electronics

MDPI

Article

A Robust Adaptive Hierarchical Learning Crow Search Algorithm for Feature Selection

Yilin Chen [1,2], Zhi Ye [1], Bo Gao [1], Yiqi Wu [3], Xiaohu Yan [4] and Xiangyun Liao [5,*]

[1] School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan 430073, China; yilinchen@wit.edu.cn (Y.C.); zhiye@stu.wit.edu.cn (Z.Y.); bogao@stu.wit.edu.cn (B.G.)
[2] Hubei Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan 430073, China
[3] School of Computer Science, China University of Geosciences, Wuhan 100083, China; wuyq@cug.edu.cn
[4] School of Artificial Intelligence, Shenzhen Polytechnic, Shenzhen 518055, China; yanxiaohu@szpt.edu.cn
[5] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China
* Correspondence: xy.liao@siat.ac.cn

Abstract: Feature selection is a multi-objective problem, which can eliminate irrelevant and redundant features and improve the accuracy of classification at the same time. Feature selection is a great challenge to balance the conflict between the two goals of selection accuracy and feature selection ratio. The evolutionary algorithm has been proved to be suitable for feature selection. Recently, a new meta-heuristic algorithm named the crow search algorithm has been applied to the problem of feature selection. This algorithm has the advantages of few parameters and achieved good results. However, due to the lack of diversity in late iterations, the algorithm falls into local optimal problems. To solve this problem, we propose the adaptive hierarchical learning crow search algorithm (AHL-CSA). Firstly, an adaptive hierarchical learning technique was used to adaptive divide the crow population into several layers, with each layer learning from the top layer particles and the topmost layer particles learning from each other. This strategy encourages more exploration by lower individuals and more exploitation by higher individuals, thus improving the diversity of the population. In addition, in order to make full use of the search information of each level in the population and reduce the impact of local optimization on the overall search performance of the algorithm, we introduce an information sharing mechanism to help adjust the search direction of the population and improve the convergence accuracy of the algorithm. Finally, different difference operators are used to update the positions of particles at different levels. The diversity of the population is further improved by using different difference operators. The performance of the method was tested on 18 standard UCI datasets and compared with eight other representative algorithms. The comparison of experimental results shows that the proposed algorithm is superior to other competitive algorithms. Furthermore, the Wilcoxon rank-sum test was used to verify the validity of the results.

Keywords: feature selection; crow search algorithm; hierarchical learning; information sharing; multi-strategy

Citation: Chen, Y.; Ye, Z.; Gao, B.; Wu, Y.; Yan, X.; Liao, X. A Robust Adaptive Hierarchical Learning Crow Search Algorithm for Feature Selection. *Electronics* **2023**, *12*, 3123. https://doi.org/10.3390/electronics12143123

Academic Editor: Gemma Piella

Received: 7 June 2023
Revised: 14 July 2023
Accepted: 17 July 2023
Published: 18 July 2023

1. Introduction

Feature selection is a technique used in machine learning and data analysis [1] to select the most relevant and important features from a dataset. It can improve the performance of a machine learning model by reducing dimensionality, removing irrelevant or redundant features, and focusing on the most informative ones. Feature selection methods comprise three categories: the filter method [2], embedded method [3] and wrapper method [4]. The filter method enjoys low computational cost and can avoid overfitting. However, it suffers from neglecting future dependencies, limited adaptability and insensitivity to classification boundaries. The embedded method automatically selects the optimal subset of features for a model, saving time and effort. By integrating feature selection into the model training process, it effectively reduces the risk of overfitting, but it increases the computational

intensity during model training. The wrapper approach captures feature interactions and non-linear relationships, tailoring feature selection for specific machine learning algorithms. It is effective with high-dimensional data. Therefore, we adopt the wrapper approach in our proposed algorithm.

A major challenge in feature selection is balancing the trade-off between accuracy and rate. The results demonstrate that using a meta-heuristic algorithm effectively balances this conflict by leveraging meta-heuristic information to narrow down the search space. In recent years, a heuristic search algorithm known as the crow search algorithm, inspired by crows, has been introduced. The CSA is a population-based strategy inspired by the behavior of crows, which involves storing excess food in hidden places for later retrieval. The algorithm uses awareness probability to detect if the crow is being followed and flight length to update the crow's position iteratively. Prior studies have demonstrated the effectiveness of the CSA algorithm in tackling complex optimization problems involving feature selection.

In group-based algorithms, achieving a balance between exploration and exploitation is a challenging task. In the early stage of algorithm iteration, global exploration becomes particularly crucial. In the late stage of algorithm iteration, the focus shifts to local development, where a refined search is needed to find the optimal solution.

Parameter setting in optimization algorithms is time-consuming and considered a disadvantage. Algorithms with fewer tuning parameters are easier to implement. CSA is a population-based optimization algorithm with adjustable parameters: flight length and awareness probability. Adjusting only these two parameters makes implementation easier. However, the simplistic setting of flight distance parameters limits the algorithm's exploration and exploitation. The author sets the awareness probability to 0.1, indicating the consideration of local search ability but lack of optimal individual guidance, resulting in slightly insufficient performance. However, mere parameter changes are insufficient for the algorithm to achieve improved results. Therefore, the algorithm undergoes the following improvements:

1. To enhance population diversity, we employed the adaptive hierarchical strategy, utilizing the Hamming stratification method to determine the level number.
2. Multi-strategy variance is a reinforcement method based on differential evolutionary algorithms. It generates new individuals using randomly selected strategies and incorporates the best individuals into the subsequent population. This technique helps populations avoid local optima and discover superior alternatives.
3. The information-sharing mechanism selects parts of top individuals at each level and involves them in particle updates, accelerating convergence rate.

2. Related Works

Feature selection [5] is a challenging NP-hard combinatorial problem, which can be effectively addressed using meta-heuristic algorithms. These algorithms use defined rules and randomness to simulate natural phenomena, thus avoiding the gradient problem with the optimization process [6], which is different from traditional mathematical programming methods [7]. So, some evolutionary algorithms [8] are generally applied to feature selection.

For example, the binary particle swarm optimization algorithm (BPSO) [9] simulates the behavior of birds in a flock, seeking optimal solutions through cooperation and information sharing between individuals in the flock. Mafarja et al. proposed a dragonfly-based optimization algorithm that uses binary to solve the feature selection problem (BDA) [10]. The genetic algorithm [11–13] (GA) is a computational model simulating the biological evolution process of Darwinian biology. It adopts selection, crossover, mutation and other operations to make the algorithm avoid local mimina and increase the diversity. The ant colony algorithm [14] (ACO) is a probabilistic algorithm used to find the optimal path, which is inspired by the behavior of ants looking for the path in the process of finding food. The search process adopts the distributed computing method, and multiple individuals carry out parallel calculations at the same time. The computational power and operation

efficiency of the algorithm are greatly improved. The artificial bee colony Algorithm [15,16] (ABC) is an optimization algorithm to simulate the behavior of bees. It is composed of a simple individual and represents extremely complex behavior. Its advantages are strong global optimization ability and fast convergence speed. The gray wolf-based optimization algorithm [17] (GWO) is inspired by the predatory behavior of the gray wolf population. It has the advantages of relatively high diversity, fast convergence and wide application. Okwu et al. proposed a locust optimization algorithm to solve some problems, which mainly uses the social forces among locusts to divide the space into attractive space, comfortable space and repellent space, and it changes the space scope according to the changes in the distance between different locusts to find the optimal location [18]. The whale optimization algorithm (WOA) [19] is a new group intelligent optimization algorithm simulating whale hunting behavior in nature, but the algorithm has the shortcomings of slow convergence, low accuracy, and easily falling into the local optimal. The crow search algorithm [20] (CSA) mimics the behavior of crows searching for food, but because of the greedy newer method, the algorithm is prone to falling into local optimization.

To address issues with local optimization and lack of diversity in meta-heuristic algorithms, researchers have proposed optimizing these algorithms by modifying their search mechanisms. Adaptive parameters such as non-dominant sorting, positive and corotational functions, transformation function optimization, multi-strategy mechanism, etc. can help achieve the purpose of algorithm optimization.

A particle swarm optimization algorithm with adaptive inertial weights is also used [21,22]. With the progress of iteration, by changing the weights in the particle swarm, the global search and local search capabilities of the algorithm can be better balanced. Mafarja and others have also optimized the dragonfly algorithm. Changing the fly length of the dragonfly so that the flight length varies with iteration is a strategy [23]. In genetic algorithms, non-dominated sorting [24] is used to increase the diversity of the population in the selection process and can well avoid local optimization problems. To solve the problem whereby the crow search algorithm easily fall into local optimal, a binary crow search algorithm based on time flight was proposed [25]. A strategy that can vary the flight distance of each crow as the iteration progresses can be used to optimize problems that fall into local optimality. Another is the chaos crow search algorithm [26,27]. After the use of chaos, instead of a random function, using a positive rotation function to determine the size of the data also plays an optimization role. De Souza et al. proposed a V-shaped transfer function to optimize the crow search algorithm. A good transfer function can better reflect the advantages of the algorithm [28]. Aiming at the problems of the slow convergence and easy convergence of the gray wolf optimization algorithm, a dynamic gray wolf optimizer is proposed. After completing its own search, the optimizer can directly compare with the first wolf without waiting for other wolves, thus improving the iterative convergence speed. Hichem et al. proposed a new way of thinking to solve the feature selection problem of the binary grasshopper optimization algorithm and also improve the performance of the algorithm [29,30]. The whale optimization algorithm has the advantages of simple operation, few adjustment parameters and strong ability to jump out of local optimization. However, some people have proposed optimization ideas for this algorithm, proposing the idea of non-dominant sorting and adaptive parameters [31], which also applies to the whale optimization algorithm. Wang et al. proposed an adaptive hierarchical learning algorithm [32,33], which can also jump out of the local optimal. Xiaoyu et al. proposed a multi-strategy differential evolution algorithm [34–36], which also greatly improves the algorithm. These methods can improve the optimization performance of algorithms. The above are some optimization ideas for individual algorithms as well as adaptive weights [37]. However, millimeter-wave communication remains challenging. Jamil et al. analyzed 5G networks in terms of throughput and energy efficiency, and they proposed OAA [38] to solve the formulaic problem. Nyiam et al. proposed an extension of the multi-objective simplex algorithm (MSA) to generate a set of all non-dominated and

non-redundant points [39]. Wambua et al. proposed a bat algorithm for prioritizing test cases in regression testing [40].

