	0.706	**0.909**
D5	0.502	0.502	**0.502**	0.879	0.879	**0.879**	0.639	0.639	**0.639**
D6	0.807	0.817	**0.847**	0.811	0.825	**0.851**	0.809	0.821	**0.849**
D7	0.842	0.886	**0.904**	0.904	0.908	**0.931**	0.872	0.897	**0.917**
D8	0.967	0.941	**0.973**	0.968	0.941	**0.973**	0.967	0.941	**0.973**
D9	0.658	0.655	**0.714**	0.923	0.914	**0.932**	0.767	0.762	**0.807**
D10	0.898	0.874	**0.995**	0.904	0.879	**0.996**	0.901	0.876	**0.996**
D11	0.880	0.906	**0.907**	0.899	0.926	**0.919**	0.889	0.916	**0.913**
D12	0.847	0.863	**0.872**	0.858	0.871	**0.880**	0.853	0.867	**0.876**
D13	0.742	0.687	**0.745**	0.720	0.759	**0.747**	0.731	0.712	**0.745**
D14	0.785	0.763	**0.791**	0.866	0.787	**0.837**	0.823	0.775	**0.813**
D15	0.950	0.954	**0.961**	0.942	0.944	**0.951**	0.946	0.949	**0.956**
D16	0.819	0.813	**0.830**	0.818	0.812	**0.830**	0.818	0.812	**0.830**
D17	0.950	0.958	**0.964**	0.944	0.952	**0.959**	0.947	0.955	**0.961**
D18	0.881	0.867	**0.895**	0.923	0.916	**0.939**	0.902	0.890	**0.916**
AVG	0.842	0.833	**0.871**	0.890	0.888	**0.914**	0.861	0.855	**0.888**
Rank	2	3	1	2	3	1	2	3	1

Note* **Ours-V:** $HCSWOA_4$-V **Ours-M:** $HCSWOA_4$-M

Table 15 displays the results of BCSA and the proposed algorithms $HCSWOA_4$-V and $HCSWOA_4$-M in terms of metric performances, such as precision, recall, and F-score time, on 18 UCI standard datasets. The experiment results of $HCSWOA_4$-M show that it performs superiorly to other algorithms in all metrics. Therefore, we chose $HCSWOA_4$-M to compare with other meta-heuristic algorithms.

Table 16 represents the results of WOA, ALO, PSO, GWO, GSA, BA, SCA, GA, BDA, PIL-BOA, GWOCSA, WOA-CM, and the proposed algorithm HCSWOA-M in terms of classification accuracy on 18 UCI standard datasets. In the evaluation results, $HCSWOA_4$-M performs superior to other meta-heuristic algorithms in terms of classification accuracy on 7 of 18 datasets, such as D1, D3, D6, D8, D10, D11, and D16. Overall, the average classification accuracy of $HCSWOA_4$-M is the best efficient to find the optima is 0.908.

Table 16. Average classification accuracy by different feature selection algorithm on 18 UCI standard datasets

