Electronic Research Archive

https://www.aimspress.com/journal/era

ERA, 32(5): 3588–3617.
DOI: 10.3934/era.2024165
Received: 04 April 2024
Revised: 24 May 2024
Accepted: 30 May 2024
Published: 03 June 2024

Research article

Feature selection via a multi-swarm salp swarm algorithm

Bo Wei[1,2], Xiao Jin[1,*], Li Deng[3,*], Yanrong Huang[4] and Hongrun Wu[5]

[1] School of Computer Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China
[2] Longgang Research Institute, Zhejiang Sci-Tech University, Longgang 325000, China
[3] School of Science, Zhejiang Sci-Tech University, Hangzhou 310018, China
[4] College of Economics and Management, Zhejiang University of Water Resource and Electric Power, Hangzhou 310018, China
[5] College of Physics and Information Engineering, Minnan Normal University, Zhangzhou 363000, China

* **Correspondence:** Email: 202130504101@mails.zstu.edu.cn, lideng75@zstu.edu.cn.

Abstract: Feature selection (FS) is a promising pre-processing step before performing most data engineering tasks. The goal of it is to select the optimal feature subset with promising quality from the original high-dimension feature space. The Salp Swarm Algorithm (SSA) has been widely used as the optimizer for FS problems. However, with the increase of dimensionality of original feature sets, the FS problems propose significant challenges for SSA. To solve these issues that SSA is easy to fall into local optimum and have poor convergence performance, we propose a multi-swarm SSA (MSSA) to solve the FS problem. In MSSA, the salp swarm was divided into three sub-swarms, the followers updated their positions according to the optimal leader of the corresponding sub-swarm. The design of multi-swarm and multi-exemplar were beneficial to maintain the swarm diversity. Moreover, the updating models of leaders and followers were modified. The salps learn from their personal historical best positions, which significantly improves the exploration ability of the swarm. In addition, an adaptive perturbation strategy (APS) was proposed to improve the exploitation ability of MSSA. When the swarm stagnates, APS will perform the opposition-based learning with the lens imaging principle and the simulated binary crossover strategy to search for promising solutions. We evaluated the performance of MSSA by comparing it with 14 representative swarm intelligence algorithms on 10 well-known UCI datasets. The experimental results showed that the MSSA can obtain higher convergence accuracy with a smaller feature subset.

Keywords: swarm intelligence; feature selection; wrapper; salp swarm algorithm; artificial intelligence

1. Introduction

With the development of 5G, big data, blockchain, and other technologies, data volume in many fields shows a trend of geometric exponential growth. These data are often high-dimensional datasets with hundreds or even thousands of features, some of which are irrelevant, redundant, or noisy [1]. These features not only affect the performance of the learning algorithm [2], but also greatly increase the calculation cost. Feature selection (FS) aims to remove such irrelevant and redundant features from the original feature set while selecting the most effective ones. It plays an important role in improving algorithm performance and reducing computing costs. Therefore, it has been successfully used in medical analysis [3, 4], diagnosis of cancer data [5, 6], drug activity prediction [7, 8], fault diagnosis [9–11], and so on.

According to evaluation criteria, FS methods are commonly divided into filter-based and wrapper-based approaches. The filter-based approach usually selects the optimal feature subset based on four metrics (distance, information, consistency, and dependency) during the evaluation process. It ranks features by computing the correlation between features and classes and the redundancy between features. Since the filter-based approach only needs to process datasets without employing a learning algorithm, it has high computational efficiency. The wrapper-based approach combines a search strategy with a learning algorithm for solving FS tasks. Compared with filter-based methods, wrapper-based methods can usually obtain better feature subsets with promising quality in the same tasks. However, wrapper-based approaches have the disadvantages of high time complexity and poor generalization ability. Therefore, many scholars have proposed hybrid approaches with combining filter-based and wrapper-based approaches.

To find the optimal feature subset, FS is treated as an NP-hard discrete optimization problem [12]. Therefore, it is impractical for high-dimensional datasets to find the optimal solution from all possible feature subsets by using a similar exhaustive search strategy. Assuming that a dataset contains N features, then 2^N feature subsets need to be generated and evaluated to obtaining the optimal feature subset finally. As a result, traditional FS methods are no longer applicable to most data engineering tasks.

The swarm intelligence (SI) algorithm is inspired by simulating the interaction of certain groups of organisms in nature [13–16]. By simulating the characteristics of biological foraging, the SI algorithm can obtain better approximate solutions by conducting the strong random search in the search space. In view of this, more and more swarm-based optimization algorithms have been proposed, such as particle swarm optimization (PSO) [17–19], ant colony optimization (ACO) [20, 21], whale optimization algorithm (WOA) [22], grey wolf optimization algorithm (GWO) [23], firefly algorithm (FA) [24], artificial bee colony algorithm (ABC) [25], and so on. As FS is an NP-hard discrete optimization problem, it is difficult to obtain the optimal solution. Considering the strong random search ability of swarm-based optimization algorithms, they have been employed to solve the FS problem successfully.

Inspired by the group behavior of salps in the ocean, the swarm-based Salp Swarm Algorithm (SSA) is proposed by Mirjalili [26]. The salps feed on phytoplankton in the water and move by inhaling and spewing seawater. Due to the advantages of concise implementation and fast convergence in most cases, SSA has been widely applied to solve feature selection problems in recent years. However, the existing algorithms based on SSA have problems such as premature convergence and low search efficiency.

To address the issue mentioned above, we propose a multi-swarm SSA (MSSA) to solve the FS problem. The main contributions of this work are summarized as follows:

1) The salp swarm is divided into three sub-swarms, aiming at maintaining the swarm diversity. When one sub-swarm falls into the local optimum, it will not affect the other two sub-swarms, thus maintaining the diversity of the whole swarm.

2) The updating models of leaders and followers are modified to improve the exploration ability of the MSSA. Based on three sub-swarms, the followers move towards the current optimal leader of the corresponding sub-swarm, so that salps of the swarm have different learning exemplars. In this way, MSSA can avoid premature convergence effectively.

3) An adaptive perturbation strategy (APS) is proposed to enhance the exploitation ability of MSSA. In APS, an adaptive parameter *per* is designed to adjust the implementing frequency of the lens-OBL strategy and the SBX strategy during swarm evolution, which can help the swarm escape from the local optimum.

The remainder of this paper is structured as follows: Section 2 briefly reviews related work on SI algorithms and SSAs. The proposed MSSA is elaborated in Section 3. Section 4 shows the experimental setup. Section 5 provides experimental results and analysis. The final section concludes the study.

2. Related work

2.1. Classical SI algorithm

To solve optimization problems in the practical application fields, e.g., FS, many swarm-based algorithms have been proposed one after another. They include gravitational search algorithm (GSA) [27], bat algorithm (BA) [28], gray wolf optimization (GWO) algorithm [23, 29], SSA [26], and so on. Especially, some of these algorithms have been used as search strategies for wrapper-based approaches.

However, these optimization algorithms may have problems such as premature convergence, falling into local optimization solutions, and too many parameters needed in the evolution process. Therefore, many scholars have improved these optimization algorithms. For example, Feng Wang et al. introduced a reinforcement learning strategy into PSO, which combines with the level competition mechanism to improve the search efficiency and convergence ability of the algorithm [30]. Ke Chen et al. proposed CUS-PSO based on the surrogate model [31], which uses the correlation among features to generate candidate particles, and then approximates their fitness values. Ghosh et al. proposed a hybrid algorithm based on ACO [32], which combines the wrapper-based approach with the filter-based approach. The hybrid algorithm can reduce the computational complexity by employing a filter for evaluating the candidate feature subsets. Yong Zhang et al. presented the TMABC-FS algorithm [33], which uses different search strategies for different roles of bees. In addition, many swarm-based approaches haven't been listed due to the limited space.

2.2. Canonical SSA

Inspired by the group behavior of salps in the ocean, Mirjalili et al. presented the salp swarm algorithm (SSA) in recent years [26]. In SSA, the swarm of salps moves and forages in the form of a chain structure, and the roles of salps consist of leader and follower. The individual at the front of the salp chain acts as the leader, while the rest of the salps act as followers. The leader leads the

movement direction of the whole salp chain, while the followers follow the previous leader. In this way, the leader searches for food sources, and followers move towards the leader gradually. This moving pattern enables the salp chain to have strong global exploration and local exploitation abilities. Like other swarm-based algorithms, the position of a salp is defined as a d-dimensional vector, where d represents the dimension number of an optimization problem. Then, the swarm of salps can be defined as a $N \times d$ matrix, where N represents the size of the swarm. The matrix is shown in Eq (2.1).

$$X = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_d^1 \\ x_1^2 & x_2^2 & \cdots & x_d^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^n & x_2^n & \cdots & x_d^n \end{bmatrix} \tag{2.1}$$

The leader updates its position according to Eq (2.2).

$$x_j^1 = \begin{cases} F_j + c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) & if\, c_3 \geq 0.5 \\ F_j - c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) & else \end{cases} \tag{2.2}$$

where x_j^1 represents the jth dimension of position vector denoting the first salp, which is the leader of salps. F is the position of the food source. c_1 is an important parameter defined as Eq (2.3). It has the function of balancing exploration ability and exploitation ability of SSA. c_2 and c_3, which determine the step size and movement direction of the leader, respectively, are two random numbers in the range of [0, 1]. ub_j and lb_j represent the upper boundary and the lower boundary of the jth dimension, respectively.

$$c_1 = 2e^{-\left(\frac{4l}{L}\right)^2} \tag{2.3}$$

where l is the current number of iterations and L is the maximum number of iterations.