3. Proposed Method

3.1. Crow Search Algorithm

The crow search algorithm is a meta-heuristic algorithm, known by its name, based on the behavior of the crow population. The crow search algorithm is a group-based technology. Its working principle is that crows store surplus food in a hidden place and take it out when they need it. Crows are very greedy. They will track each other and steal each other's food. However, tracking actions is not an easy task. In fact, crows will stalk another crow to find a new source of food and steal it from there. On the other hand, if the crow realized that another crow was following it, it did not go to the spot where it had hidden its food but instead went to a random spot and tricked the following crow.

The implementation principles of the crow search algorithm follow:

- Crows live in groups.
- Crows will remember where they hide their food.
- Crows will follow other crows to steal food.
- Crows protect their own food from theft.

First, the population size N and dimension d should be initialized, and two parameters AP and FL should be defined. After calculating the position of each crow, initialize the position in the crow's memory. In the Algorithm 1, N represents the size of the population, d represents the dimension of the population, and the position of the crow is represented by the vector X = $(x_1, x_2, x_3, \ldots, x_d)$.

Algorithm 1 Crow search algorithm.

Input: pop
Output: popmem

Initialize the flock of N crows randomly in d dimensional search space;
Define awareness probability AP;
Evaluate the position of each crow;
Initialize the memory of each crow;
while $iteration < max\_iteration$ **do**
  for i = 1 to N **do**
    Randomly select one crow j to follow i
    if $r_j \geq AP$ **then**
      $x_{new} = x_{current} + r_i \times fl \times (mem_{j,current} - x_{i,current})$
    else
      x_{new} = any random position in search space
    end if
  end for
  Check the feasibility of new position;
  Evaluate the new position of the crow;
  Update the memory of the crow;
end while

To describe the specific steps of the crow search algorithm process more visually, we drew a flowchart of the crow search algorithm. The graphical representation of the algorithm makes it more intuitive and easier to understand. Figure 1 contains the diagram chart of the crow search algorithm:

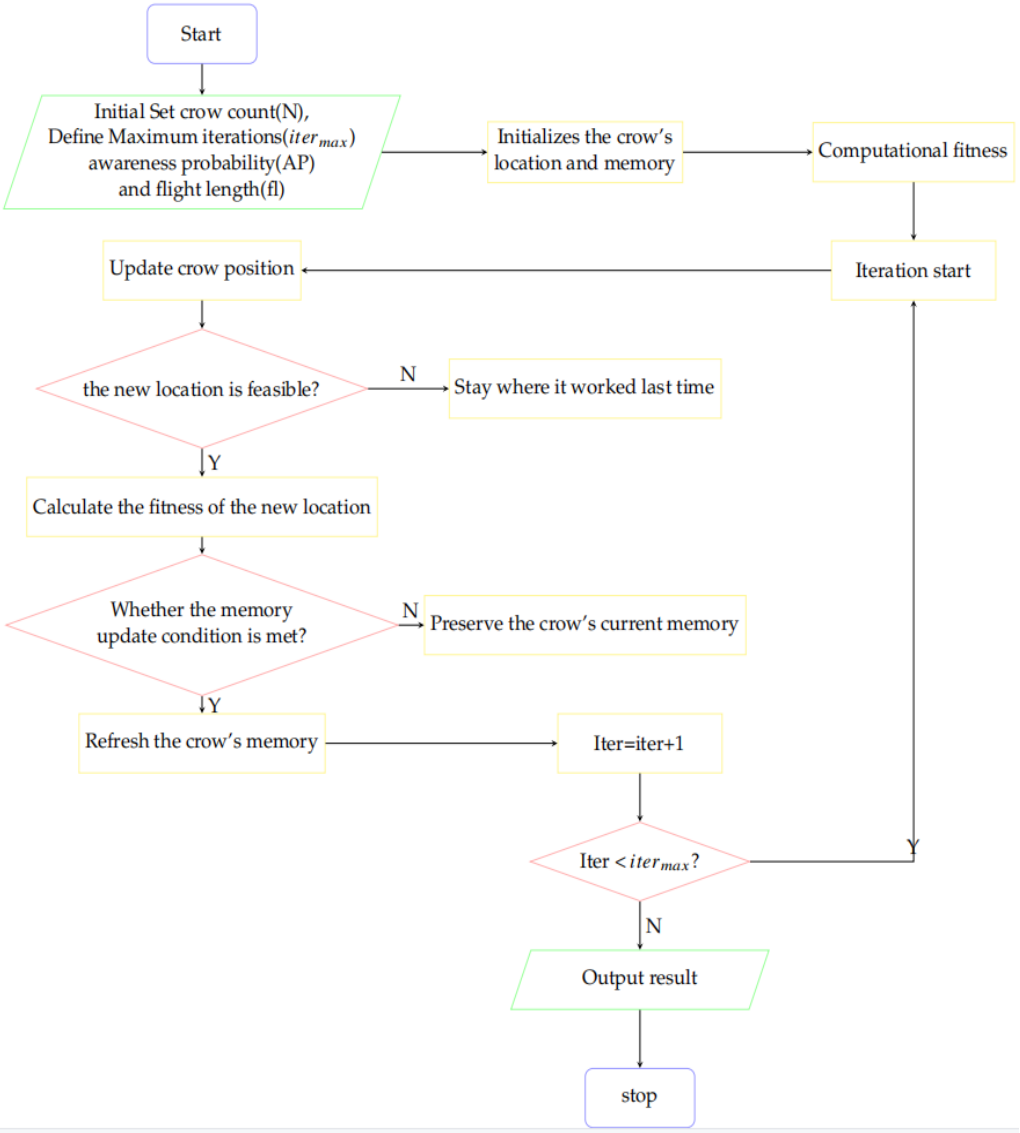


Figure 1. Flow diagram of the crow search algorithm.

3.2. Binary Crow Search Algorithm

The solution space of the simple crow search algorithm is continuous, but the feature selection in the input machine learning model requires a discrete feature subset. The crow search algorithm cannot directly solve the binary feature selection problem. To apply the crow search algorithm to the feature selection problem, we need to design a binary coding version. Since the feature selection solution will be distributed between binary numbers 0 and 1, there are two types of transform functions: S-type and V-type. Next, we will introduce these two conversion functions.

The formula of the one S-type transfer function is:

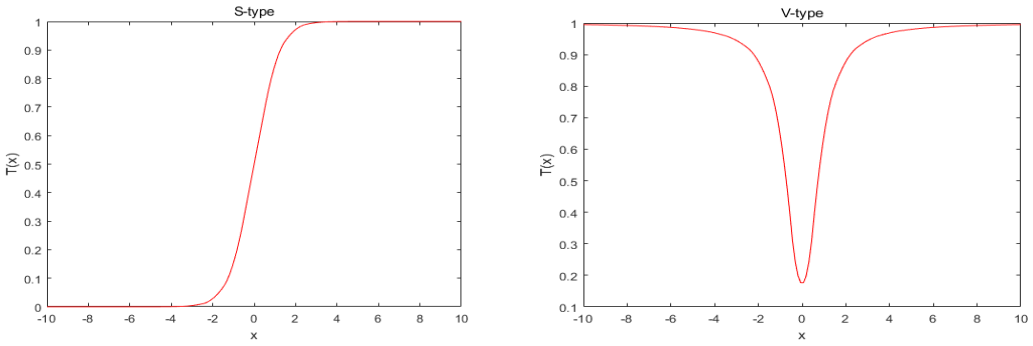
$$T(x) = \frac{1}{1 + e^{-2x}} \tag{1}$$

The formula of the one V-type transfer function is:

$$T(x) = \left| \frac{x}{\sqrt{1 + x^2}} \right| \tag{2}$$

There are two types of transfer functions with varying slopes, resulting in different exploration outcomes. A steep transfer function curve leads to poor exploration, while a

less steep or flat curve leads to poor exploitation and a higher likelihood of falling into a local minimum [41]. Please refer to the diagram (Figure 2) below, illustrating the two different transfer functions.



(**a**) s-type (**b**) v-type

Figure 2. Trend graph of s-type and v-type transform functions.

This paper aims to apply the crow search algorithm for the feature selection of discrete classes. Therefore, selecting the appropriate binary transfer functions is crucial. The two formulas presented here are employed in the feature selection process using the binary crow search algorithm, and the results obtained with these two transfer functions are compared.

3.3. Adaptive Hierarchical Learning Crow Search Algorithm

The proposed method AHL-CSA is described here. This paper introduces the operation of the system from three aspects: adaptive hierarchical operation, information sharing mechanism and differential operator with multi-policy integration.

The following is a detailed introduction to each step of the algorithm, first to determine the dimension of the dataset. N is the number of initialized crows, and d is the data dimension. The fitness of each crow was then initialized. The initialization is complete. Enter the loop, then check the feasibility of the new location, evaluate the crow's new location, and update the crow's memory. Algorithm 2 is the pseudo-code for the algorithm.

The average Hamming value of the population is calculated according to Formulas (3) and (4), and then the number of layers is calculated according to the average Hamming value of the population, using Formulas (5) and (6). Equations (7)–(10) can be used for migration learning in the formula. Particles of different levels will be used according to different difference formulas.