	WOA	ALO	PSO	GA	GWO	BA	SCA	GSA	BDA	PIL-BOA	GWOCSA	WOA-CM	HCSWOA$_4$-M
D1	0.957	0.961	0.954	0.955	0.960	0.937	0.961	0.957	0.963	0.971	0.972	0.968	**0.975**
D2	0.955	0.930	0.941	0.938	0.938	0.932	0.940	0.942	0.961	0.959	0.962	**0.971**	0.954
D3	0.930	0.929	0.937	0.938	0.933	0.872	0.935	0.951	0.967	0.954	0.963	0.956	**0.965**
D4	0.758	0.660	0.684	0.666	0.725	0.610	0.720	0.706	0.980	0.872	0.990	**1.000**	0.924
D5	0.699	0.745	0.746	0.757	0.693	0.628	0.698	0.777	0.745	**0.759**	0.746	0.742	0.758
D6	0.763	0.826	0.784	0.822	0.777	0.754	0.784	0.777	0.830	0.845	0.833	0.807	**0.851**
D7	0.890	0.866	0.843	0.834	0.898	0.877	0.883	0.881	0.930	0.910	0.915	**0.926**	0.924
D8	0.915	0.956	0.942	0.923	0.914	0.816	0.898	0.908	0.953	0.958	0.955	0.972	**0.973**
D9	0.786	0.787	0.692	0.708	0.763	0.701	0.788	0.781	0.877	0.844	**0.870**	0.852	0.859
D10	0.854	0.864	0.864	0.927	0.827	0.722	0.855	0.835	0.992	0.939	**0.996**	0.991	**0.996**
D11	0.730	0.627	0.720	0.696	0.834	0.795	0.795	0.919	0.895	0.887	0.860	0.792	**0.918**
D12	0.854	0.738	0.740	0.726	0.862	0.844	0.851	0.888	0.915	0.894	**0.906**	0.852	0.875
D13	0.788	0.801	0.769	0.775	0.785	0.800	0.787	0.783	0.853	0.855	0.816	**0.991**	0.833
D14	0.751	0.725	0.728	0.713	0.754	0.665	0.755	0.753	0.788	0.800	0.800	**0.866**	0.835
D15	0.939	0.917	0.894	0.894	0.920	0.851	0.920	0.931	0.958	**0.961**	0.948	0. 939	0.957
D16	0.713	0.773	0.761	0.767	0.710	0.669	0.704	0.695	0.750	0.808	0.729	0.785	**0.830**
D17	0.928	0.911	0.950	0.933	0.948	0.919	0.957	0.951	0.980	**0.983**	0.982	0.959	0.959
D18	0.965	0.909	0.834	0.884	0.953	0.874	0.931	0.939	0.958	0.974	0.969	**0.980**	0.954
AVG	0.843	0.829	0.821	0.825	0.844	0.793	0.842	0.854	0.905	0.899	0.901	0.906	**0.908**
Rank	8	10	12	11	7	13	9	6	3	5	4	2	1

Table 17 outlines the results of the algorithms in terms of the number of selected features on different runs of the algorithms on 18 UCI standard datasets as presented. The proposed algorithm, HCSWOA$_4$-M, demonstrated the highest achieved ability to select significant variables that are smaller than those in other meta-heuristic algorithms. In addition, the average number of selected features of HCSWOA$_4$-M is 13.

Table 17 Average the number of selected features by different feature selection algorithms on 18 UCI standard datasets

		WOA	ALO	PSO	GA	GWO	BA	SCA	GSA	BDA	PIL-BOA	GWOCSA	WOA-CM	HCSWOA$_4$-M
D1	9	5.35	6.28	5.72	5.09	6.90	3.67	6.70	6.07	4.95	5.60	5.00	**4.30**	5.75
D2	30	20.76	16.08	16.56	16.35	19.00	6.23	20.47	6.77	11.85	13.40	13.80	15.81	**5.30**
D3	16	10.35	6.98	6.83	6.62	9.80	12.40	9.00	16.57	4.61	5.40	5.00	6.45	**4.60**
D4	13	10.80	6.62	9.75	10.82	12.07	**5.73**	10.47	8.73	6.10	7.60	6.40	6.05	6.05
D5	13	5.75	10.70	6.18	6.18	7.53	6.07	9.00	5.10	2.70	2.40	4.60	5.25	**1.00**
D6	13	8.65	10.31	7.94	9.49	8.80	5.90	8.47	6.83	6.85	6.80	5.00	6.96	**4.70**
D7	34	21.45	9.42	19.18	17.31	17.33	13.40	19.07	15.40	11.49	9.20	13.00	14.42	**3.65**
D8	36	27.90	24.70	20.81	22.43	31.60	15.00	30.80	19.97	17.75	16.80	18.60	18.54	20.90
D9	18	10.55	11.05	8.98	11.05	11.80	**7.80**	10.87	9.17	8.15	8.60	8.00	8.21	8.50
D10	13	9.80	11.08	9.04	6.83	11.27	6.17	10.67	8.47	6.05	**6.00**	6.40	6.01	6.70
D11	325	144.30	164.13	178.75	177.13	162.80	126.20	182.70	157.20	123.50	142.00	165.80	128.05	**82.65**
D12	60	43.38	37.92	31.20	33.30	41.60	24.70	37.13	30.03	27.48	68.40	29.60	35.64	**25.95**
D13	22	12.10	16.15	12.50	11.75	13.20	7.97	12.60	9.53	7.94	**7.20**	8.00	8.05	9.40
D14	9	6.65	6.99	6.61	6.85	7.53	**4.70**	7.47	5.87	5.95	5.00	5.00	6.90	7.25
D15	16	7.41	9.52	8.80	6.62	8.47	6.13	9.60	8.17	4.14	4.00	4.60	7.41	**2.85**
D16	40	33.20	35.72	22.72	25.28	36.60	16.67	34.40	19.90	20.96	21.20	**18.40**	25.40	24.65
D17	13	8.85	10.70	8.36	8.63	10.73	6.67	9.40	7.37	6.31	**4.60**	6.40	6.80	6.60
D18	16	9.90	13.97	9.74	10.11	12.40	6.57	9.60	8.17	5.70	7.00	**5.20**	6.00	7.45
AVG		22.06	22.68	21.65	21.77	23.86	15.67	24.36	19.41	15.69	18.96	18.27	17.57	13.00
Rank		10	11	8	9	12	2	13	7	3	6	5	4	1