A follower updates its position according to Eq (2.4).

$$x_j^i = \frac{1}{2}\left(x_j^i + x_j^{i-1}\right) \tag{2.4}$$

where x_j^i represents the jth dimension of position vector denoting ith follower salp.

2.3. Binary SSA (BSSA)

The SSA was first proposed to solve the continuous optimization problem. Since FS is a discrete optimization problem, SSA cannot effectively deal with it. To address the issue, binary SSA (BSSA) was presented in recent years. In BSSA, the components of position vectors need to be mapped into 0 or 1 after each iteration. The mapping model of position vectors is shown in Eqs (2.5) and (2.6).

$$S\left(x_j^i\right) = \frac{1}{1 + \exp^{-x_j^i}} \tag{2.5}$$

$$newx_j^i = \begin{cases} 1 & if\, rand \geq S\left(x_j^i\right) \\ 0 & else \end{cases} \tag{2.6}$$

where x_j^i represents the jth dimension of position vector denoting the ith salp; *rand* is a random number uniformly distributed in [0, 1]; sigmoid function S represents the probability of selecting a candidate feature; $newx_j^i$ is the jth dimension of position vector denoting ith salp.

2.4. The variants of SSA

Since it was proposed in 2017, SSA has attracted a lot of attention from many scholars. Mechanisms were introduced to improve the performance of SSA. According to different improvement directions, these modifications can generally be divided into three situations: parameter adjustment, learning model adjustment, and hybrid strategy.

1) Parameter Adjustment: From the beginning, SSA stood out from many swarm-based algorithms with fewer parameters and better performance. Therefore, parameters optimization of SSA has also become a core content, aiming at enhancing its performance. Rohit et al. proposed an adaptive SSA (ASSA) with time-varying parameters [34]. In ASSA, the parameter c_1 is adaptively adjusted according to the current number of iterations. This strategy can effectively control the process of the exploration and exploitation. Furthermore, considering that the search process of SSA is nonlinear and complex, Aljarah et al. assigned a variety of nonlinear parameter combinations to SSA, aiming at maintaining the diversity of swarm and bringing different search behaviors of salps [35]. In addition, c_1 and c_2 are important parameters controlling the movement status of salps. Therefore, many scholars introduced chaotic mapping to adjust the two parameters, aiming at improving its global search ability and robustness. Sayed et al. proposed a novel chaotic salp swarm algorithm (CSSA) [36]. In CSSA, the chaotic map is employed to replace parameter c_2 with chaotic variables. This design introduces the chaotic map into the updating strategy of the swarm, which improves the performance and convergence speed of the algorithm. An improved salp swarm algorithm was proposed by Tawhid et al.. The algorithm introduces tent mapping into the update strategy of the leader, and experiments show that tent mapping can effectively improve the performance of the algorithm [37].

2) Learning Model Adjustment: In fact, it is undoubtedly that SSA can achieve better results commonly when solving continuous optimization problems. However, for some complex optimization problems, SSA may can not solve these problems well because the canonical updating models of salps are not efficient enough. Therefore, the updating models of leader and follower were modified according to the actual situation. For example, Choura et al. proposed Chaotic SSA and Dynamic SSA by introducing multiple chaotic maps and adjusting the update method of the follower, respectively [38]. Hegazy et al. introduced inertia weights based SSA to modify the updating model of the leader and follower. Experiments show that the proposed algorithm can obtain higher convergence accuracy [39]. Moreover, various adjustments about the topology have been made in SSAs because the neighborhood topology determines the learning mechanism. In this way, SSA can select learning exemplars flexibly and then enhance its performance. In the literature [40], an elite pool selection strategy was proposed by Tang et al.. In this strategy, an individual in the elite pool is randomly selected and used as a food source. This strategy enhances the exploration ability of the leader and enriches the swarm diversity. Liu et al. proposed a global search-oriented adaptive leader salp swarm algorithm [41]. In the leader position update formula, the position of the previous generation of the salp group is introduced as a learning exemplar, which improves the sufficiency of global search and effectively avoids the algorithm from falling into the local extremum.

3) Hybrid Strategy: Since the integration of multiple strategies can combine the advantages of

different strategies, it is a common way of improving the performance of algorithms. In particular, various local search strategies were integrated into SSAs, which can achieve a good balance between convergence speed and the diversity of the swarm. Zhang et al. proposed an enhanced SSA (ESSA) [42]. In ESSA, various strategies such as orthogonal learning, quadratic interpolation, and generalized oppositional learning are combined with SSA to improve its global exploration and local exploitation performance. Yavuz proposed an SSA with a refreshing-gap strategy (DSSA-R) based on a diversified position update equation [43]. Instead of using a single leader strategy, DSSA-R accepts half of the swarm as leaders. In addition, the refreshing-gap strategy and linear population reduction strategy are introduced into DSSA-R to strengthen the performance of the DSSA-R. Furthermore, the hybrid of different algorithms is also a research direction in SSAs for obtaining better solutions. For example, SSA was combined with PSO to make the exploration and exploitation abilities more efficient [44]. In this approach, the position of each salp is updated according to one of the PSO and SSA updating models randomly. In the literature [45], SSA was integrated with GWO, where the leader and the followers are updated according to SSA and GWO, respectively. This hybrid design can improve the exploration and exploitation abilities to a large extent. Given the advantages of the hybrid strategy, our work has done relevant research on this basis.

3. MSSA

In standard SSA, the leader is responsible for searching in the neighborhood of the food source. Furthermore, the followers update their positions according to their previous salps. However, the updating model of followers may make them move closer to the leader quickly. Once the leader cannot find a better food source, the diversity of the swarm will decline rapidly, and then the SSA will prematurely converge. Considering this problem, the salp swarm of the proposed MSSA is divided into three sub-swarms. In this way, the followers of each sub-swarm follow the current optimal leader of the corresponding sub-swarm. When one sub-swarm falls into local optimum, it will not affect the other two sub-swarms, thus maintaining the diversity of the whole swarm.

Furthermore, the setting for the number of leaders also affects the exploitation ability of SSAs. In canonical SSA, only one leader searches the neighborhood region of the food source. The rest of salps are regarded as followers who are used for exploring the search space. When solving high-dimensional problems with many local optima solutions, this setting may lead to low performance of SSAs. In this work, the optimal number of leaders is obtained through experimental verification. The MSSA consists of the following subsections.

3.1. Swarm initialization based on cubic chaotic mapping strategy

Most swarm-based approaches contain random parameters, which are usually generated from Uniform and Gaussian distributions. However, more and more swarm-based approaches were proposed based on chaos theory, which can achieve better accuracy, convergence speed , and stability [36]. Considering its advantages of non-linearity, ergodicity, and randomness, a chaotic mapping strategy was employed for initialization of the swarm. In particular, this strategy made the approaches obtain better diversity of the swarm in the previous research works [46, 47]. Inspired by these researches, the cubic chaotic mapping strategy is used for initializing the swarm of salps, aiming at obtaining a diversity swarm in our work. The expression of the cubic chaotic mapping strategy is shown in Eq (3.1).

$$y_{i+1,j} = \rho y_{i,j}\left(1 - y_{i.j}^2\right) \tag{3.1}$$

where $y_{i,j} \in (0,1)$; ρ is a control parameter and its value is in the range of (1.5, 3). The chaotic sequence is mapped into decision space and then converted into the position vector of a salp. The conversion equation is shown in Eq (3.2).

$$x_j^i = lb_j + y_j^i\left(ub_j - lb_j\right) \tag{3.2}$$

where x_j^i is the value of jth dimension of position vector denoting ith salp; ub_j and lb_j represent the upper boundary and the lower boundary of the jth dimension, respectively.

The process of swarm initialization is shown in Algorithm 1.

Algorithm 1: Pseudo-code for swarm initialization

Input: The size of a swarm N, decision boundary *bounds*, dimension d
Output: Initialized swarm *pop*

```
Randomly initialize the first salp of the swarm
for (i = 2 : N) do
    for (j = 1 : d) do
        Generate chaotic sequence according to Eq (3.1)
    end
end
Transform the chaotic sequence into the decision space according to Eq (3.2)
return pop
```

3.2. Multi-swarm updating strategy

In the original SSA, the updating formula of the leader is controlled by three parameters, i.e., c_1, c_2, and c_3. c_1 decreases nonlinearly with the iteration of SSA. c_2 and c_3 are random numbers between [0, 1]. This updating strategy is very random, and it does not take into account the informations carried by other salps in the swarm so that the SSA is easy to fall into a local optimum.