Feature selection is a multi-objective problem. After the feature vectors are obtained by the evolutionary algorithm, these feature vectors are input into the machine learning model, and their performance is evaluated to obtain the classification accuracy and feature ratio. Feature selection is to select important features to improve the classification accuracy and select the least number of features. These two goals are in conflict with each other, so we use the following formula to obtain the fitness value:

$$Fitness(x) = \delta \times \gamma_R(x) + \beta \times \frac{|x|}{|n|} \tag{3}$$

In the formula, $Fitness(x)$ is the fitness value of the x vector, $\gamma_R(x)$ is the classification error rate obtained from the feature vector x, $\frac{|x|}{|n|}$ is the feature selection ratio of x, and δ and β represents fixed values that satisfy $\delta + \beta = 1$, $\beta = 0.01$.

Algorithm 2 Hierarchical learning crow search algorithm.

Determine the number of feature in the dataset, call it d;
Initialize the flock of N crows randomly in d-dimensional binary vectors;
Evaluate the position of each crow;
Initialize the memory of each crow;
while $iteration < max\_iteration$ **do**
    Calculate the average Hamming value of the population;
    Calculate the number of layering;
    Sort the population according to fitness and find out the position of the best value and the position of the worst value;
    Calculate the average position of the population at this time;
    Through the difference formula, the different offsets required for different particles are $learn_{i,current}$;
    for i = 1 to N **do**
        Randomly select one crow j to follow i
        Define awareness probability AP
        if $r_j \geq AP$ **then**
            $x_{new} = x_{current} + r_i \times fl \times (mem_{j,current} - x_{i,current}) + learn_{i,current}$
        else
            x_{new} = any random position in search space
        end if
    end for
    Check the feasibility of the new position;
    Evaluate the new position of the crow;
    Update the memory of the crow;
end while

3.3.1. Adaptive Hierarchical Operation

The BCSA-TVFL proposed by Abhilasha et al. changes the flight length to an adaptive value, which means that the flight length (FL) will gradually decrease linearly with the iterations of the algorithm. Although the flight length will change with iterations and is consistent with the habitat of crows in nature, this simple optimization parameter is still flawed.

Therefore, we use a hierarchical strategy to optimize this problem. Since the population has different levels, the particles in the population can maintain a good diversity. High-order particles focus on the exploitation of the solution space, and low-order particles focus on the exploration of the solution space, which is balanced by the self-organization of the population. The hierarchy is implemented as follows:

1. Arrange population individuals in ascending order based on their fitness values after calculating the fitness for each particle;
2. The ranked population is divided into NL layers, with higher ranked particles assigned to higher levels and lower ranked particles assigned to lower levels. The number of particles in each stratum is the same, except when the number of population cannot be evenly divided by the number of strata. In this case, the last NP%NL particles are placed in the lowest stratum;
3. Each individual learns from the random individual in the higher level, and the highest level individual updates its position by learning from the random individual in the same level.

The number of individuals that it learns varies across different levels. Higher-level individuals learn fewer individuals compared to lower-level ones. This approach facilitates the exploration of low-level individuals and the exploitation of high-level individuals. Additionally, the performance of feature selection is influenced by the number of NL layers.

Yang et al. proposed a dynamic version of the method to determine the number of layers [42]. First, initialize an array with different numbers that represent the number of layers of the population. Then, at each iteration, select the number of strata for this iteration based on the probability of each number and record the performance using this number of strata at the end of this iteration. The more the fitness value of the current optimal solution in the array improves, the more likely it is to choose this integer in the next iteration . One major difficulty in determining the number of layers by this method would be the need to initially give a set of candidate integers. It is very time and effort consuming to give the set of layers, so an adaptive method is proposed to determine the number of layers, and the calculation formula for obtaining the number of layers (NL) is as follows:

1. First, use the following formula to calculate the Hamming value of pairwise in the population. Because the Hamming value can reflect the distance and similarity between individuals:

$$H_{i,j} = \sum_{d=1}^{D} \left| x_{i,d} - x_{j,d} \right| \tag{4}$$

   In the formula, $H_{i,j}$ is the Hamming distance between x_i and x_j, x_i and x_j are two different individuals in the population, and D is the dimension of the population.
2. Then, calculate the average Hamming value of the whole population:

$$Ave_t = mean\left(\sum_{i=1}^{i=N-1} \sum_{j=i+1}^{N} H_{i,j} \right) \tag{5}$$

   Ave_t is used to find the average Hamming value of the whole population, and N is the population size.
3. In the initial iteration, we will divide the whole population into more levels so that the population can avoid falling into local optimum at an early stage. At the end of the iteration, fewer levels will be obtained so that the population can conduct a more refined search in a small space. Improving the learning probability of dominant individuals is conducive to a more intensive use of the search space by groups. For this purpose, the rate of decline of the average Hamming value in the population is used to determine NL. There is the following formula:

$$rate = \frac{Ave_t}{Ave_0} \times N \tag{6}$$

$$NL = \begin{cases} 4, rate < 4 \\ fix(rate), 4 \leq rate \leq N \\ [N/2], other \end{cases} \tag{7}$$

   In the formula, rate is the decline rate of the average Hamming value in the population, N is the size of the population, and the $fix(rate)$ function is the maximum integer that returns, which is no greater than the rate.

3.3.2. Information Sharing Mechanism

In the process of searching, the search information of the population cannot be updated in time, which affects the convergence performance of the algorithm. Therefore, we propose an information sharing mechanism in which population particles cooperate to search under the guidance of their neighbors within a search cycle. Once a particle falls into the local optimal, the information sharing mechanism will adjust the search direction in time, avoid local mimina and search toward the population optimal direction. This paper divides the whole population into several levels. The information sharing mechanism uses the average location of the optimal solution from each level in the population. On the one hand, individuals in the population use optimal strategies to find better solutions. On the other

hand, it also obtains information from other individuals, with evolutionary information at different levels. Information from other levels might push stagnant particles back to life.

Since the simple difference operator cannot solve the diversity problem of the population well, an information sharing formula is added to the difference operator formula to solve the diversity problem and local optimal problem of the simple difference operator by combining the difference operator and information sharing. The optimal average position is calculated as:

$$\bar{x} = \frac{1}{p} \sum_{k=1}^{p} x_{lbest}^{k} \tag{8}$$

In the formula, p is the optimal number of selected individuals in each layer, $\bar{x}$ is the optimal average position, and x_{lbest}^{k} is the optimal particle in each layer.

The average location guidance mechanism of the information sharing mechanism is illustrated in Figure 3. The P point represents the guide point individuals selected in the population search space, while the other points (Xi) represent the individuals selected from the optimal part in each layer. These vectors provide the general direction for the next population update and enable escaping local optima.

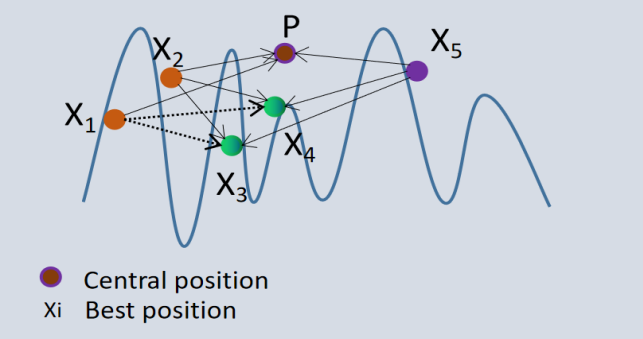


Figure 3. Diagram of information sharing mechanism.

3.3.3. Differential Operator with Multi-Strategy Integration

The difference algorithm [43] is a meta-heuristic random search algorithm based on group differences. The differential evolution [44] algorithm has the advantages of a simple principle, few controlled parameters and strong robustness. Different from the general evolutionary algorithm [45], the difference algorithm uses mutation operators and crossover operators to generate new particles [46]. The updating process of DE is realized by three steps: mutation, crossover and selection.

In the iteration of the algorithm, each individual particle will evolve to different degrees [47]. Due to the different characteristics of each particle, the learning strategies of each layer of particles are also different. According to the characteristics of different populations, different mutation strategies [48] and control parameters are used. At the same time, the mutation strategy [49,50] of individuals is determined by probability judgment for the common population to accelerate the convergence rate. Particles with higher fitness are more likely to approach the optimal location and exhibit superior accuracy in search ability. They play a pivotal role in guiding the population toward the optimal direction. Conversely, particles with lower fitness are more likely to deviate from the optimal position. However, they possess greater exploration capability, aiding in space expansion and avoiding local minima. So, there are three difference formulas to update each particle in the population:

1. The update operator performs a more refined search in a smaller area:

$$L_{i,j} = F \times \left(x_{gbest} - x_{i,j} \right) + F \times (x_{r1} - x_{r2}) + \sigma \times \left(\bar{x} - x_{i,j} \right) \tag{9}$$

2. In order to find the particles in the middle of the population carefully and extensively, we will give consideration to both:

$$L_{i,j} = F \times (x_{r1} - x_{i,j}) + F \times (x_{r1} - x_{r2}) + \sigma \times (\bar{x} - x_{i,j}) \quad (10)$$

3. It facilitates the population in avoiding local optima by enabling extensive searches for particles positioned in the middle of the population:

$$L_{i,j} = F \times \left(x_{gbest} - x_{worst}\right) + F \times \left(x_{gbest} - x_r\right) + \sigma \times (\bar{x} - x_{i,j}) \quad (11)$$

In the above formula, F is a random number in a [−1,1] interval, and $L_{i,j}$ is the location offset $learn_{i,current}$ mentioned above. x_{gbest} is the global optimal value in the population, and x_{worst} is the global worst in the population. $x_{i,j}$ is the current particle, while x_{r1} and x_{r2} are two different random particles in the interval with better fitness than the current particle. σ is a constant value of 0.04, and $\bar{x}$ is the average value of the current population.