For the test, we used a confidence level of 0.95 for the statistical analysis of the Wilcoxon rank-sum test, and p-values greater than or equal to 0.05 are highlighted in bold by the results of fitness value, as demonstrated in Table 18. The table concluded that the p-value scores obtained by the Wilcoxon rank-sum test had 5% accuracy from a pair of samples for two algorithms of 20 independent runs to test the null hypothesis for UCI standard datasets. As the table shows, p-values reveal that there are significant differences between the results obtained by the HCSWOA$_4$-V, BCSA, and the proposed HCSWOA$_4$-M for all UCI-standard datasets. However, there is no significant difference between HCSWOA$_4$-M and HCSWOA$_4$-V for only the D5 and D11 p-value table results.

Table 18 The performance comparison of proposed algorithm on wilcoxon rank sum test on 18 UCI standard datasets

Algorithms	Datasets								
	D1	D2	D3	D4	D5	D6	D7	D8	D9
HCSWOA$_4$-V	7.11E-03	1.52E-01	1.74E-06	1.78E-03	**NaN**	2.20E-04	1.65E-05	1.06E-07	7.17E-04
BCSA	6.02E-08	7.55E-07	2.82E-04	9.56E-04	3.34E-04	1.66E-06	6.68E-08	1.84E-07	7.15E-04
	D10	D11	D12	D13	D14	D15	D16	D17	D18
HCSWOA$_4$-V	3.03E-05	**1.64E-01**	1.55E-02	4.57E-04	3.27E-05	2.70E-07	6.01E-07	3.67E-02	3.40E-03
BCSA	4.80E-08	3.93E-07	3.07E-06	1.41E-05	9.22E-03	8.21E-07	7.66E-08	6.49E-08	1.16E-04

Table 19 shows the results of BGWO, BGSA, BBA, BPSO, BDE, WOA, ALO, BGA, BGWO2, CBMFSO, EBGWO, BCS, and the proposed algorithm HCSWOA$_4$-M in terms of classification accuracy, the number of selected features, and CPU computational time on 2 DNA-Microarray datasets. The experimentation results of the EBGWO and CBMFSO algorithms obtained the highest accuracy in terms of classification accuracy, namely, EBGWO algorithm is 0.919 and CBMFSO algorithm is 1.0 for the colon and leukemia datasets, respectively. In the case of WOA, it can reduce the number of selected features as well as locate the most relevant optimal feature subsets; namely, the number of selected features is 12 for colon dataset and 506.16 for leukemia dataset, respectively. In addition, the proposed HCSWOA$_4$-M performs superior to other algorithms in terms of CPU computational time.

Interestingly, the average classification accuracy of colon and leukemia datasets of HCSWOA$_4$-M algorithm is the highest achieved among other algorithms, which is 0.939. However, EBGWO and CBMFSO algorithms obtain the highest classification accuracy in colon and leukemia datasets, respectively. On the other hand, both EBGWO and CBMFSO algorithms are not stable to evaluate feature selection problems because their performances are decreasing in different datasets.

According to Table 15 and Table 19, it seems clear that HCSWOA$_4$-M algorithm outperformed other wrapper approaches across 0.908 and 0.939 for 18 UCI standard datasets and 2 DNA-Microarray datasets, respectively. In addition, HCSWOA$_4$-M algorithm achieved the shortest CPU computational time for optimization across two different datasets. Moreover, the result of HCSWOA$_4$-M algorithm is explicit that the proposed algorithm outperforms both CSA and WOA in terms of classification accuracy, the subset size of selected features, and CPU computational time.