To address this issue, a new updating model of the leader is proposed in MSSA. In MSSA, the parameter c_1 takes into account the position informations of two random salps for controlling the movement direction of a salp. Like the canonical SSA, however, c_2 and c_3 are also random numbers between [0, 1]. c_2 is employed to control the stepsize of the movement. c_3 controls whether the best information in personal history is used by salps. The updating model of the leader is shown in Eq (3.3).

$$x_j^i = \begin{cases} gbest_j + c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) & if\, c_3 \geq 0.5 \\ pbest_j^i & else \end{cases} \tag{3.3}$$

where ub_j and lb_j are the upper boundary and lower boundary of the jth dimension in the decision space, respectively; $pbest_j^i$ is the historical personal optimal position of the ith leader; *gbest* is the current selected food source for the swarm. x_j^i is the jth dimension of position vector denoting the ith salp. The updating formula of c_1 is shown in Eq (3.4).

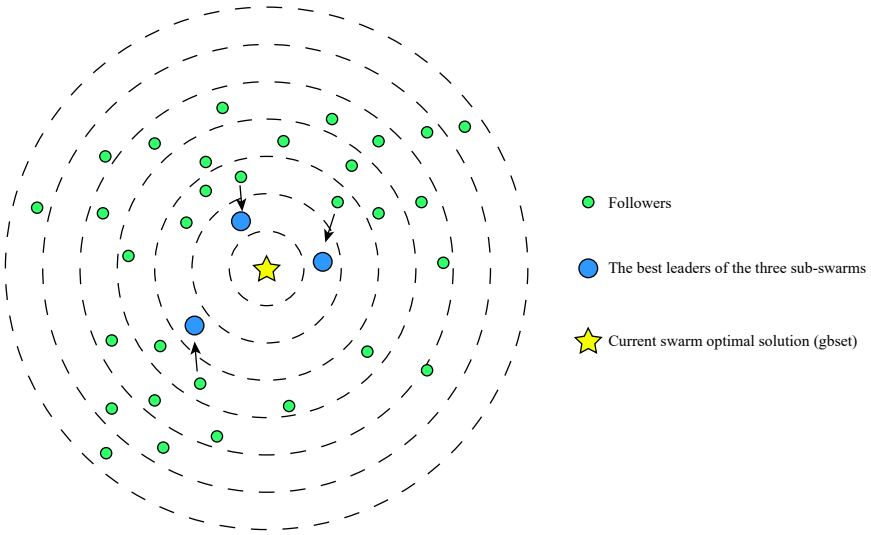


Figure 1. Illustration of followers moving towards the best leader in the same sub-swarm.

$$c_1 = e^{-(t/T)^2} \times \left(x_j^{r1} - x_j^{r2}\right) \tag{3.4}$$

where t is the current number of iterations, and T is the maximum number of iterations. x_j^{r1} and x_j^{r2} represent two salps selected randomly from the swarm. It can be seen that c_1 utilizes the informations of two salps. Compared with changing randomly, this setting mode of c_1 can improve the convergence efficiency of MSSA.

Furthermore, the swarm of MSSA is divided into three sub-swarms. The followers move towards the current optimal leader of the corresponding sub-swarm, so that salps of the swarm have different learning exemplars. In this way, the swarm has better diversity (shown in Figure 1). The updating model of followers is shown in Eq (3.5).

$$x_j^i = \begin{cases} \frac{1}{2}\left(x_j^{i-1} + x_j^i\right) & if\ rand \geq 0.5 \\ pbest_j^i & else \end{cases} \tag{3.5}$$

where $pbest_j^i$ is the personal historical optimal position of the ith follower; x_j^i is jth dimension of the position vector of ith salp in the swarm.

The process of updating positions of salps is shown in Algorithm 2. In lines 3–10, the positions of all leaders are updated first according to the Eq (3.3). In lines 11 and 12, the optimal leader of each sub-swarm is selected. At last, followers are updated with reference to the position of the optimal leader in the corresponding sub-swarm.

3.3. Adaptive perturbation strategy (APS)

The opposite-based learning (OBL) was introduced into many swarm-based optimization algorithms, aiming at improving the performance of these algorithms [48, 49]. It can search for the inverse solution of the current solution to expand the search range rapidly, and then maintain the diversity of

Algorithm 2: Pseudo-code of updating positions of salps in the swarm

Input: current swarm pop, Number of leaders $leaderNum$, dimension d, the historical optimum of the swarm $pbest$, decision boundary $bounds$

Output: Updated swarm pop_{next}

```
for (i = 1 : leaderNum) do
    for (j = 1 : d) do
        Update c1 according to Eq (3.4)
        Take c2 and c3 as random numbers [0, 1]
        Update the leader's position according to Eq (3.3)
        Update the leader's position with the upper and lower boundaries
    end
end
Evaluate leaders in a swarm
Find the leaders with the highest fitness values in each of the three sub-swarms
for (i = leaderNum + 1 : N) do
    for (j = 1 : d) do
        Update the follower's position according to Eq (3.5)
        Update the follower's position with the upper and lower boundaries
    end
end
return pop_next ;
```

the swarm. These OBL-based algorithms can quickly obtain good candidate solutions in the early iterations, while they may fall into local optima in the late iterations. In this instance, the opposition-based learning with lens imaging principle (lens-OBL) was proposed in [50], which achieved better performance in maintaining swarm diversity. The formula for calculating the opposite solution through lens-OBL is shown in Eq (3.6). a and b are the maximum and minimum values of the positions in the current population, as shown in Eq (3.7).

$$x^* = \frac{a+b}{2} + \frac{a+b}{2k} - \frac{x}{k} \tag{3.6}$$

$$\begin{cases} a_j = max\left(X_j\right) \\ b_j = min\left(X_j\right) \end{cases} \tag{3.7}$$

where a_j and b_j are values of a and b in the jth dimension, respectively. When $k = 1$, the standard OBL formula can be obtained from the lens-OBL strategy. At this time, OBL is a special case of lens-OBL strategy. When extending to d dimensional space, assuming that the position vector of a salp is $X\left(x_1, x_2, \ldots, x_d\right)$, the reverse solution $X^*\left(x_1^*, x_2^*, \ldots, x_d^*\right)$ can be obtained according to Eq (3.6).

The simulated binary crossover (SBX) strategy is a real-coded recombination operator [51]. Since SBX strategy has strong local search ability, it is widely used in multi-objective evolutionary algorithms. To be specific, it uses two parents to produce two offsprings by blending the genes on their chromosomes one by one. The two offspring inherit the information of the parent and also mutate to a

certain extent. As an operator, it can effectively improve the search ability of the algorithm. Assuming there are two parents x_1 and x_2, their two offsprings, i.e., off_1 and off_2, can be generated by Eq (3.8).

$$\begin{cases} off_1 = 0.5 \times [(1+\beta) \times x_1 + (1-\beta) \times x_2] \\ off_2 = 0.5 \times [(1-\beta) \times x_1 + (1+\beta) \times x_2] \end{cases} \tag{3.8}$$

where β is determined dynamically by the distribution factor η , as shown in Eq (3.9).

$$\beta = \begin{cases} (rand \times 2)^{1/(1+\eta)} & if \;\; rand \leq 0.5 \\ \left[\frac{1}{(2-rand \times 2)}\right]^{1/(1+\eta)} & otherwise \end{cases} \tag{3.9}$$

where η is a constant representing the distribution index. When the value of η is larger, the offsprings will more similar to their parents. The value of η is commonly set to 10 or 20 [52, 53].

To improve the exploitation ability of the swarm, an adaptive perturbation strategy (APS) based on lens-OBL and SBX strategy is proposed in MSSA, as shown in Algorithm 3. In APS, an adaptive parameter *per* is designed to adjust the implementing frequency of the lens-OBL strategy and the SBX strategy during swarm evolution. Specifically, the SBX strategy is effectively used to promote communication between different sub-swarms, while the lens-OBL strategy is employed to enhance the local search ability of the swarm. In this way, the proposed APS can enhance the exploitation ability of MSSA effectively, which can also help MSSA escape from the local optimum. The parameter *per* is calculated as shown in the Eq (3.10).

$$per = \frac{0.3}{1 + e^{4-num}} \tag{3.10}$$

where *num* is used for calculating the number of stagnant iterations. After each iteration of swarm, the *num* is increased by 1 when the *gbest* does not improve; otherwise, *num* is reset to 0. The APS is executed multiple times, which may cause the population to search excessively in unpromising areas, Therefore, we limit the value of *per* to [0, 0.3]. With the increasing of the *num*, the value of *per* can increase from 0 to 0.3 when adopting the adaptive strategy. The more times of iteration stagnation, the greater the impact of the lens-OBL strategy and SBX strategies on the swarm.

3.4. The Framework of MSSA

Together with the aforementioned modules, MSSA can be described as in Algorithm 4. The flow chart of MSSA is shown in Figure 2.

3.5. Time complexity analysis

It can be seen from the framework of MSSA that the main computational complexity of the algorithm is mainly affected by the updating process and the local search process. Actually, the fitness evaluation is related to the real problem, so this process is not considered in the analysis. In this section, only the time complexity of the population update process and the local search process are analyzed.