4. Experiments and Results

4.1. Experimental Design and Parameters

This experiment is conducted on a WIN10 system, with an NVIDIA GTX 1660 graphics card, Inter Core i5-11400 processor, 2.6 GHz main frequency and 16 GB running memory. The 2020b version of Matlab is used.

In order to verify the correctness of the idea of feature selection based on the hierarchical learning crow algorithm, the dataset information, comparison algorithm information and experimental parameter settings for the experiment are given. The experimental results of classification accuracy and dimension reduction of this algorithm and the comparison algorithm are given. The following datasets are used (Table 1):

Table 1. Summary of Dataset.

Dataset	Features	Classes	Samples	Dataset	Features	Classes	Samples
haberman	3	2	306	australia	14	2	690
ecoli	8	6	336	zoo	16	7	101
balancescale	4	3	625	vehicle	18	4	846
iris	4	3	150	lymphography	18	2	148
glass	9	7	214	spectheart	22	2	267
contraceptive	9	3	1473	breastEW	30	2	569
tictactoe	9	2	958	inosphere	33	2	351
breastcancer	10	2	683	dermatology	34	6	366
wine	13	3	178	sonar	60	2	208

To create a meta-heuristic algorithm, the parameters of a specific algorithm need to be adjusted. Different algorithms have different algorithm parameters. These parameters are obtained through a large number of experiments, and the setting of these parameters is a complex and time-consuming task. There are parameters to the following algorithms: binary particle swarm optimization (BPSO) (M. Mafarja, Jarrar, et al., 2018 [9]), genetic algorithm (GA) (Sikora & Piramuthu, 2007 [11–13]), binary dragonfly optimization algorithm (BDA) (M. M. Mafarja et al., 2017 [10]), binary crow search algorithm (BCSA1) (Laabadi, Naimi, Amri, & Achchab, 2020 [20]), V-shaped binary crow search algorithm (BCSA2) (De Souza et al., 2018 [26]), binary gray wolf optimization (BGWO) (Emary, Zawbaa, & Hassanien, 2016 [17]), binary grasshopper optimization algorithm (BGOA) (M. Mafarja et al., 2019 [29]), and binary crow search algorithm with time-varying flight length (BCSA-TVFL) (Abhilasha et al., 2020 [25]).

In these algorithms, certain parameters are derived from extensive experiments conducted by other researchers. These parameters remain constant throughout the iterations the algorithm. In BPSO, the authors use the inertia weight, particle renewal rate, global

learning factor, individual learning factor and other parameters. The BCSA uses two parameters: flight length and perceived probability. The BDA uses separation weight, alignment weight, cohesion weight, food factor, enemy factor and other parameters. The GA uses crossover probability, mutation probability and other parameters. BCSA-TVFL has the same parameters as the BCSA, with an additional interval [1.5, 2.5]. A value is set for each parameter during initialization, which remains constant through all iterations. The values of these parameters are empirically determined, so setting these parameters is a complex and time-consuming task. Therefore, we can say that the fewer the number of parameters, the better. The following are the parameters used in the above article (Table 2).

Table 2. Parameter setting for AHL-CSA and other comparative algorithms.

Algorithm	Parameter	Values
BPSO	Inertia of Particle	0.9
	c1	0.5
	c2	0.5
BCSA	Flight Length fl	2
	Awareness Probability AP	0.1
BDA	Separation s	0.1
	Alignment a	0.1
	Cohesion c	0.7
	Attraction f	1
	Distraction e	1
GA	Crossover Probability	0.8
	Mutation Probability	0.02
BCSA-TVFL	Awareness Probability AP	0.1
	Flight Length fl	[1.5, 2.5]
AHL-CSA	AP	0.1
	F	$[-1, 1]$
	σ	0.04
	fl	[1.5, 2.5]

4.2. Experimental Results and Analysis

Abhilasha et al. compared eight transfer functions in terms of average fitness [25], average accuracy and feature selection ratio. Through a lot of experiments, it can be concluded that the best transfer function is S1 type transfer function, which is Formula (1). Therefore, we use this transfer function in the following experiments.

To explore the effect of the proposed optimization strategy on the performance of the crow search algorithm, we compare the results of the AHL-CSA algorithm with other algorithms. The mentioned algorithm and other representative algorithms use 30 independent replicate experiments to guarantee the validity of the results. The dataset was divided into a 20% test set and an 80% training set. kNN classifiers are stable, simple, and have few parameters. To validate the classification of feature subsets, we used k = 5 as the nearest neighbors of the kNN classifier. The number of populations N in each experiment was set to 50. There were three evaluation criteria. The average fitness and average accuracy, standard deviation, feature selection rate, and maximum accuracy for 30 independent runs were obtained under the same experimental conditions. In the table, the best results are shown in bold, and the number of optimal results obtained for each algorithm is counted.

4.2.1. Results Comparison

The AHL-CSA proposed in this paper is compared with other algorithms. This is a comparison of three numbers. Table 3 records accuracy, Table 4 records fitness value, and Table 5 records feature ratio. These three numbers are the average of 30 separate operations. At the same time, the deviation value obtained by each algorithm is recorded to reflect the stability of each algorithm. After recording the data obtained in each experiment, the superiority of the algorithm is proved from three aspects: the number of times that

the algorithm ranks first, the number of times that the algorithm ranks first with other algorithms, and the number of times that the algorithm ranks last. Finally, each algorithm has a maximum precision to reflect the superiority of the algorithm.

Table 3. Average accuracy comparison of AHL-CSA with competitors.

Dataset		BCSA1	BCSA2	BGWO	BGOA	BPSO	BDA	GA	BCSA-TVFL	AHL-CSA
haberman	Avg	0.8361	0.8197	0.8361	0.7541	0.7153	0.7705	0.7869	0.8361	**0.8420**
	STD	0.0056	0.0810	0.0572	0.0129	0.0724	0.0000	0.0000	0.0000	0.0000
ecoli	Avg	0.8906	0.8507	0.8791	0.8652	0.8932	0.8945	0.8507	**0.9255**	0.8955
	STD	0.0067	0.0000	0.0060	0.0027	0.1071	0.0000	0.0000	0.0057	0.0000
balancescale	Avg	**0.9200**	0.8640	0.8240	0.8160	0.6240	0.8380	0.7760	0.8400	0.8240
	STD	0.0040	0.0032	0.0000	0.0036	0.1507	0.3450	0.0000	0.0000	0.0000
iris	Avg	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.8700	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	0.0000	0.0038	0.0061	0.0004	0.2098	0.0000	0.0000	0.0000	0.0000
glass	Avg	0.8079	0.7659	0.8056	0.7128	0.7798	0.7686	0.7760	0.7721	**0.8095**
	STD	0.0108	1.0460	0.0126	0.0213	1.3287	1.2910	1.4369	1.5450	0.0000
contraceptive	Avg	0.5475	0.5306	0.5361	0.5525	0.4426	**0.5612**	0.5544	0.5180	0.5442
	STD	0.0074	0.0042	0.0033	0.0369	0.2085	0.2940	0.0092	0.0086	0.0000
tictactoe	Avg	0.8300	0.8534	0.8162	0.8159	0.7990	0.8469	0.7887	0.7813	**0.8562**
	STD	0.0000	0.0000	0.0300	0.0048	0.0009	0.0054	0.0234	0.0303	0.0000
breastcancer	Avg	**1.0000**	0.9897	**1.0000**	0.9837	0.8600	0.9712	**1.0000**	**1.0000**	**1.0000**
	STD	0.0018	0.0036	0.0035	0.0032	0.1400	0.0000	0.0000	0.0034	0.0000
wine	Avg	**1.0000**	**1.0000**	0.9838	**1.0000**	0.7952	**1.0000**	**1.0000**	**1.0000**	**1.0000**
	STD	0.0000	0.0000	0.0144	0.0087	0.0973	0.0000	0.0000	0.0167	0.0000
australian	Avg	0.8915	0.8671	0.8778	0.8614	0.7536	0.8913	0.8952	0.8425	**0.9130**
	STD	0.0084	0.0177	0.0126	0.0212	0.0782	0.0000	0.0037	0.0223	0.0000
zoo	Avg	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.8650	0.9500	**1.0000**	**1.0000**	**1.0000**
	STD	0.0000	0.0000	0.0091	0.0153	0.0765	0.0000	0.0000	0.0260	0.0000
vehicle	Avg	0.7278	0.7694	0.7641	0.7704	0.6649	0.7700	0.7333	0.7598	**0.7759**
	STD	0.0071	0.0065	0.0129	0.0115	0.0524	0.0059	0.0150	0.0094	0.0104
lymphography	Avg	0.9161	0.9299	0.8724	0.8782	0.8781	0.8920	0.8862	0.8655	**0.9945**
	STD	0.0174	0.0063	0.0242	0.0297	0.0123	0.0234	0.0161	0.0209	0.0000
spectheart	Avg	0.9157	0.8654	0.8836	0.9182	0.8026	0.9390	0.9390	0.8830	**0.9417**
	STD	0.0108	0.0065	0.0112	0.0194	0.0443	0.0081	0.0081	0.0152	0.0095
breastEW	Avg	0.9926	0.9646	0.9906	0.9826	0.9106	0.9900	0.9817	0.9611	**1.0000**
	STD	0.0057	0.0000	0.0061	0.0064	0.0250	0.0022	0.0022	0.0055	0.0044
inosphere	Avg	0.9610	0.9343	0.8955	0.8710	0.9543	0.9362	**0.9614**	0.9216	0.9514
	STD	0.0083	0.0096	0.0046	0.0127	0.0276	0.0090	0.0067	0.0068	0.0125
dermatology	Avg	**1.0000**	**1.0000**	0.9712	**1.0000**	**1.0000**	**1.0000**	0.9978	0.9753	**1.0000**
	STD	0.0000	0.0000	0.0122	0.0000	0.0000	0.0000	0.0051	0.0091	0.0000
sonar	Avg	0.9526	0.8919	0.9569	0.9163	0.7667	0.9854	0.9805	0.9563	**0.9890**
	STD	0.0108	1.1864	1.2549	0.0202	0.0418	0.8444	0.0099	2.0816	0.0124
Ranking	W | T | L	**1 | 5 | 12**	**0 | 4 | 14**	**0 | 3 | 15**	**0 | 4 | 14**	**0 | 1 | 17**	**1 | 3 | 14**	**1 | 4 | 13**	**1 | 4 | 13**	**9 | 5 | 4**

The transfer function adopts the above Equation (1), as used by Abhilasha et al. According to the average accuracy measurement, the transfer function S1 ranks first, with an average fitness and standard deviation of 30 independent runs (Table 6). The transfer function S1 has the best result 70% of the time [25]. The results of Mafarja et al. show the importance of the selection of transfer function in the algorithm, because selecting the appropriate transfer function can significantly improve the performance of the algorithm. Furthermore, it is clear that the behavior of adjusting the transfer function through the optimization process has a big impact on improving the performance of the algorithm [23]. To better evaluate the proposed method, the experimental results of a comparison between AHL-CSA and other algorithms are calculated on the same computer. The standard deviation of each datum effectively reflects the quality of the data processed by the algorithm. The higher the standard deviation, the more unstable the algorithm

processing results, and the less convincing the results. The lower the standard deviation, the better the stability of the algorithm for data processing, and the more convincing the data results.