Table 19 Average Acc, SF and Time by different feature selection algorithms on 2 DNA-Microarray datasets

Algorithm	Dataset						Summary		
	MD1			MD2					
	Acc	SF	CT	Acc	SF	CT	Acc	SF	CT
BGWO	0.66	1042.1	-	0.8843	3663.77	-	0.772	2352.94	-
BGSA	0.766	995.83	-	0.8435	3555.13	-	0.805	2275.48	-
BBA	0.682	827.5	-	0.8769	827.5	-	0.779	827.5	-
BPSO	0.839	936.5	-	0.814	3514.9	-	0.827	2225.7	-
BDE	0.794	965.3	-	0.784	3531.2	-	0.789	2248.25	-
WOA	0.884	12	8.68	0.963	256.64	43.99	0.924	**134.32**	26.34
ALO	0.866	112	43.45	0.909	506.16	68.99	0.888	309.08	56.22
BGA	0.878	987.3	-	0.792	3481.8	-	0.835	2234.55	-
BGWO2	0.9	455.2	-	0.874	1805.5	-	0.887	1130.35	-
CBMFSO	0.667	992.42	9.52	1	3652.97	24.35	0.834	2322.7	16.94
EBGWO	0.919	143.4	-	0.903	649.8	-	0.911	396.6	-
BCS	0.603	1101.71	3.68	0.95	3944.35	14.16	0.777	2523.03	8.92
HCSWOA$_4$-M	0.911	343.9	**1.39**	0.966	746.4	**1.53**	**0.939**	545.15	**1.46**

6. Discussion

Statistical performance reveals our proposed algorithms can improve the weaknesses of CSA by handling high-dimensional data. This study introduced four proposed algorithms based on CSA and WOA, such as HCSWOA$_1$–HCSWOA$_4$, whose main idea is to employ the strengths across them. As the results show, the two proposed algorithms are superior to others; these are HCSWOA$_2$ and HCSWOA$_4$. In the majority of situations, the performance of the HCSWOA$_4$ is superior to the standard CSA and WOA and is stable and robust as the dimensionality increases, as shown in Tables 2, 4, and 5. The reasons why the HCSWOA$_4$ performs excellently and efficiently in a stable and robust manner are explained next. Begin with using the inertial weight to improve the convergence speed and also control exploration and exploitation capacities in every update position (Eq. (9)). In HCSWOA$_4$, we proposed algorithms to manipulate high-dimensional data, so balancing the exploration and exploitation capacities is needed to increase the efficiency of the exploration phase of the CSA and the exploitation phase of the WOA. In addition, when AP_r is less than 0.8, nearly half of the iterations for an updated position are used to exploit the search space, which aids in performing a local search; when AP_r is greater than 0.8, the remaining iterations are devoted to performing a global search. In the exploration phase, the proposed algorithms use the formula of extended CSA to update the position of search spaces, which may be the cause for the avoidance of stagnation in local minima. According to the HCSWOA$_4$ algorithm, the mechanisms described in the preceding paragraph are the reasons why our proposed algorithm is advantageous for manipulating high-dimensional data by employing benchmark functions. From Section 5.1, it is evident that the HCSWOA$_4$ algorithm performs better than the other algorithms on most of the benchmark functions because of the extended CSA, WOA, and inertia weight, which provide the ability to smoothly balance exploration and exploitation to confront enormous dimension sizes, as shown in Tables 11 and 12. In conclusion, the advantages of the HCSWOA$_4$ include performing much more lightly and conveniently and having only a few parameters to employ in the optimization problems, as that comparison on PSO, GSA, WOA, ABC, GWO, SCA, ALO, ABC, DE, and DA, as shown in Tables 6-9.

The findings in Section 6 prove that the HCSWOA$_4$ is very effective in solving feature selection problems to identify significant feature subsets. The results of this problem showed that the HCSWOA$_4$ with the mutation operator excellent chooses the feature subsets and outperforms the other methods in different metrics, such as classification accuracy, precision, recall, and f-score, as shown in Tables 14–15. In addition, the strategy to select the feature subsets can increase the diversity of the search spaces and jump out to the global optimum to make the algorithm more effective. The main reason that HCSWOA4 can perform well in this type of problem is that $HCSWOA_4$ -M performs other meta-heuristic algorithms that perform well in feature selection problems with different sizes of features, such as 9 up to 7129 features, as expressed in Tables 16–19. As shown in Tables 10 and 18, all statistical results support our proposed algorithm's claim that there are significant differences for both CSA and WOA on the Wilcoxon rank-sum test. As a result of combining these two algorithms, $HCSWOA_4$ outperforms so well in feature selection problems. The comprehensive study conducted here reveals that the $HCSWOA_4$ has a stronger ability to encounter a global optimum, is more stable and robust than other meta-heuristic algorithms, and solves real-world engineering problems efficiently.