In the process of updating the population position, the best leader needs to be found. Therefore, the time complexity of sorting $N/2$ leaders in the population is $O((Nlog(N/2))/2)$. Suppose the dimension of the search space is D, all N salps update their position vectors in D dimensions. Therefore,

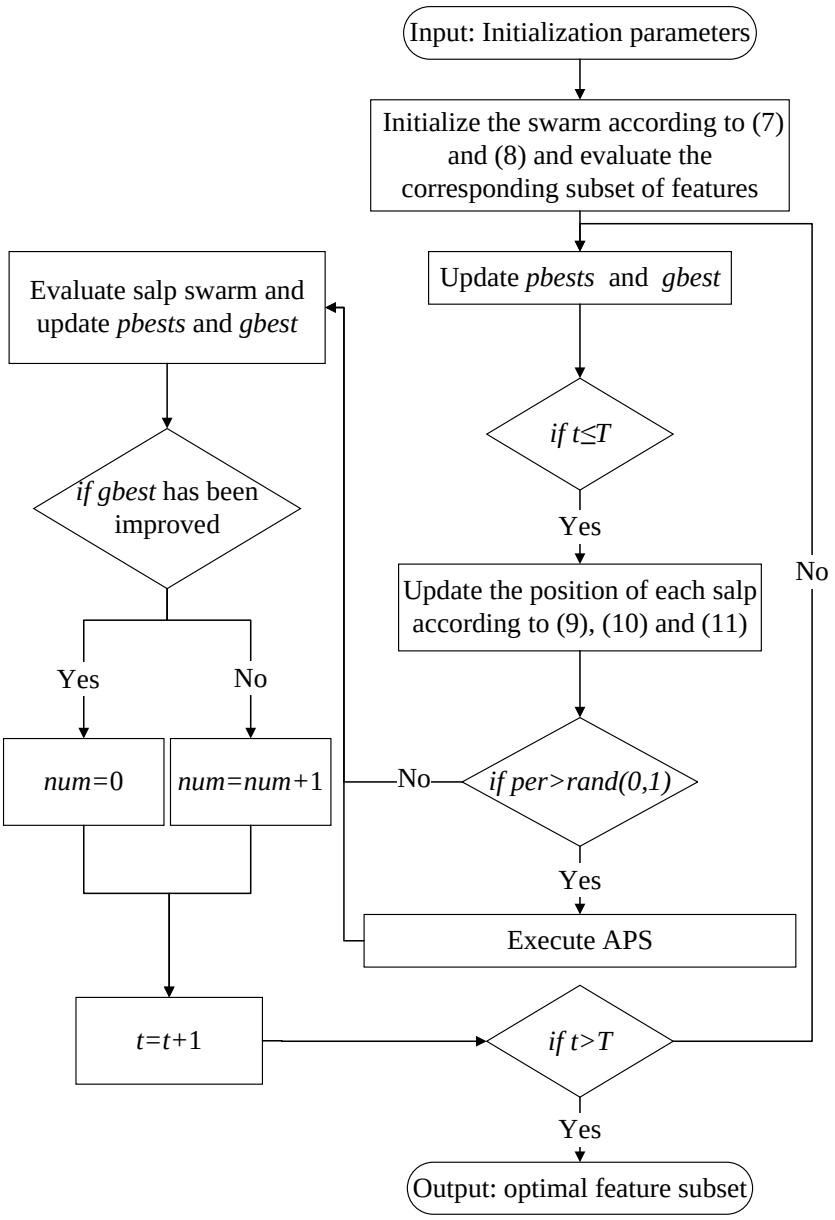


Figure 2. Flowchart of the proposed MSSA.

.

Algorithm 3: Pseudo-code of APS

Input: current swarm pop, decision boundary $bounds$, dimension d and scaling factor k
Output: updated swarm pop_{new}

```
if per > rand then
    Randomly select three leaders in each of the three sub-swarms and randomly select three
      dimensions
    Update the leader's position according to lens-OBL
    Update the leader's position with the upper and lower boundaries
    Evaluate the generated leader
    if f(X) < f(X_new) then
        X = X_new
    else
        Retain the original leader
    end
    Randomly choose two followers to perform the SBX operator using Eqs (3.8) and (3.9)
end
return pop_new ;
```

the time complexity of updating the positions of particles is $O(ND)$. In summary, the time complexity of the population update phase is $O(((Nlog(N/2))/2)) + ND)$. In the local search phase, the time complexity of executing the APS strategy is $O(ND/2)$.

Based on the above analysis, the time complexity of MSSA is $O(((Nlog(N/2))/2)) + ND)$ plus $O(ND/2)$, i.e., $O(Nlog(N) + ND)$. We can see that the time complexity of MSSA is slightly increased compared with the canonical SSA. It means that although MSSA introduces a new update model and adaptive perturbation strategy, its time complexity does not increase significantly, and this conclusion proves the effectiveness of MSSA.

4. Experimental setup

4.1. Datasets

To evaluate the performance of MSSA, three group experiments were conducted on 10 well-known UCI datasets that cover different domains and different attributes. Table 1 shows these datasets, including the number of classes, features, and data instances.

4.2. Setup of experimental

In this subsection, the parameters and the environment of experiments are briefly introduced. After a series of experiments in MSSA, the optimal values of the main parameters are obtained. The values of the main parameters of MSSA and comparison algorithms are shown in Table 2. The main parameters of MSSA are explained as follows: T represents the maximum number of iterations and is set to 100. $leaderNum$ represents the number of leaders in the swarm and is set to $N/2$. The per is an adaptive parameter, which controls the probability of executing the lens-OBL and the SBX strategy. N is the

Algorithm 4: Pseudo-code of the MSSA

Input: The maximum number of iterations T, decision boundary *bounds*, number of leaders $leaderNum$, scaling factor k and the swarm size is N. set $num = 0$, $t = 1$

Output: optimal subset of features

```
Randomly initialize the positions of the salp swarm between the upper and lower bounds by chaotic maps
Calculate the fitness of each salp
Set the best salp position to the gbest and pbests
while t ≤ T do
    ref = gbest
    Update per according to Eq (3.10)
    for (i = 1 : leaderNum) do
        for (j = 1 : d) do
            Update c1 according to Eq (3.4)
            Update the position of the leaders according to Eq (3.3)
            Update the leader's position with the upper and lower boundaries
        end
    end
    Evaluate leaders to get fitness values
    Select the three leaders with the best fitness in the three sub-swarms respectively
    for (i = leaderNum + 1 : N) do
        for (j = 1 : d) do
            Update the position of the follower according to the chosen leaders using Eq (3.5)
            The adaptive perturbation strategy is executed according to Algorithm 3
            Update the salps's position with the upper and lower boundaries
        end
    end
    Calculate the fitness of each salp
    Update gbest and pbests for each salp in the swarm
    t = t + 1
    If ref is equal to gbest, add num to 1, otherwise reset num to 0
end
```

size of a salp swarm and is set to 60. Consider that for larger values of η, the offspring will be more like their parents. In order to strengthen the local search ability of the population, η is set to 10 in the SBX strategy. k is the scaling factor in the lens-OBL strategy. It is worth noting that the threshold for decoding is set to 0.5 in this paper. If the value of a feature is greater than the threshold, the feature is added to the feature subset.

All experiments were implemented on MATLAB R2016a software. The computer is equipped with an Intel Core (TM) i7 CPU and the Windows 10 operating system. To reduce experimental statistical errors, for each dataset all the algorithms have been conducted 20 times. Furthermore, to better test the performance of the proposed MSSA, this paper also uses the Friedman test and Wilcoxon signed rank

Table 1. List of used datasets.

No.	Datasets	#features	#instances	#classes
1	Tic-tac-toe	9	958	2
2	WineEW	13	178	3
3	Zoo	16	101	6
4	CongressEW	16	435	2
5	Lymphography	18	148	2
6	IonosphereEW	34	351	2
7	KrvskpEW	36	3196	2
8	WaveformEW	40	5000	3
9	SonarEW	60	208	2
10	Clean1	166	476	2

test to analyze the experimental results.

4.3. Fitness function

For FS task in dataset classification problem, we usually need to focus on two goals. One is to have a higher classification accuracy, and the other is to select a smaller size of feature subset. Therefore, Eq (4.1) is used as the fitness function in our experiments.

$$Minf(x) = \alpha \times (1 - accuracy) + (1 - \alpha) \times \frac{l}{L} \tag{4.1}$$

where α is the weight factor that determines the relative importance of the classification accuracy, and α is set to 0.9 in this work. The *accuracy* denotes the classification accuracy of a candidate solution. l represents the number of features selected by the candidate solution. L represents the total number of features of the original feature set. In this way, we turn a two-objective problem into a minimal one. In our work, K Nearest Neighbor (KNN) classifier is used for evaluating candidate solutions. When the classification accuracies of two potential solutions are the same, the smaller candidate feature subset will be considered as a better one.

5. Experimental results and analysis

5.1. Analysis of parameters

The parameter k is the scaling factor of the lens-OBL strategy. The value of k affects the scope of lens-OBL strategy and then affects the accuracy of MSSA. Similarly, the parameter $leaderNum$ determines the number of leaders in the swarm. When $leaderNum$ takes a more appropriate value, it can effectively balance the exploration and exploitation abilities of MSSA. In this part, we analyze the

Table 2. Parameter settings of algorithms.