The faster the convergence curve reaches stability, the better the data processing effect of the algorithm and vice versa. The convergence curve shows the exploration and development ability of the algorithm, and whether the two methods can be compromised in the process of evolution is also an evaluation criterion of the algorithm.

To ensure that the experimental comparison is fair and equitable, the ending criteria should be consistent with the original text. In each algorithm, the population size is 50, the number of iterations is 100, and the results of each algorithm are generated from 30 independent runs.

Table 4. Average fitness comparison of AHL-CSA with competitors.

Dataset		BCSA1	BCSA2	BGWO	BGOA	BPSO	BDA	GA	BCSA-TVFL	AHL-CSA
haberman	Avg	0.1690	0.1852	0.1690	0.2501	0.2014	0.2339	0.2177	0.1690	**0.1500**
	STD	0.0055	0.0131	0.0316	0.0101	0.0000	0.0120	0.0126	0.0130	0.0000
ecoli	Avg	0.1697	0.1535	0.1256	0.1382	0.1254	0.1126	0.1535	0.1578	**0.1106**
	STD	0.0000	0.0000	0.0065	0.0032	0.0000	0.0012	0.0085	0.0048	0.0000
balancescale	Avg	**0.0892**	0.1446	0.1842	0.1922	0.1763	0.1705	0.2308	0.1684	0.1842
	STD	0.0043	0.0082	0.0155	0.0075	0.0110	0.0230	0.0156	0.0086	0.0000
iris	Avg	0.0355	0.0075	0.0109	**0.0050**	0.0055	**0.0050**	**0.0050**	0.0355	**0.0050**
	STD	0.0000	0.0000	0.0047	0.0050	0.0000	0.0000	0.0000	0.0000	0.0000
glass	Avg	0.1957	0.2412	0.1967	0.2228	0.2840	0.3109	0.2626	0.3058	**0.1930**
	STD	0.0084	0.0121	0.0117	0.0173	0.0159	0.0211	0.0166	0.0143	0.0000
contraceptive	Avg	0.4536	0.4704	0.4642	0.4508	0.4759	**0.4399**	0.4489	0.4834	0.4759
	STD	0.0000	0.0121	0.0054	0.0033	0.0036	0.0140	0.0131	0.00092	0.0000
tictactoe	Avg	0.1800	0.1551	0.1909	0.1921	0.1603	0.1596	0.2175	0.2235	**0.1521**
	STD	0.0000	0.0000	0.0278	0.0039	0.0009	0.0063	0.0213	0.0287	0.0044
breastcancer	Avg	0.0052	0.0151	0.0197	0.0217	0.0192	0.0325	0.0091	0.0288	**0.0040**
	STD	0.0010	0.0035	0.0035	0.0029	0.0000	0.0000	0.0016	0.0031	0.0000
wine	Avg	0.0024	0.0044	0.0205	0.0073	**0.0015**	0.0032	0.0016	0.0966	0.0039
	STD	0.0105	0.0012	0.0144	0.0056	0.0088	0.0034	0.0032	0.0064	0.0067
australian	Avg	0.1107	0.1354	0.1250	0.1411	0.1127	0.1119	0.1070	0.1597	**0.0891**
	STD	0.0086	0.0180	0.0131	0.0210	0.0017	0.0110	0.0033	0.0226	0.0000
zoo	Avg	0.0027	0.0053	0.0062	0.0103	0.0535	0.0520	0.0027	0.0027	**0.0027**
	STD	0.0049	0.0039	0.0069	0.0069	0.0000	0.0043	0.0025	0.0051	0.0000
vehicle	Avg	0.2745	0.2359	0.2388	0.2327	**0.2086**	0.2200	0.2687	0.2427	0.2300
	STD	0.0070	0.0067	0.0131	0.0111	0.0066	0.0057	0.0154	0.0093	0.0116
lymphography	Avg	0.0867	0.0748	0.1321	0.1262	0.1250	0.1102	0.1161	0.1381	**0.0049**
	STD	0.0168	0.0066	0.0233	0.0288	0.0123	0.0114	0.0153	0.0205	0.0072
spectheart	Avg	0.0879	0.1400	0.1202	0.0873	**0.0622**	0.0645	0.1161	0.1207	0.0819
	STD	0.0100	0.0064	0.0109	0.0190	0.0000	0.0078	0.0089	0.0144	0.0091
breastEW	Avg	0.0120	0.0419	0.0151	0.0231	0.0321	0.0042	0.0216	0.0433	**0.0033**
	STD	0.0053	0.0000	0.0057	0.0060	0.0044	0.0018	0.0022	0.0052	0.0041
inosphere	Avg	**0.0404**	0.0703	0.1037	0.1328	0.0550	0.0659	0.0409	0.0679	0.0504
	STD	0.0081	0.0102	0.0048	0.0127	0.0086	0.0089	0.0064	0.0043	0.0125
dermatology	Avg	0.0031	0.0055	0.0356	0.0035	0.0034	**0.0027**	0.0050	0.0296	0.0036
	STD	0.0000	0.0000	0.0117	0.0007	0.0004	0.0000	0.0051	0.0087	0.0000
sonar	Avg	0.0104	0.1132	0.0484	0.0890	0.0949	0.0172	0.0217	0.1568	**0.0141**
	STD	0.0096	0.0120	0.0164	0.0040	0.0091	0.0560	0.0098	0.0186	0.0155
Ranking	W|T|L	**1|1|16**	**0|0|18**	**0|0|18**	**0|1|17**	**3|0|15**	**2|1|15**	**0|1|17**	**0|0|18**	**10|1|7**

Table 5. Feature selection ratio of AHL-CSA and competitors.

Dataset		BCSA1	BCSA2	BGWO	BGOA	BPSO	BDA	GA	BCSA-TVFL	AHL-CSA
haberman		0.67	0.67	0.67	0.67	0.58	0.67	0.67	0.67	**0.53**
ecoli		0.71	0.57	0.59	**0.47**	0.51	0.71	0.57	0.76	0.71
balancescale		1.00	1.00	1.00	1.00	**0.60**	1.00	1.00	1.00	1.00
iris		**0.25**	0.75	0.97	0.50	0.47	0.65	0.50	**0.25**	0.50
glass		0.55	0.64	0.39	0.51	0.43	0.44	**0.33**	0.41	0.44
contraceptive		0.56	0.57	0.49	0.77	0.50	0.55	0.77	0.63	**0.44**
tictactoe		1.00	1.00	0.89	0.98	**0.61**	0.78	0.82	0.70	1.00
breastcancer		**0.33**	0.49	0.41	0.55	0.48	0.40	0.40	0.39	0.40
wine		**0.24**	0.44	0.44	0.45	0.42	0.32	0.17	0.42	0.39
australian		0.33	0.39	0.39	0.38	0.46	0.43	0.33	0.38	**0.30**
zoo		0.27	0.53	0.46	0.54	0.49	0.63	**0.28**	0.40	0.32
vehicle		0.50	0.76	0.53	0.39	0.52	0.46	**0.28**	0.49	0.43
lymphography		0.36	0.54	0.58	0.55	**0.28**	0.32	0.35	0.50	0.39
spectheart		**0.44**	0.67	0.51	0.63	0.47	0.51	0.46	0.49	**0.44**
breastEW		0.47	0.68	0.57	0.59	0.49	0.54	0.38	0.47	**0.37**
inosphere		0.35	0.52	0.42	0.50	0.46	0.27	0.27	0.44	**0.23**
dermatology		0.31	0.55	0.50	0.24	0.24	0.27	**0.21**	0.52	0.36
sonar		0.32	0.61	0.57	0.61	0.48	**0.27**	0.24	0.47	0.32
Ranking	W|T|L	**3|2|13**	**0|0|18**	**0|0|18**	**1|0|17**	**3|0|15**	**1|0|17**	**4|0|14**	**0|1|17**	**5|1|12**

Table 6. Maximum accuracy obtained in 30 runs by AHL-CSA and competitors.