To conclude, the limitation of this work is that it focuses on the strategy for choosing significant feature subsets for feature selection problems. In this study, we chose two strategies for selecting features: the v-shaped transfer function and the mutation operator. The proposed algorithm with the mutation operator is superior to other meta-heuristic algorithms. According to the limitation, it can be used to improve and enrich various search spaces, as well as help the algorithm bounce out of the local optimum for finding the significant feature subset more effectively and efficiently on classification tasks. Furthermore, the fitness function is a critical weight for indicating the appropriateness of selecting feature subsets, which are scored by balancing the number of selected features and classification accuracy.

7. Conclusions and Future Direction

This paper proposed a novel hybrid algorithm, HCSWOA, which is an improved crow search algorithm with whale optimization algorithm to solve both a high-dimensional optimization problem and a real-world optimization problem, such as feature selection problems. In HCSWOA, the inertia weight parameter plays a critical role in balancing exploration and exploitation. The performance of the proposed algorithm is tested on twenty-three standard benchmark functions, such as unimodal and multimodal functions. According to the evaluation results, this can significantly improve the HCSWOA's performance in terms of balancing exploitation and exploration and increasing the convergence speed for supporting high-dimensional optimization problems. Furthermore, the best HCSWOA variant with a mutation operator is called HCSWOA-M, which is employed as a feature selection approach and its performance is validated on eighteen UCI standards and two DNA-microarray datasets. The results of HCSWOA-M as a feature selection approach are compared against well-known feature selection methods, such as ALO, GA, PSO, CSA, WOA, GWO, DE, DA, BA, and PSO. The experiment results demonstrated that the comprehensive performance of HCSWOA is very competitive in terms of the optimal solution and classification efficiency. For future studies, HCSWOA can be utilized as a filter feature selection approach to evaluate the generality of the selected features. Moreover, HCSWOA can be applied to more efficient problems in real-life world applications, such as vehicle scheduling problems, knapsack problems, image thresholding problems, etc.

Funding
No funding is provided for the preparation of manuscript.
Conflict Of Interest Statement
The authors have no conflicts of interest to declare.

References

[1] Sharma, M.; Kaur, P. (2020). A comprehensive analysis of nature-inspired meta-heuristic techniques for feature selection problem. Archives of Computational Methods in Engineering, pp.1–25.
[2] Purkayastha, R.; Chakraborty, T.; Saha, A.; Mukhopadhyay, D. (2020). Study and Analysis of Various Heuristic Algorithms for Solving Travelling Salesman Problem—A Survey. Proceedings of the Global AI Congress 2019, pp.61–70.
[3] Chouksey, M.; Jha, R. K.; Sharma, R. (2020). A fast technique for image segmentation based on two Meta-heuristic algorithms. Multimedia Tools and Applications, pp.1–53.
[4] Sharma, N.; Gupta, V. (2020). Meta-heuristic based optimization of WSNs Localisation Problem-a Survey. Procedia Computer Science, 173, pp.36–45.
[5] Wang, L.; Wang, Y.; Chang, Q. (2016). Feature selection methods for big data bioinformatics: A survey from the search perspective. Methods, 111, pp.21–31.
[6] Mohamad, M.; Hassan, H.; Nasien, D.; Haron, H. (2015). A review on feature extraction and feature selection for handwritten character recognition. International Journal of Advanced Computer Science and Applications, 6(2).
[7] Alyasiri, O. M.; Cheah, Y.-N.; Abasi, A. K.; Al-Janabi, O. M. (2022). Wrapper and Hybrid Feature Selection Methods Using Metaheuristic Algorithms for English Text Classification: A Systematic Review. IEEE Access.
[8] Liu, Y.; Ghandar, A.; Theodoropoulos, G. (2019). A metaheuristic strategy for feature selection problems: Application to credit risk evaluation in emerging markets. 2019 IEEE Conference on Computational Intelligence for Financial Engineering Economics (CIFEr), pp.1–7.
[9] Vieira, S. M.; Mendonça, L. F.; Farinha, G. J.; Sousa, J. M. (2013). Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients. Applied Soft Computing, 13(8), pp.3494–3504.
[10] Ouadfel, S.; Abd Elaziz, M. (2022). Efficient high-dimension feature selection based on enhanced equilibrium optimizer. Expert Systems with Applications, 187, pp.115882.