Algorithm	Year	Parameter settings
GA [55]	1989	$p_c = 0.9, p_m = 0.1$
BPSO [17]	1995	$w \in [0.9, 0.6], c_1 = c_2 = 2$
BGSA [27]	2010	$G_0 = 100, \alpha = 20$
BBA [28]	2014	$Q_{min} = 0, Q_{max} = 2, A = 0.5, r = 0.5$
BGWO [54]	2016	α is linearly decreased from 2 to 0
bGWO1 [54]	2016	α is linearly decreased from 2 to 0
bGWO2 [54]	2016	α is linearly decreased from 2 to 0
WOASA [58]	2017	α is linearly decreased from 2 to 0
BSSA [26]	2017	c_1 is nonlinear decreased from 2 to $2 \times e^{-16}$
TCSSA3 [35]	2018	c_1 is nonlinear decreased from 2.5 to 0.5
BSSA_S3_CP [13]	2018	c_1 is nonlinear decreased from 2 to $2 \times e^{-16}$
S-bBOA [57]	2019	-
HGSA [56]	2019	$G_0 = 10$
BHOA_S4_Cr2 [59]	2022	-
MSSA	-	$per \in [0, 0.3], k = 0.8, \eta = 10$

optimal value of k and *leaderNum* through experimental results. Eight representative UCI datasets, covering from low to high dimensions, are selected for experiments. The performance of experiments with different parameters is shown in Tables 3 and 4, respectively.

5.1.1. Analysis about parameter k

In this experiment, different values of k are designed to analyze its impact on the performance of MSSA. Through the analysis, if k is too small, it is easy to exceed the upper or lower boundaries when executing the lens-OBL strategy. When $k = 1$, the lens-OBL strategy becomes the standard OBL strategy. In summary, to make the value of k more representative, the value range of k is set to [0.6, 2] in these experiments. Based on this, comparative experiments were performed on eight datasets. As shown in Table 3, the average classification accuracies were obtained, with bold representing the optimal results.

It can be clearly seen from Table 3 that different values of k have the same accuracies on the Tic-tac-toe and Zoo datasets. For some low-dimensional datasets, like WineEW, CongressEW, and Lymphography, the average accuracies obtained by MSSA with a larger k value are higher. For high-dimensional datasets, such advantages are not obvious. From all experimental results, it can be seen

that MSSA has the best performance when k is set to 0.8. Therefore, the value of k is set to 0.8 in the subsequent experiments.

Table 3. Average classification accuracy (in %) based on different k values.

Dataset	k=0.6	k=0.7	k=0.75	k=0.8	k=0.9	k=1	k=2
Tic-tac-toe	81.63	81.63	81.63	81.63	81.63	81.63	81.63
WineEW	99.72	99.72	99.78	**99.83**	**99.83**	**99.83**	99.78
Zoo	99.01	99.01	99.01	99.01	99.01	99.01	99.01
CongressEW	96.64	96.63	96.61	**96.72**	96.67	96.67	96.67
Lymphography	87.77	87.64	**87.84**	**87.84**	87.77	**87.84**	87.77
IonosphereEW	96.68	96.61	96.61	**96.72**	96.57	96.55	96.65
SonarEW	97.16	97.07	97.21	**97.31**	97.19	97.07	97.12
Clean1	99.74	99.72	99.79	**99.86**	99.82	99.77	99.77

5.1.2. Analysis about the number of leaders

Table 4 shows the average classification accuracies of MSSA with different numbers of leaders (from $3 \times N/4$ to $N/5$). It can be seen that, for most datasets, MSSA has higher classification accuracies and yields the best results when the number of leaders is set to $N/2$. The reason for these results is that the structure with $N/2$ leaders effectively extended the exploitative searching pattern of MSSA. Furthermore, the diversity of the swarm was not reduced by the excessive number of leaders. Furthermore, based on the results of the Friedman nonparametric statistical test (F-test), MSSA with $N/2$ leaders results in more promising results.

Figure 3 shows the convergence processes of MSSA with different numbers of leaders on six datasets, where the horizontal axis is the number of iterations and the vertical axis is the classification accuracy. It can be seen from Figure 3 that MSSA with more leaders can often find better results in the early stages of iterations, while the diversities of the swarm are reduced greatly in the later stages of iterations. This is because excessive salps are easy to gather when there are too many leaders in the swarm. However, as shown in Figure 3, the exploration and exploitation abilities of MSSA are well balanced when the number of leaders is set to $N/2$. Therefore, this setting about the number of leaders can balance the exploration and exploitation abilities of the swarm successfully, so that MSSA can achieve higher performance.

To analyze the stability changes of MSSA with different numbers of leaders, the box diagram toolbox is employed in our work. The box diagram toolbox shows the discrete distribution of data in a relatively stable way, while it can not be affected by outliers. In this subsection, several poor experimental results are considered as outliers. Figure 4 shows the box diagrams of experimental results on six datasets, aiming at analyzing the stabilities of MSSA with different numbers of leaders.

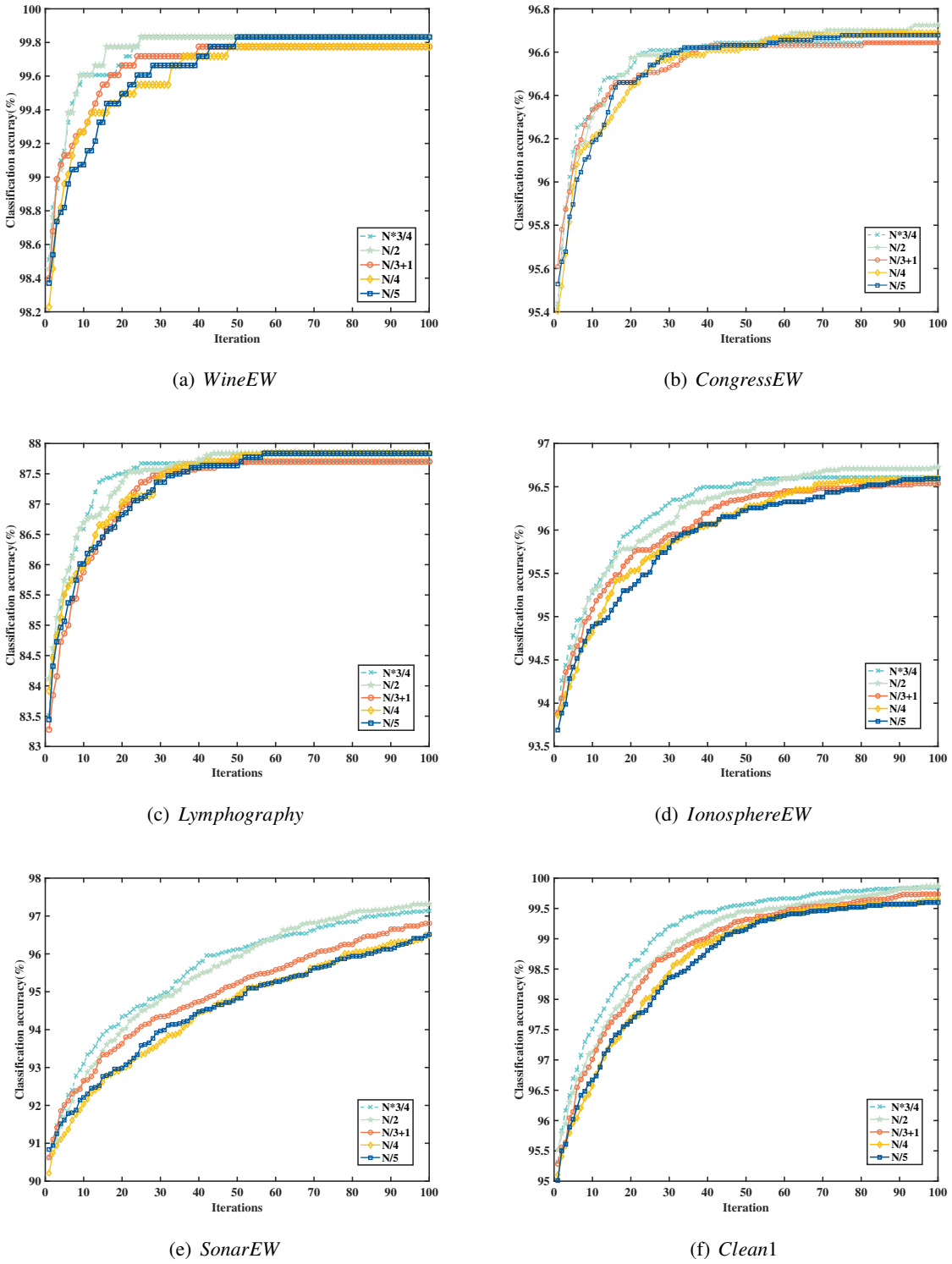
It can be seen from Figure 4 that, MSSA achieves the same experimental results on WineEW and Lymphography datasets. That is to say, the number of leaders has no significant effect on the stability

Table 4. Average classification accuracy (in %) based on different number of leaders.