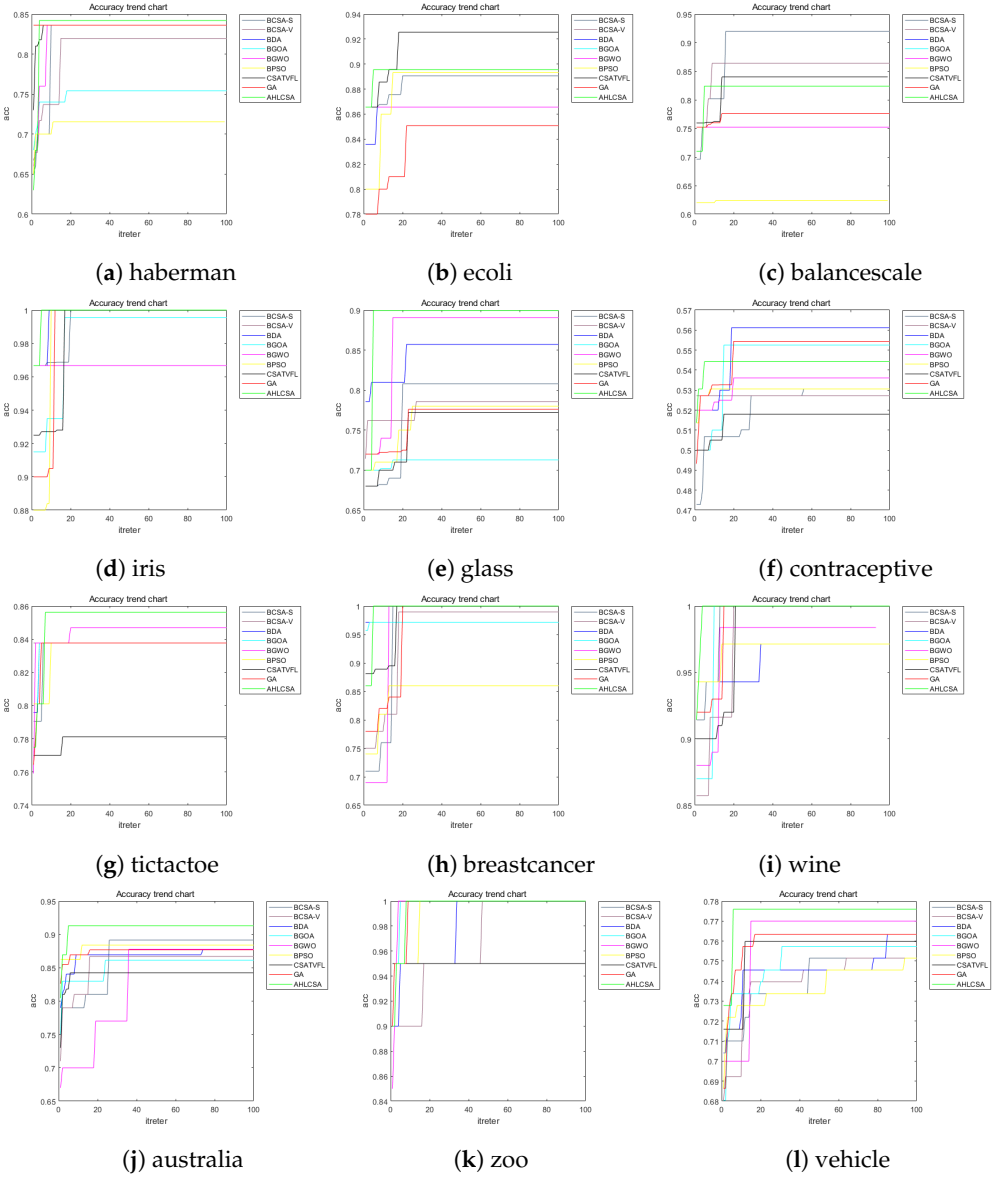
Dataset		BCSA1	BCSA2	BGWO	BGOA	BPSO	BDA	GA	BCSA-TVFL	AHL-CSA
haberman		0.8361	0.8197	0.8361	0.7541	0.8033	0.7705	0.7869	0.8361	**0.8420**
ecoli		0.8912	0.8507	0.8806	0.8657	0.9012	0.8955	0.8507	**0.9266**	0.8955
balancescale		**0.9200**	0.8640	0.8240	0.8160	0.8320	0.8720	0.7760	0.8400	0.8240
iris		**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
glass		**0.8095**	0.7659	0.8056	0.7857	0.7798	0.7686	0.7826	0.7826	**0.8095**
contraceptive		0.5476	0.5306	0.5408	0.5544	0.5204	**0.5612**	0.5544	0.5306	0.5442
tictactoe		0.8300	0.8534	0.8325	0.8168	0.8482	0.8470	0.8115	0.8429	**0.8562**
breastcancer		**1.0000**	0.9928	**1.0000**	0.9856	0.9712	0.9712	**1.0000**	**1.0000**	**1.0000**
wine		**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
australian		0.8986	0.8986	0.8913	0.8986	0.8900	0.8913	0.8986	0.8768	**0.9130**
zoo		**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9500	**1.0000**	**1.0000**	**1.0000**	**1.0000**
vehicle		0.7396	0.7800	0.7870	0.7800	0.7396	0.7800	0.7556	0.7751	**0.7866**
lymphography		0.9310	0.9310	0.8966	0.9655	0.8912	0.8966	0.8966	0.8966	**0.9945**
spectheart		0.9434	0.8679	0.9057	0.9434	0.9057	0.9434	0.9410	0.9057	**0.9490**
breastEW		**1.0000**	0.9646	**1.0000**	**1.0000**	0.9469	**1.0000**	0.9823	0.9735	**1.0000**
inosphere		**0.9714**	0.9571	0.9143	0.9000	0.9543	0.9571	**0.9714**	0.9425	**0.9714**
dermatology		**1.0000**	**1.0000**	0.9812	**1.0000**	**1.0000**	**1.0000**	0.9979	0.9863	**1.0000**
sonar		**1.0000**	0.9268	**1.0000**	0.9512	0.8049	**1.0000**	**1.0000**	0.8780	**1.0000**
Ranking	W|T|L	**1|8|9**	**0|4|14**	**0|6|12**	**0|5|13**	**0|3|15**	**1|7|10**	**0|6|12**	**1|4|13**	**6|9|3**

The average accuracy of Table 3 shows that the proposed algorithm achieves very good results. The proposed algorithm achieves the best results on nine datasets, and there are five datasets where it achieves the same results as the other algorithms. The standard deviation of the proposed algorithm for each dataset is almost all 0, which indicates that the proposed algorithm has good stability. In Table 4, for the mean accuracy, the proposed algorithm outperforms other representative algorithms on 10 datasets and achieves the same results as other algorithms on one dataset. In Table 5, for the feature selection rate, the proposed algorithm achieves the best results on five datasets and achieves the same results as the compared algorithms on one dataset.

The adaptive hierarchical strategy, differential operator with multi-strategy integration and information sharing mechanism proposed in this paper can effectively improve the performance of the algorithm. Because the proposed three innovations can effectively improve the diversity of the population, each particle has the potential to achieve the optimal.

4.2.2. Results of Convergence Comparison

The following Figure 4 is the accuracy convergence diagram of each algorithm. From the following 18 datasets, we can see that the AHL-CSA algorithm converges faster than other algorithms, and the final accuracy value is also better. In each algorithm, the population size is 50 and the number of iterations is 100 so as to run the convergence diagram of the accuracy algorithm.



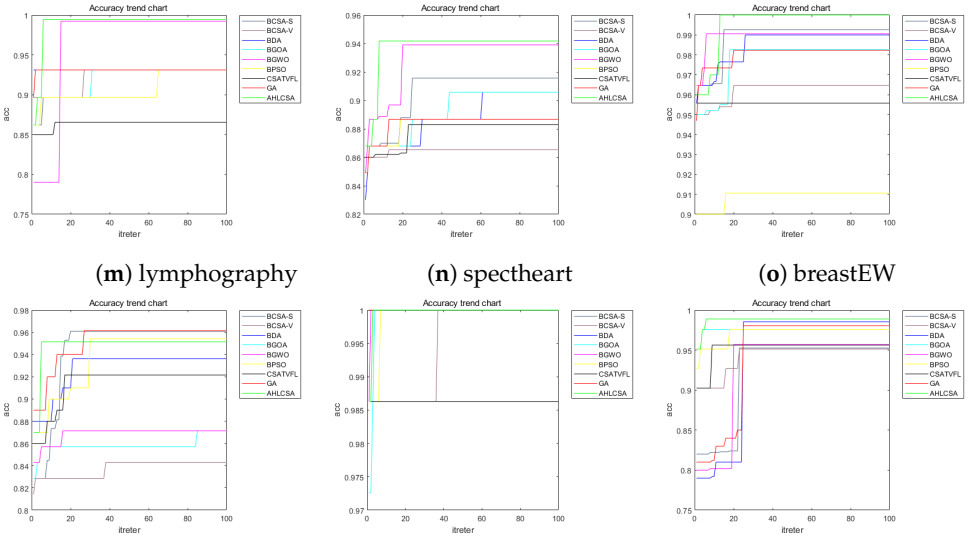
(**a**) haberman (**b**) ecoli (**c**) balancescale

(**d**) iris (**e**) glass (**f**) contraceptive

(**g**) tictactoe (**h**) breastcancer (**i**) wine

(**j**) australia (**k**) zoo (**l**) vehicle

Figure 4. *Cont.*



(**m**) lymphography (**n**) spectheart (**o**) breastEW

(**p**) inosphere (**q**) dermatology (**r**) sonar

Figure 4. Convergence curves of AHL-CSA and other representative wrapper methods on 18 datasets.

As can be seen from the convergence rate table above, the algorithm can converge quickly in most datasets and achieve good results in algorithm accuracy. This also shows that the algorithm has a strong ability to avoid the local mimina problem.

4.2.3. Wilcoxon Rank-Sum Test

The rank-sum test (Table 7) is used to infer whether there is a difference between two population distribution locations from two independent measurements or samples of rank data. This is a non-parametric paired two-sample *t* test that can be used to compare two independent sample groups. It is used when the data are not normally distributed and still do not meet the normality requirement after some numerical conversion attempts. It compares the non-normal sample values of two independent sample groups.

Table 7. The *p*-value of the Wilcoxon rank-sum test results.