[11] Kiliç, Ü.; Keleş, M. K. (2018). Feature selection with artificial bee colony algorithm on Z-Alizadeh Sani dataset. 2018 Innovations in Intelligent Systems and Applications Conference (Asyu), pp.1–3.
[12] Manoj, R. J.; Praveena, M. A.; Vijayakumar, K. (2019). An ACO–ANN based feature selection algorithm for big data. Cluster Computing, 22(2), pp.3953–3960.
[13] Kundu, R.; Chattopadhyay, S.; Cuevas, E.; Sarkar, R. (2022). AltWOA: Altruistic Whale Optimization Algorithm for feature selection on microarray datasets. Computers in Biology and Medicine, 144, pp.105349.
[14] Zivkovic, M.; Jovanovic, L.; Ivanovic, M.; Krdzic, A.; Bacanin, N.; Strumberger, I. (2022). Feature selection using modified sine cosine algorithm with COVID-19 dataset. In Evolutionary Computing and Mobile Sustainable Networks, Springer, pp. 15–31.
[15] Ewees, A. A.; Mostafa, R. R.; Ghoniem, R. M.; Gaheen, M. A. (2022). Improved seagull optimization algorithm using Lévy flight and mutation operator for feature selection. Neural Computing and Applications, 34(10), pp.7437–7472.
[16] Liu, W.; Wang, J. (2019). A brief survey on nature-inspired metaheuristics for feature selection in classification in this decade. 2019 IEEE 16th International Conference on Networking, Sensing and Control (ICNSC), pp.424–429.
[17] Hancer, E. (2020). New filter approaches for feature selection using differential evolution and fuzzy rough set theory. Neural Computing and Applications, pp.1–16.
[18] Manbari, Z.; AkhlaghianTab, F.; Salavati, C. (2019). Hybrid fast unsupervised feature selection for high-dimensional data. Expert Systems with Applications, 124, pp.97–118.
[19] Pereira, L. A. M.; Rodrigues, D.; Almeida, T. N. S.; Ramos, C. C. O.; Souza, A. N.; Yang, X.-S.; Papa, J. P. (2014). A binary cuckoo search and its application for feature selection. In Cuckoo Search and Firefly Algorithm, Springer, pp.141–154.
[20] Chuang, L.-Y.; Chang, H.-W.; Tu, C.-J.; Yang, C.-H. (2008a). Improved binary PSO for feature selection using gene expression data. Computational Biology and Chemistry, 32(1), pp.29–38.
[21] Mafarja, M.; Mirjalili, S. (2018). Whale optimization approaches for wrapper feature selection. Applied Soft Computing, 62, pp.441–453.
[22] Ramasamy, R.; Rani, S. (2018). Modified binary bat algorithm for feature selection in unsupervised learning. Int. Arab J. Inf. Technol.; 15(6), pp.1060–1067.
[23] Nouri-Moghaddam, B.; Ghazanfari, M.; Fathian, M. (2020). A novel filter-wrapper hybrid gene selection approach for microarray data based on multi-objective forest optimization algorithm. Decision Science Letters, 9(3), pp.271–290.
[24] Chinnaswamy, A.; Srinivasan, R. (2016). Hybrid feature selection using correlation coefficient and particle swarm optimization on microarray gene expression data. In Innovations in bio-inspired computing and applications, Springer, pp.229–239.
[25] Loey, M.; Jasim, M. W.; El-Bakry, H. M.; Taha, M. H. N.; Khalifa, N. E. M. (2020). Breast and Colon Cancer Classification from Gene Expression Profiles Using Data Mining Techniques. Symmetry, 12(3), pp.408.
[26] Sun, L.; Kong, X.; Xu, J.; Zhai, R.; Zhang, S. (2019). A hybrid gene selection method based on ReliefF and ant colony optimization algorithm for tumor classification. Scientific Reports, 9(1), pp.1–14.
[27] Bolón-Canedo, V.; Sánchez-Marono, N.; Alonso-Betanzos, A.; Benítez, J. M.; Herrera, F. (2014). A review of microarray datasets and applied feature selection methods. Information Sciences, 282, pp.111–135.
[28] De Souza, R. C. T.; dos Santos Coelho, L.; De Macedo, C. A.; Pierezan, J. (2018). A V-shaped binary crow search algorithm for feature selection. 2018 IEEE Congress on Evolutionary Computation (CEC), pp.1–8.