Dataset	Metric	$3 \times N/4$	$N/2$	$N/3+1$	N/4	N/5
Tic-tac-toe	AVG	81.63	81.63	81.63	81.63	81.63
	STD	0	0	0	0	0
WineEW	AVG	**99.83**	**99.83**	99.78	99.78	**99.83**
	STD	0.4116	0.4116	0.4611	0.4611	0.4116
Zoo	AVG	99.01	99.01	99.01	99.01	99.01
	STD	0	0	0	0	0
CongressEW	AVG	96.64	**96.72**	96.64	96.69	96.68
	STD	0.1375	0.1021	0.1155	0.1375	0.1173
Lymphography	AVG	87.70	**87.84**	87.70	**87.84**	**87.84**
	STD	0.4159	0	0.4159	0	0
IonosphereEW	AVG	96.61	**96.72**	96.54	96.60	96.60
	STD	0.2046	0.2168	0.1673	0.1955	0.1955
SonarEW	AVG	97.14	**97.31**	96.80	96.49	96.51
	STD	0.5935	0.8015	0.6096	0.4705	0.6595
Clean1	AVG	99.84	**99.86**	99.74	99.64	99.60
	STD	0.1505	0.1707	0.2031	0.0988	0.0940
F-test		2	**1**	2.375	2.5	2.5

of MSSA on the two datasets, which is consistent with the average classification accuracies shown in Table 4. For dataset CongressEW, the experimental results are more concentrated when the number of leaders is set to $N/2$, even though different numbers of leaders do not lead to outliers. However, MSSAs with different numbers of leaders obtain some outliers on higher dimensional datasets (IonphosphereEW, SonarEW, and Clean1). This means that MSSA may obtain poor results even though it achieves the optimal solutions on these higher-dimensional datasets. In other words, MSSA has a higher classification accuracy and better stability to a large extent.

On the whole, when the number of leaders is set to $N/2$, MSSA does not get outliers on the CongressEW, IonosphereEW, SonarEW, and Clean1 datasets. For dataset Clean1, the experimental results obtained by MSSA with $N/2$ leaders are very similar to the optimal results. That is to say, MSSA with $N/2$ leaders can obtain higher classification accuracies and better stability on these datasets, which is consistent with the experimental results shown in Table 4. Based on the above experimental analysis,



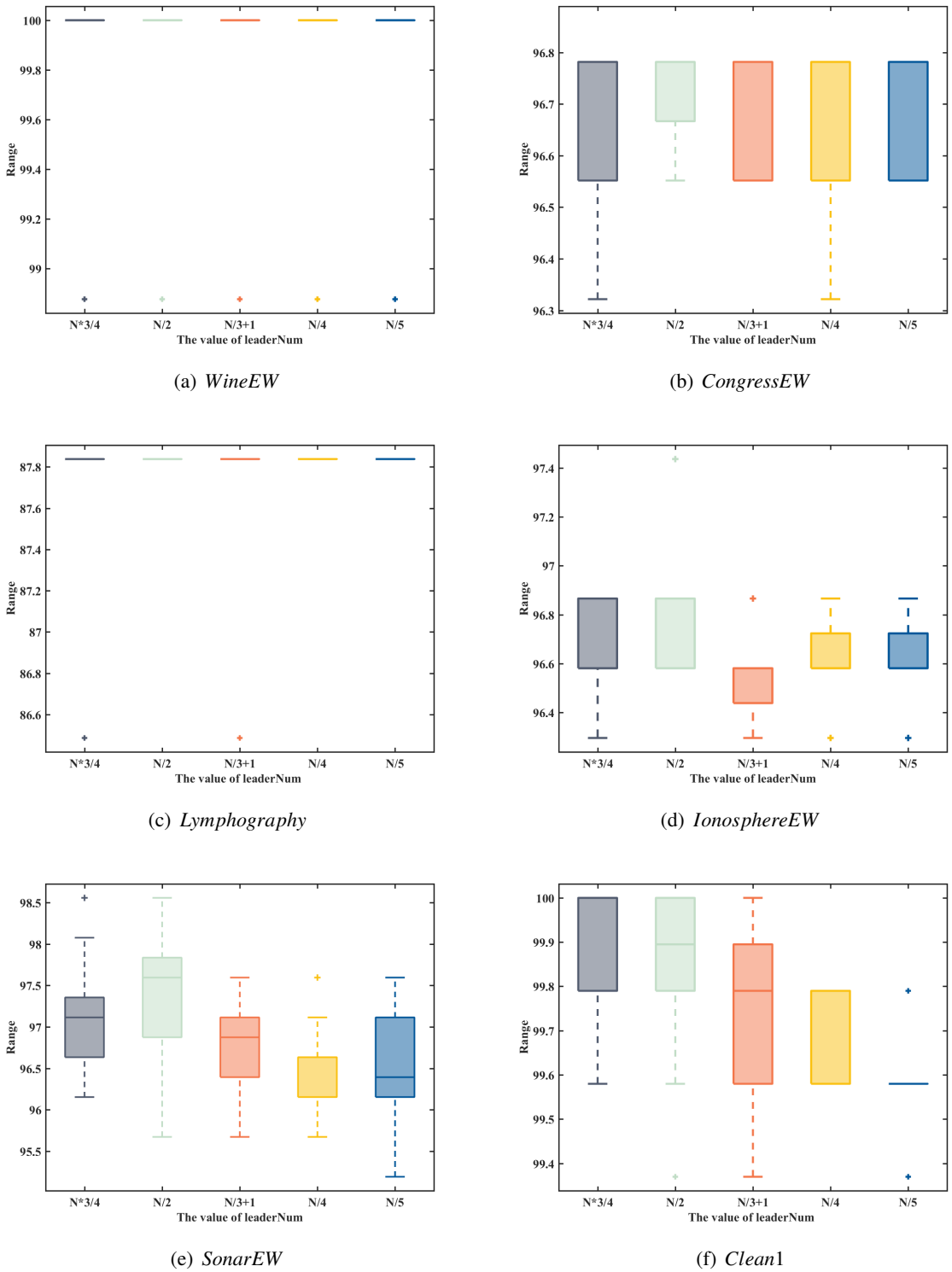
(a) *WineEW* (b) *CongressEW*

(c) *Lymphography* (d) *IonosphereEW*

(e) *SonarEW* (f) *Clean*1

Figure 3. Convergence curves for MSSA with different number of leaders for six datasets.

the number of leaders of MSSA is set to $N/2$ in our work.



(a) *WineEW* (b) *CongressEW*

(c) *Lymphography* (d) *IonosphereEW*

(e) *SonarEW* (f) *Clean*1

Figure 4. Box diagram of MSSA with different number of leaders in six datasets.

5.2. Comparison with representative algorithms

To validate the performance of MSSA, it is compared with five swarm-based algorithms in this subsection. The five approaches include Binary Gravitational Search Algorithm (BGSA) [27], Binary Grey Wolf Optimizer (BGWO) [54], Binary Bat Algorithm (BBA) [28], Genetic Algorithm (GA) [55], and BPSO [17]. The KNN classifier is employed for evaluating the fitness values of candidate solutions. From Table 5, it can be seen that MSSA achieves smaller fitness values among six competitors on 10 well-known UCI datasets. Especially, MSSA has a significant advantage in performance when solving the classification problem with high-dimensional datasets.

Table 5. The average fitness values of MSSA and other five swarm-based algorithms.

Dataset	BGSA	BGWO	BBA	GA	BPSO	MSSA
Tic-tac-toe	0.2878	0.2960	0.3564	0.2796	0.2982	**0.2209**
WineEW	0.1008	0.1026	0.1198	0.1935	0.1206	**0.0619**
Zoo	0.1057	0.0877	0.2086	0.0970	0.1232	**0.0589**
CongressEW	0.0862	0.0928	0.1544	0.0972	0.1364	**0.0607**
Lymphography	0.2480	0.2299	0.3121	0.2147	0.2536	**0.1597**
IonosphereEW	0.1522	0.1604	0.1505	0.1353	0.1642	**0.0577**
KrvskpEW	0.1382	0.1355	0.2070	0.1207	0.1431	**0.0810**
WaveformEW	0.3246	0.3287	0.3393	0.3531	0.3422	**0.2287**
SonarEW	0.1513	0.2084	0.2816	0.1630	0.1826	**0.0700**
Clean1	0.1421	0.1561	0.1952	0.1768	0.1938	**0.0203**

Table 6 shows the average classification accuracies obtained by MSSA and these five methods. It can be seen that MSSA achieves the highest classification accuracies among these six approaches on 10 UCI datasets. In particular, MSSA achieves classification accuracies of over 96% on seven out of ten datasets. Furthermore, BGWO and GA obtain suboptimal performance among these six approaches on most of the datasets. In particular, BGWO can obtain better results in low-dimensional datasets, while GA is superior in high-dimensional datasets. Besides, it can be seen from Table 6 that, the higher the dimension of the dataset is, the more significant the performance of MSSA has.

On the other hand, in order to better verify the performance of the proposed algorithm, we also conducted a statistical analysis of the experimental results. According to the results of the Friedman test and Wilcoxon signed rank test in Tables 6 and 7. As shown in Table 6, MSSA outperforms the comparison algorithms on all 10 datasets, so MSSA achieves the highest ranking in the Friedman test, which verifies that MSSA has superior performance. As can be seen from Table 7, the null hypothesis is rejected by all six algorithms. This shows that MSSA is significantly better than the comparison algorithm at the significance level of 0.05.