Dataset	BCSA1	BCSA2	BGWO	BGOA	BPSO	BDA	GA	BCSA-TVFL
haberman	<0.05	<0.05	<0.05	**0.092**	<0.05	<0.0556	<0.05	<0.05
ecoli	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	**0.068**
balancescale	<0.05	**0.135**	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
iris	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
glass	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
contraceptive	<0.05	<0.05	<0.05	<0.05	**0.0745**	**0.056**	<0.05	<0.05
tictactoe	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
breastcancer	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
wine	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
australian	**0.073**	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	**0.0657**
zoo	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
vehicle	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
inosphere	<0.05	<0.05	<0.05	**0.081**	<0.05	<0.05	<0.05	<0.05
lymphography	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
spectheart	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
breastEW	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
dermatology	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05
sonar	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05	<0.05

To test the validity of the results, a Wilcoxon rank-sum test was performed on the data [51]. The *p*-values were tested at the significance level of 0.05 by the accuracy of 30

independent runs for each dataset. *p*-values greater than 0.05 are indicated in bold. From this table, we can see that the data are statistically valid.

4.2.4. Results Comparison with Well-Known Filter Methods in Literature

Filter methods have gained popularity in feature selection as they require low computational cost and show high processing speed. These methods reduce the dimension of feature space by evaluating and ranking features, thereby finding the optimal predictive subset. The following table compares the representative filter methods and our proposed algorithm in terms of the maximum accuracy.

Filter methods often ignore the interrelationships between features. They typically select features by measuring a single feature, such as variance, correlation coefficient, Chi-square test, and information gain. These indicators solely consider the predictive power of each feature itself, however, neglecting the relationship between features. There may be complex interactions between features that can have an important impact on the accuracy of classification or regression models.

Another type of approach is to use a wrapper or embedded method. These methods directly consider feature selection as part of model training, take into account the interrelationships between features, and optimize through techniques such as cross-validation. However, these methods are more computationally intensive and time consuming than filter methods. Therefore, in practical applications, we need to weigh the relationship between computational cost and accuracy and choose the appropriate method.

Table 8 shows that the proposed algorithm takes the lead compared with the filter method. The hierarchical strategy makes the number of candidate samples of individuals differ at every level, thus improving the population diversity. The introduction of a difference strategy makes the particles with poor fitness mutate and cross, increasing the diversity of particles, boosting the possibility of obtaining solutions in space, and thus obtaining stronger global search ability. Furthermore, the combination of a hierarchical strategy and differential strategy takes our algorithm to a better level, achieving promising results.

Table 8. Classification accuracy results of filter-based methods compared to the AHL-CSA.

Dataset		ReliefF	InfoGain	CFS	GainRatio	AHL-CSA
haberman		0.7483	0.7483	0.7483	0.7483	**0.8420**
ecoli		0.8452	0.8452	0.8541	0.7887	**0.8955**
balancescale		0.7712	**0.9040**	0.6352	**0.9040**	0.8240
iris		0.9600	0.9600	0.9600	0.9600	**1.0000**
glass		0.4953	0.5000	0.4953	0.4719	**0.8995**
contraceptive		0.4460	0.5010	0.5547	0.4915	**0.5612**
tictactoe		0.0938	0.0598	0.1560	0.1956	**0.8562**
breastcancer		0.5900	0.5944	0.6002	0.5944	**1.0000**
wine		0.9444	0.8889	0.7778	0.8878	**1.0000**
australian		0.8434	0.7478	0.8188	0.7478	**0.9130**
zoo		0.8000	0.8500	0.8000	0.8500	**1.0000**
vehicle		0.5917	0.6052	0.6095	0.6059	**0.7866**
lymphography		0.7000	0.7214	0.6954	0.7516	**0.9945**
spectheart		0.8623	0.6713	0.7854	0.7958	**0.9417**
breastEW		0.9241	0.8923	0.8756	0.9123	**1.0000**
inosphere		0.8471	0.8490	0.9202	0.8575	**0.9514**
dermatology		0.8493	0.9381	0.8662	0.8804	**1.0000**
sonar		0.5238	0.1905	0.3095	0.4320	**0.9634**
Ranking	W|T|L	**0|0|18**	**0|1|17**	**0|0|18**	**0|1|17**	**17|0|1**

5. Conclusions and Future Work

In this paper, a meta-heuristic search strategy is used in feature selection. Problems such as local optimality and population diversity cannot be guaranteed and may arise

when using this method. Based on a meta-heuristic crow search algorithm, we propose an adaptive hierarchical learning crow search algorithm (AHL-CSA). To improve the population diversity, we adopted an adaptive stratification strategy. This strategy divides the population into several levels according to fitness level. In addition, we also use multi-strategy differential and information sharing strategies to further avoid the local optimal problem. Multi-strategy difference is an improved method based on the differential evolution algorithm, which uses multiple randomly selected strategy vectors to generate new individuals and selects the best individuals from them to join the next generation population. This method avoids the current local optimum and enables the discovery of a superior alternative. To verify the validity and accuracy of these methods, 18 UCI standard datasets were selected for fitness, classification accuracy and feature selection ratio tests. The comparison table and graph clearly demonstrate the algorithm's strong convergence speed and effectiveness. The proposed method can effectively balance exploration and exploitation in the process of evolution. The proposed algorithm outperforms other representative algorithms in terms of accuracy, selection ratio, and fitness value. Finally, we have uploaded the algorithm code to GitHub: https://github.com/eagdog/Algorithm-code.git (accessed on 21 June 2023).

Although the proposed algorithm utilizes multiple strategies to calculate various levels, it incurs a slight time disadvantage. Furthermore, the algorithm is limited to text datasets on UCI and cannot process other data types. In the future work, we will try this algorithm for feature selection in high-dimensional data so that this algorithm can better play to its strengths. We will also focus on applying the crow search algorithm to a wider range of applications. The algorithm has applications not only in feature selection but also in remote sensing image classification, medical image classification, and various fused object detection methods. Our future research direction aims to fulfill the diverse needs of different application areas.

Author Contributions: Conceptualization, Z.Y.; Methodology, Z.Y.; Software, Z.Y.; Validation, Z.Y.; Investigation, Z.Y.; Writing—original draft, Z.Y. and B.G.; Writing—review and editing, Y.C., Y.W., X.Y. and X.L.; Supervision, Y.C. and X.L.; Project administration, Y.C. and X.L.; Funding acquisition, Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by The National Key Research and Development Program of China under Grant 2020YFB1313900, Innovation Fund of Hubei Key Laboratory of Intelligent Robotics under Grant HBIRL202107, Wuhan Institute of Technology Scientific Research Fund under Grant 21QD53, Science and Technology Research Project of Hubei Provincial Education Department under Grant Q20221501, Shenzhen Science and Technology Program under Grant JCYJ20200109115201707 and JCYJ20220818101408019, Stable Supporting Program for Universities of Shenzhen under Grant 20220812102547001, National Natural Science Foundation of China under Grant 62102268, Shenzhen Polytechnic Research Fund under Grant 6022312044K, Shenzhen Polytechnic Research Fund under Grant 6023310030K.

Data Availability Statement: The dataset used in the experiments is publicly available from UCI https://archive.ics.uci.edu/ml/datasets.php (accessed on 17 April 2022).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CSA	crow search algorithm
AHLCSA	adaptive hierarchical learning crow search algorithm
PSO	particle swarm optimization algorithm
GA	genetic algorithm
BDA	dragonfly optimization algorithm
GOA	grasshopper optimization algorithm
GWO	gray wolf optimization algorithm
CSATVFL	crow search algorithm with time-varying flight length