[29] Al-Thanoon, N. A.; Algamal, Z. Y.; Qasim, O. S. (2021). Feature selection based on a crow search algorithm for big data classification. Chemometrics and Intelligent Laboratory Systems, pp.104288.
[30] Ouadfel, S.; Abd Elaziz, M. (2020). Enhanced crow search algorithm for feature selection. Expert Systems with Applications, 159, pp.113572.
[31] Allahverdipour, A.; Soleimanian Gharehchopogh, F. (2018). An improved k-nearest neighbor with crow search algorithm for feature selection in text documents classification. Journal of Advances in Computer Research, 9(2), pp.37–48.
[32] Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. Computers Structures, 169, pp.1–12.
[33] Mohammadi, F.; Abdi, H. (2018). A modified crow search algorithm (MCSA) for solving economic load dispatch problem. Applied Soft Computing, 71, pp.51–65.
[34] Shi, Z.; Li, Q.; Zhang, S.; Huang, X. (2017). Improved crow search algorithm with inertia weight factor and roulette wheel selection scheme. 2017 10th International Symposium on Computational Intelligence and Design (ISCID), 1, pp.205–209.
[35] Pasandideh, S. H. R.; Khalilpourazari, S. (2018). Sine cosine crow search algorithm: A powerful hybrid meta heuristic for global optimization. ArXiv Preprint ArXiv:1801.08485.
[36] Khalilpourazari, S.; Pasandideh, S. H. R. (2019). Sine–cosine crow search algorithm: Theory and applications. Neural Computing and Applications, pp.1–18.
[37] Arora, S.; Singh, H.; Sharma, M.; Sharma, S.; Anand, P. (2019a). A new hybrid algorithm based on Grey wolf optimization and crow search algorithm for unconstrained function optimization and feature selection. IEEE Access, 7, pp.26343–26361.
[38] Qu, C.; Fu, Y. (2019). Crow search algorithm based on neighborhood search of non-inferior solution set. IEEE Access, 7, pp.52871–52895.
[39] Jain, M.; Rani, A.; Singh, V. (2017). An improved Crow Search Algorithm for high-dimensional problems. Journal of Intelligent Fuzzy Systems, 33(6), pp.3597–3614. https://doi.org/10.3233/JIFS-17275
[40] Farnad, B.; Jafarian, A.; Baleanu, D. (2018). A new hybrid algorithm for continuous optimization problem. Applied Mathematical Modelling, 55, pp.652–673. https://doi.org/10.1016/j.apm.2017.10.001
[41] Talbi, E.-G. (2002). A taxonomy of hybrid metaheuristics. Journal of Heuristics, 8(5), pp.541–564.
[42] Mirjalili, S.; Lewis, A. (2016). The whale optimization algorithm. Advances in Engineering Software, 95, pp.51–67.
[43] Singh, N.; Hachimi, H. (2018). A new hybrid whale optimizer algorithm with mean strategy of grey wolf optimizer for global optimization. Mathematical and Computational Applications, 23(1), pp.14.
[44] Emary, E.; Zawbaa, H. M.; Salam, M. A. (2017). A proposed whale search algorithm with adaptive random walk. 2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), pp.171–177.
[45] Luo, J.; Shi, B. (2019). A hybrid whale optimization algorithm based on modified differential evolution for global optimization problems. Applied Intelligence, 49(5), pp.1982–2000.
[46] Chakraborty, S.; Saha, A. K.; Sharma, S.; Mirjalili, S.; Chakraborty, R. (2021). A novel enhanced whale optimization algorithm for global optimization. Computers Industrial Engineering, 153, pp.107086.
[47] Ling, Y.; Zhou, Y.; Luo, Q. (2017). Lévy flight trajectory-based whale optimization algorithm for global optimization. IEEE Access, 5, pp.6168–6186.
[48] Wolpert, D. H.; Macready, W. G. (1997). No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1(1), pp.67–82.
[49] Mirjalili, S.; Mirjalili, S. M.; Lewis, A. (2014). Grey wolf optimizer. Advances in Engineering Software, 69, pp.46–61.
[50] Mirjalili, S. (2016). Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Computing and Applications, 27(4), pp.1053–1073.