Furthermore, to further validate the performance of MSSA, we compare it with other six swarm-based algorithms (i.e., HGSA [56], S-bBOA [57], bGWO1 [54], bGWO2 [54], WOASA [58], and

Table 6. Comparison between MSSA and other five swarm-based algorithms based on average classification accuracy (in %).

Dataset	BGSA	BGWO	BBA	GA	BPSO	MSSA
Tic-tac-toe	75.26	75.38	66.54	76.09	75.18	**81.63**
WineEW	95.09	95.96	91.87	95.36	95.21	**99.83**
Zoo	93.92	97.45	87.39	92.94	94.51	**99.01**
CongressEW	95.12	94.76	87.17	94.13	92.35	**96.72**
Lymphography	78.11	81.31	70.14	81.64	79.06	**87.84**
IonosphereEW	88.13	88.47	87.65	89.38	88.03	**96.72**
KrvskpEW	93.39	90.81	81.64	92.15	92.00	**98.34**
WaveformEW	69.46	72.27	66.93	69.21	71.92	**80.03**
SonarEW	88.75	83.56	84.39	87.50	86.67	**97.31**
Clean1	89.82	90.77	82.65	86.97	85.49	**99.86**
$W \mid T \mid L$	10 | 0 | 0	10 | 0 | 0	10 | 0 | 0	10 | 0 | 0	10 | 0 | 0	-
F-test	3.5	3.1	5.9	3.3	4.2	**1**

Table 7. The results obtained by MSSA and competitors on Wilcoxon signed rank test (α = 0.05).

Algorithms	BGSA	BGWO	BBA	GA	BPSO
R+	55	55	55	55	55
R-	0	0	0	0	0
p	0.005	0.005	0.005	0.005	0.005
Assuming	reject	reject	reject	reject	reject
Select	MSSA	MSSA	MSSA	MSSA	MSSA

BHOA_S4_Cr2 [59]) on these 10 UCI datasets. The average classification accuracies obtained by these methods are shown in Table 8.

From Table 8, it can be seen that MSSA achieves the highest classification accuracies among seven competitors on six out of ten datasets. Furthermore, BHOA_S4_Cr2 obtains the highest classification accuracies on low-dimensional datasets (WineEW and Zoo). WOASA and HGSA achieve the best results on CongressEW and Lymphography datasets, respectively. In particular, MSSA obtains the

highest classification accuracies on Tic-tac-toe, IonosphereEW, KrvskpEW, WaveformEW, SonarEW, and Clean1 datasets. Therefore, the superior performance of MSSA is verified again by experimental results. Through analysis, it is found that MSSA can achieve a good balance between exploration and exploitation abilities when solving the classification problem on high-dimensional datasets.

Table 8. Comparison between the MSSA and other six swarm-based algorithms based on average classification accuracy (in %).

Dataset	HGSA	S-bBOA	bGWO1	bGWO2	WOASA	BHOA_S4_Cr2	MSSA
Tic-tac-toe	78.80	79.83	72.80	72.70	79.00	80.38	**81.63**
WineEW	98.90	91.90	93.00	92.00	95.90	**99.94**	99.83
Zoo	95.80	87.40	87.90	87.90	98.00	**100.00**	99.01
CongressEW	96.60	95.93	93.50	93.80	**98.00**	97.68	96.72
Lymphography	**89.20**	86.76	74.40	70.00	89.00	86.83	87.84
IonosphereEW	93.40	90.70	80.70	83.40	96.00	94.74	**96.72**
KrvskpEW	97.80	96.60	94.40	95.60	98.00	96.98	**98.34**
WaveformEW	75.10	66.90	78.60	78.90	75.30	75.00	**80.03**
SonarEW	95.80	93.62	73.10	72.90	97.00	92.26	**97.31**
Clean1	NA	88.32	NA	NA	NA	87.94	**99.86**

Table 9 shows the Wilcoxon signed rank test of the accuracy comparison results between MSSA and six swarm-based algorithms. It can be seen from Table 9 that the null hypothesis is accepted by WOASA, while none of the other five algorithms accepts it. This indicates that MSSA is not significantly different from WOASA, but MSSA is significantly better than the other five comparison algorithms. Although MSSA does not have significant advantages over WOASA, it has achieved higher classification accuracy on most datasets. It indicates that MSSA is superior to the comparison algorithm on these 10 datasets in the comparison experiment.

5.3. Compare with other versions of SSA

In order to further verify the effectiveness of MSSA, it is compared with three variants of SSA (BSSA, TCSSA3, and BSSA_S3_CP) in this subsection. BSSA is the variant of SSA according to Eqs (2.5) and (2.6). In TCSSA3, the swarm of salps is also divided into multiple sub-swarms, which improves the updating model of leader [35]. BSSA_S3_CP adopts the combination of crossover operator and a new transfer function [13]. Table 10 shows the fitness values obtained by MSSA and three versions of SSA. Figures 5 and 6 show the average classification accuracy, the ratio of selected features to total features, and the ranking of F-test obtained by these SSAs on all 10 UCI datasets. Through three indicators denoted by different colors in heatmaps, we can effectively analyze the advantages and disadvantages of four SSAs.

As shown in Table 10, MSSA achieves better fitness values on 70% of all the datasets. In par-

Table 9. The results obtained by MSSA and competitors on Wilcoxon signed rank test (α = 0.05).

Algorithms	HGSA	S-bBOA	bGWO1	bGWO2	WOASA	BHOA_S4_ Cr2
R+	41	55	45	45	34	49
R-	4	0	0	0	11	6
p	0.028	0.005	0.008	0.008	0.173	0.028
Assuming	reject	reject	reject	reject	accept	reject
Select	MSSA	MSSA	MSSA	MSSA	Both	MSSA

Table 10. Comparison between MSSA and other versions of SSA based on average fitness.

Dataset	BSSA	TCSSA3	BSSA_S3_CP	MSSA
Tic-tac-toe	0.2625	0.2600	0.2282	**0.2209**
WineEW	0.1061	0.0607	**0.0547**	0.0619
Zoo	0.0939	0.0604	**0.0419**	0.0589
CongressEW	0.0801	0.0764	0.0693	**0.0607**
Lymphography	0.1794	0.2062	**0.1560**	0.1597
IonosphereEW	0.2062	0.1168	0.1203	**0.0577**
KrvskpEW	0.1022	0.0978	0.0889	**0.0810**
WaveformEW	0.3090	0.3074	0.2971	**0.2287**
SonarEW	0.1689	0.1186	0.1121	**0.0700**
Clean1	0.2046	0.1490	0.1759	**0.0203**
$W \mid T \mid L$	10 | 0 | 0	10 | 0 | 0	7 | 0 | 3	-
F-test	3.9	2.8	1.6	**1.4**

ticular, MSSA outperforms TCSSA3 and BSSA on 90% of all the datasets. Furthermore, BSSA_S3_CP obtains better fitness values on the remaining three datasets (WineEW, Zoo, and Lymphography). Furthermore, for the low-dimensional datasets, the performance of MSSA and BSSA_S3_CP is very similar. For the high-dimensional datasets, however, the advantages of MSSA appear gradually. Besides, MSSA achieves the highest ranking in the F-test among these four approaches.

In Figure 5, the darker the color of the heatmap is, the higher the classification accuracy of an algorithm is. It can be seen from the Figure 5 that the colors of BSSA and TCSSA3 are lighter on most of

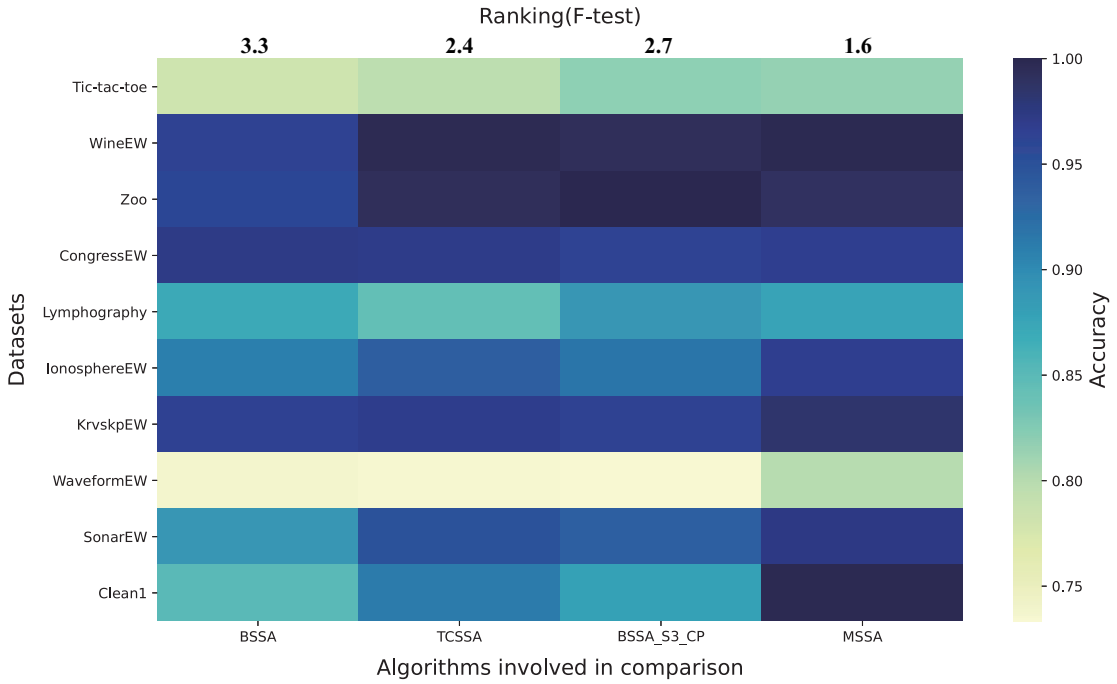


Figure 5. Average classification accuracy of different algorithms.