References

1. Zhang, L.; Li, N.; Li, Z. An Overview on Supervised Semi-structured Data Classification. In Proceedings of the 2021 IEEE 8th International Conference on 526 Data Science and Advanced Analytics (DSAA), Porto, Portugal, 6–9 October 2021; pp. 1–10.
2. Nie, P.Y.; Ma, C.F. A trust region filter method for general non-linear programming. *Appl. Math. Comput.* **2006**, *172*, 1000–1017. [CrossRef]
3. Kim, P.; Kim, J.; Jung, W.K.; Bu, S. An error embedded method based on generalized Chebyshev polynomials. *J. Comput. Phys.* **2016**, *306*, 55–72. [CrossRef]
4. Maldonado, S.; Weber, R. A wrapper method for feature selection using Support Vector Machines. *Inf. Sci.* **2009**, *179*, 2208–2217. [CrossRef]
5. Nahid, A.A.; Sikder, N.; Abid, M.H.; Toma, R.N.; Talin, I.A.; Ali, L.E. Home Occupancy Classification Using Machine Learning Techniques along with Feature Selection. *Int. J. Eng. Manuf.* **2022**, *12*, 38. [CrossRef]
6. Oh, M.; Yeak, S. A Hybrid Multiscale Finite Cloud Method and Finite Volume Method in Solving High Gradient Problem. *Int. J. Comput. Methods* **2022**, *19*, 271–279. [CrossRef]
7. Li, Y.; Han, T.; Han, B.; Zhao, H.; Wei, Z. Whale Optimization Algorithm with Chaos Strategy and Weight Factor. *J. Phys. Conf. Ser.* **2019**, *1213*, 032004. [CrossRef]
8. Ming, F.; Gong, W.; Wang, L. A Two-Stage Evolutionary Algorithm With Balanced Convergence and Diversity for Many-Objective Optimization. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 6222–6234. [CrossRef]
9. Tawhid, M.A.; Dsouza, K.B. Hybrid Binary Bat Enhanced Particle Swarm Optimization Algorithm for solving feature selection problems. *Appl. Comput. Inform.* **2018**, *16*, 117–136. [CrossRef]
10. Hammouri, A.I.; Mafarja, M.; Al-Betar, M.A.; Awadallah, M.A.; Abu-Doush, I. An improved Dragonfly Algorithm for feature selection. *Knowl.-Based Syst.* **2020**, *203*, 106131.
11. Kabir, M.M.; Shahjahan, M.; Murase, K. A new local search based hybrid genetic algorithm for feature selection. *Neurocomputing* **2011**, *74*, 2914–2928. [CrossRef]
12. Weidong, L.I.; Suhayb, M.K.; Thangavelu, L.; Marhoon, H.A.; Pustokhina, I.; Alqsair, U.F.; El-Shafay, A.S.; Alashwal, M. Implementation of AdaBoost and genetic algorithm machine learning models in prediction of adsorption capacity of nanocomposite materials. *J. Mol. Liq.* **2022**, *350*, 118527.
13. Hiremath, G.S.; Kumar, N.; CL, S.N. A Study on Ancient Temple Structural Elements Recognition Using Genetic Algorithm. *Int. J. Eng. Manuf.* **2023**, *3*, 34–47. [CrossRef]
14. Shunmugapriya, P.; Kanmani, S. A hybrid algorithm using ant and bee colony optimization for feature selection and classification (AC-ABC Hybrid). *Swarm Evol. Comput.* **2017**, *36*, 27–36. [CrossRef]
15. Karaboga, D.; Basturk, B. On the performance of artificial bee colony (ABC) algorithm. *Appl. Soft Comput.* **2008**, *8*, 687–697. [CrossRef]
16. Zhang, M.; Palade, V.; Wang, Y.; Ji, Z. Attention-based word embeddings using Artificial Bee Colony algorithm for aspect-level sentiment classification. *Inf. Sci.* **2021**, *545*, 713–738. [CrossRef]
17. Al-Tashi, Q.; Abdulkadir, S.J.; Rais, H.M.; Mirjalili, S.; Alhussian, H. Binary Optimization Using Hybrid Grey Wolf Optimization for Feature Selection. *IEEE Access* **2019**, *7*, 39496–39508. [CrossRef]
18. Mafarja, M.; Aljarah, I.; Faris, H.; Hammouri, A.I.; Ala'M, A.Z.; Mirjalili, S. Binary grasshopper optimisation algorithm approaches for feature selection problems. *Expert Syst. Appl.* **2019**, *117*, 267–286. [CrossRef]
19. Mafarja, M.M.; Mirjalili, S. Whale Optimization Approaches for Wrapper Feature Selection. *Appl. Soft Comput.* **2017**, *62*, 441–453. [CrossRef]
20. Oliva, D.; Hinojosa, S.; Cuevas, E.; Pajares, G.; Avalos, O.; Galvez, J. Cross entropy based thresholding for magnetic resonance brain images using Crow Search Algorithm. *Expert Syst. Appl.* **2017**, *79*, 164–180. [CrossRef]
21. Hu, J. A Particle Swarm Optimization Algorithm with Distributed Adaptively Weighted Delays. *Adv. Appl. Math.* **2021**, *10*, 753–762. [CrossRef]
22. Mafarja, M.; Jarrar, R.; Ahmad, S.; Abusnaina, A.A. Feature Selection Using Binary Particle Swarm Optimization with Time Varying Inertia Weight Strategies. In Proceedings of the International Conference on Future Networks and Distributed Systems, Amman, Jordan, 26–27 June 2018; pp. 1–9.
23. Mafarja, M.; Aljarah, I.; Heidari, A.A.; Faris, H.; Fournier-Viger, P.; Li, X.; Mirjalili, S. Binary Dragonfly Optimization for Feature Selection using Time-Varying Transfer functions. *Knowl.-Based Syst.* **2018**, *161*, 185–204. [CrossRef]
24. Bs, A.; Fg, B. A new approach to generate pattern-efficient sets of non-dominated vectors for multi-objective optimization—ScienceDirect. *Inf. Sci.* **2020**, *530*, 22–42.
25. Chaudhuri, A.; Sahu, T.P. Feature selection using Binary Crow Search Algorithm with time varying flight length. *Expert Syst. Appl.* **2020**, *168*, 114288. [CrossRef]
26. Rizk-Allah, R.M.; Hassanien, A.E.; Bhattacharyya, S. Chaotic Crow Search Algorithm for Fractional Optimization Problems. *Appl. Soft Comput.* **2018**, *71*, 1161–1175. [CrossRef]
27. Chung, S.W.; Freund, J.B. An optimization method for chaotic turbulent flow. *J. Comput. Phys.* **2022**, *457*, 111077. [CrossRef]
28. Jafari-Asl, J.; Azizyan, G.; Monfared, S.; Rashki, M.; Andrade-Campos, A.G. An enhanced binary dragonfly algorithm based on a V-shaped transfer function for optimization of pump scheduling program in water supply systems (case study of Iran). *Eng. Fail. Anal.* **2021**, *123*, 105323. [CrossRef]

29. Algamal, Z.Y.; Qasim, M.K.; Lee, M.H.; Ali, H. Improving grasshopper optimization algorithm for hyperparameters estimation and feature selection in support vector regression. *Chemom. Intell. Lab. Syst.* **2021**, *208*, 104196. [CrossRef]
30. Mafarja, M.; Aljarah, I.; Heidari, A.A.; Hammouri, A.I.; Faris, H.; Al-Zoubi, A.; Mirjalili, S. Evolutionary Population Dynamics and Grasshopper Optimization Approaches for Feature Selection Problems. *Knowl.-Based Syst.* **2017**, *145*, 25–45. [CrossRef]
31. Zhang, W.; Zhang, S.; Wu, F.; Wang, Y. Path Planning of UAV based on Improved Adaptive Grey Wolf Optimization Algorithm. *IEEE Access* **2021**, *9*, 89400–89411. [CrossRef]
32. Wang, J.; Zhang, Y.; Hong, M.; He, H.; Huang, S. A self-adaptive level-based learning artificial bee colony algorithm for feature selection on high-dimensional classification. *Soft Comput.* **2022**, *26*, 9665–9687. [CrossRef]
33. Jin, H.C.; Lee, J.; Park, C. Deep-space trajectory optimizations using differential evolution with self-learning. *Acta Astronaut.* **2022**, *191*, 258–269.
34. Li, X.; Wang, L.; Jiang, Q.; Li, N. Differential evolution algorithm with multi-population cooperation and multi-strategy integration. *Neurocomputing* **2021**, *421*, 285–302. [CrossRef]
35. Pan, J.S.; Liu, N.; Chu, S.C. A competitive mechanism based multi-objective differential evolution algorithm and its application in feature selection. *Knowl.-Based Syst.* **2022**, *245*, 108582.
36. Gao, D.; Wang, G.G.; Pedrycz, W. Solving Fuzzy Job-Shop Scheduling Problem Using DE Algorithm Improved by a Selection Mechanism. *IEEE Trans. Fuzzy Syst.* **2020**, *28*, 3265–3275. [CrossRef]
37. Li, H.; He, F.; Pan, Y. Multi-objective dynamic distribution adaptation with instance reweighting for transfer feature learning. *Knowl.-Based Syst.* **2023**, *263*, 110303. [CrossRef]
38. Jamil, S.; Rahman, M.; Tanveer, J.; Haider, A. Energy Efficiency and Throughput Maximization Using Millimeter Waves and ndash;Microwaves HetNets. *Electronics* **2022**, *11*, 474. [CrossRef]
39. Nyiam, P.B.; Salhi, A. A Comparison of Benson's Outer Approximation Algorithm with an Extended Version of Multiobjective Simplex Algorithm. *Adv. Oper. Res.* **2021**, *2021*, 1857030. [CrossRef]
40. Wambua, A.W.; Wambugu, G.M. A Comparative Analysis of Bat and Genetic Algorithms for Test Case Prioritization in Regression Testing. *Int. J. Intell. Syst. Appl.* **2023**, *14*, 13. [CrossRef]
41. Mirjalili, S.; Lewis, A. S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization. *Swarm Evol. Comput.* **2013**, *9*, 1–14. [CrossRef]
42. Qiang, Y.; Chen, W.N.; Deng, J.D.; Yun, L.; Zhang, J. A Level-Based Learning Swarm Optimizer for Large-Scale Optimization. *IEEE Trans. Evol. Comput.* **2018**, *22*, 578–594.
43. Wang, Y.J.; Zhang, J.S. Global optimization by an improved differential evolutionary algorithm. *Appl. Math. Comput.* **2007**, *188*, 669–680.
44. Li, C.; Deng, L.; Qiao, L.; Zhang, L. An efficient differential evolution algorithm based on orthogonal learning and elites local search mechanisms for numerical optimization. *Knowl. Based Syst.* **2022**, *235*, 107636. [CrossRef]
45. Bo, L.; Ding, L.; Li, J. Differential evolution-based parameters optimisation and feature selection for support vector machine. *Int. J. Comput. Sci. Eng.* **2016**, *13*, 355.
46. Zhan, Z.H.; Wang, Z.J.; Jin, H.; Zhang, J. Adaptive Distributed Differential Evolution. *IEEE Trans. Cybern.* **2019**, *77*, 1–15. [CrossRef]
47. Deng, J.; Hu, P.; Bai, Z.J.; Wang, C.P.; Kang, F.R.; Liu, L. Dynamic behaviours on oxidation heat release of key active groups for coal with different degrees of metamorphism. *Fuel* **2022**, *320*, 123967. [CrossRef]
48. Zl, A.; Qz, B. Variable metric evolution strategies by mutation matrix adaptation - ScienceDirect. *Inf. Sci.* **2020**, *541*, 136–151.
49. Zt, A.; Kl, A.; Yi, W.B. Differential evolution with adaptive mutation strategy based on fitness landscape analysis. *Inf. Sci.* **2021**, *549*, 142–163.
50. Rauf, H.T.; Bangyal, W.; Pervaiz, S.; Ahmad, J. An Improved Particle Swarm Optimization Algorithm with Chi-Square Mutation Strategy. *Int. J. Adv. Comput. Sci. Appl.* **2019**, *10*, 481–491.
51. Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.