[51] Xu, Y.; Chen, H.; Luo, J.; Zhang, Q.; Jiao, S.; Zhang, X. (2019). Enhanced Moth-flame optimizer with mutation strategy for global optimization. Information Sciences, 492, pp.181–203.
[52] Long, W.; Jiao, J.; Liang, X.; Wu, T.; Xu, M.; Cai, S. (2021). Pinhole-imaging-based learning butterfly optimization algorithm for global optimization and feature selection. Applied Soft Computing, 103, pp.107146.
[53] Abudayor, A.; Nalbantoğlu, Ö. U. (2022). An Improved Crow Search Algorithm with Grey Wolf Optimizer for High-Dimensional Optimization Problems. International Conference on Soft Computing and Its Engineering Applications, pp.51–64.
[54] Blake, C. (1998). UCI repository of machine learning databases. Http://Www. Ics. Uci. Edu/~ Mlearn/MLRepository.
[55] Friedman, J.; Hastie, T.; Tibshirani, R. (2001). The elements of statistical learning. Springer series in statistics New York, 1(10).
[56] Too, J.; Abdullah, A. R.; Mohd Saad, N.; Mohd Ali, N.; Tee, W. (2018). A new competitive binary grey wolf optimizer to solve the feature selection problem in EMG signals classification. Computers, 7(4), pp.58.
[57] Mafarja, M. M.; Eleyan, D.; Jaber, I.; Hammouri, A.; Mirjalili, S. (2017). Binary dragonfly algorithm for feature selection. 2017 International Conference on New Trends in Computing Sciences (ICTCS), pp.12–17.
[58] Zawbaa, H. M.; Emary, E.; Parv, B. (2015). Feature selection based on antlion optimization algorithm. 2015 Third World Conference on Complex Systems (WCCS), pp.1–7.
[59] Kennedy, J.; Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, 5, pp.4104–4108.
[60] Eiben, A. E.; Raue, P.-E.; Ruttkay, Z. (1994). Genetic algorithms with multi-parent recombination. International Conference on Parallel Problem Solving from Nature, pp.78–87.
[61] Arora, S.; Anand, P. (2019). Binary butterfly optimization approaches for feature selection. Expert Systems with Applications, 116, pp.147–160.
[62] Faris, H.; Mafarja, M. M.; Heidari, A. A.; Aljarah, I.; Ala'M, A.-Z.; Mirjalili, S.; Fujita, H. (2018). An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. Knowledge-Based Systems, 154, pp.43–67.
[63] Khurma, R. A.; Aljarah, I.; Sharieh, A. (2020). An Efficient Moth Flame Optimization Algorithm using Chaotic Maps for Feature Selection in the Medical Applications. ICPRAM, pp.175–182.
[64] Segera, D.; Mbuthia, M.; Nyete, A. (2020). An Excited Binary Grey Wolf Optimizer for Feature Selection in Highly Dimensional Datasets: Proceedings of the 17th International Conference on Informatics in Control, Automation and Robotics, pp.125–133. https://doi.org/10.5220/0009805101250133
[65] Yan, C.; Ma, J.; Luo, H.; Wang, J. (2018). A hybrid algorithm based on binary chemical reaction optimization and tabu search for feature selection of high-dimensional biomedical data. Tsinghua Science and Technology, 23(6), pp.733–743.

Authors Profile

A. Abudayor received a BE degree in computer engineering from Naresuan University, Phisanulok, Thailand, in 2011 and an ME degree in computer engineering from Prince of Songkhla University, Songkhla, Thailand, in 2016. She is currently pursuing a Ph.D. degree in computer engineering at Erciyes University, Kayseri, Turkey. She has a total of five years of teaching experience. Her interests are optimization, heuristic algorithms, classification, data mining, soft computing, and machine learning.

Dr. Ö. U. Nalbantoğlu received his B.S. in Electrical and Electronics Engineering from Bogazici University, Istanbul, Turkey. He was awarded Ph.D. Electrical Engineering from University of Nebraska at Lincoln, Lincoln, Nebraska, United States. Currently, he is working as an Associate Professor in the Department of Computer Engineering at Erciyes University, Kayseri, Turkey. His research interests include machine learning, Artificial intelligence, bioinformatics, information theory, computational metagenomics, and meta-omics.