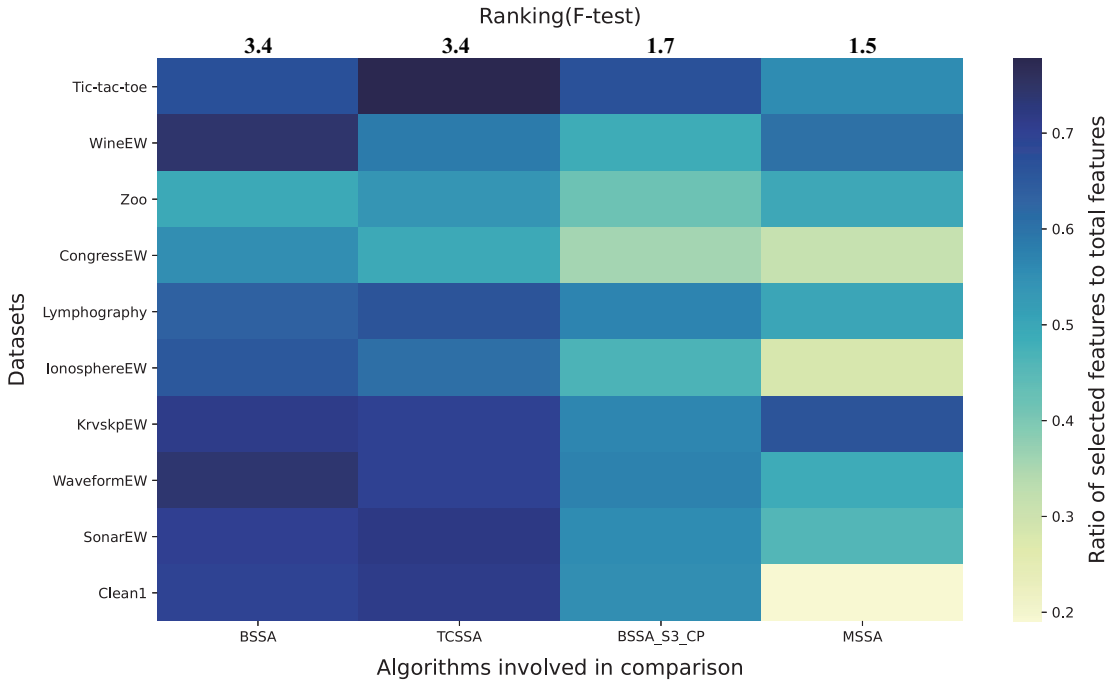


Figure 6. Ratio of features selected to total features by different algorithms.

the 10 datasets, which means that the average classification accuracies obtained by them are not significantly different on these 10 UCI datasets. BSSA_S3_CP and MSSA have similar colors in the first few

low-dimensional datasets. For high-dimensional datasets (i.e., IonphosphereEW, KrvskpEW, WaveformEW, SonarEW, and Clean1), the colors denoting MSSA are significantly darker than BSSA_S3_CP. Combined with the ranking values of F-test, it can be concluded that MSSA has a better performance than the three competitors.

In Figure 6, the lighter the color of the heatmap, the smaller the size of the selected feature subset. It can be seen that MSSA obtains smaller feature subsets on nine out of ten datasets. Especially, for the highest dimensional dataset Clean1, MSSA obtains a very small subset of features compared with three competitors. From Table 10 and Figure 5, it can be found that MSSA obtains the highest classification accuracy by using the least features on the dataset Clean1, while the fitness value obtained by it is far lower than these three methods. According to the ranking values of F-test, MSSA can also obtain smaller sizes of feature subsets on almost all of these datasets.

6. Conclusions

FS is a challenging data pre-processing step in data mining and machine learning. Inspired by SSA, this paper proposes a novel MSSA to solve the FS task in the dataset classification problem. In MSSA, the diversity of the swarm is maintained dynamically through the chaos mapping strategy and multi-swarm strategy. Furthermore, the updating models of leaders and followers are improved so as to utilize more useful informations carried by salps of three sub-swarms. Moreover, the lens-OBL and SBX strategies are employed to enhance the exploitation ability of MSSA. Through analysis, it is found that MSSA can achieve a better balance between exploration and exploitation abilities when solving the FS task. The experimental results show that MSSA is superior to the other 14 competitors on 10 well-known UCI datasets. Especially, MSSA has significant advantages when facing high-dimensional datasets.

Future research can combine the filter strategy with MSSA to improve its performance. We can also try to apply MSSA to other different fields, such as image classification, image processing, and large-scale optimization.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This study is funded by the Basic Public Welfare Research Project of Zhejiang Province (LGF22F020020), the National Natural Science Foundation of China (72101235, 62011530130, 62272415, 62106092), Scientific Research Starting Foundation of Zhejiang Sci-Tech University (20032309-Y), the Key Research and Development Program of Zhejiang Province (2020C03060), Joint Fund of Zhejiang Provincial Natural Science Foundation (LSZ19F010001), the Natural Science Foundation of Fujian Province (2022J01916), and the Research Fund Project of Zhejiang Sci-Tech University Longgang Research Institute (LGYJY2023003).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. M. Rostami, K. Berahmand, E. Nasiri, S. Forouzandeh, Review of swarm intelligence-based feature selection methods, *Eng. Appl. Artif. Intell.*, **100** (2021), 104210. https://doi.org/10.1016/j.engappai.2021.104210
2. B. H. Nguyen, B. Xue, M. Zhang, A survey on swarm intelligence approaches to feature selection in data mining, *Swarm Evol. Comput.*, **54** (2020), 100663. https://doi.org/10.1016/j.swevo.2020.100663
3. C. H. Chen, A hybrid intelligent model of analyzing clinical breast cancer data using clustering techniques with feature selection, *Appl. Soft Comput.*, **20** (2014), 4–14. https://doi.org/10.1016/j.asoc.2013.10.024
4. H. L. Chen, B. Yang, J. Liu, D. Y. Liu, A support vector machine classifier with rough set-based feature selection for breast cancer diagnosis, *Expert Syst. Appl.*, **38** (2011), 9014–9022. https://doi.org/10.1016/j.eswa.2011.01.120
5. S. Tounsi, I. F. Kallel, M. Kallel, Breast cancer diagnosis using feature selection techniques, in *2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, IEEE, (2022), 1–5. https://doi.org/110.1109/IRASET52964.2022.9738334
6. B. Sowan, M. Eshtay, K. Dahal, H. Qattous, L. Zhang, Hybrid PSO feature selection-based association classification approach for breast cancer detection, *Neural Comput. Appl.*, **47** (2023), 5291–5317. https://doi.org/10.1007/s00521-022-07950-7
7. K. Koras, D. Juraeva, J. Kreis, J. Mazur, E. Staub, E. Szczurek, Feature selection strategies for drug sensitivity prediction, *Sci. Rep.*, **10** (2020), 9377. https://doi.org/10.1038/s41598-020-65927-9
8. Z. Zhao, G. Fu, S. Liu, K. M. Elokely, R. J. Doerksen, Y. Chen, et al., Drug activity prediction using multiple-instance learning via joint instance and feature selection, *BMC Bioinf.*, **14** (2013), 1–12. https://doi.org/10.1186/1471-2105-14-S14-S16
9. M. Al-Ayyoub, Y. Jararweh, A. Rabab'ah, M. Aldwairi, Feature extraction and selection for Arabic tweets authorship authentication, *J. Amb. Intell. Hum. Comput.*, **8** (2017), 383–393. https://doi.org/10.1007/s12652-017-0452-1
10. G. Wang, F. Zhang, Z. Li, Multiview feature selection with information complementarity and consensus for fault diagnosis, *IEEE Trans. Syst. Man Cybern.: Syst.*, **53** (2023), 5058–5070. https://doi.org/110.1109/TSMC.2023.3260100
11. Z. Wang, H. Huang, Y. Wang, Fault diagnosis of planetary gearbox using multi-criteria feature selection and heterogeneous ensemble learning classification, *Measurement*, **173** (2021), 108654. https://doi.org/10.1016/j.measurement.2020.108654

12. T. Dokeroglu, A. Deniz, H. E. Kiziloz, A comprehensive survey on recent metaheuristics for feature selection, *Neurocomputing*, **494** (2022), 269–296. https://doi.org/10.1016/j.neucom.2022.04.083

13. H. Faris, M. M. Mafarja, A. A. Heidari, I. Aljarah, A. Z. Ala'm, S. Mirjalili, et al., An efficient binary Salp Swarm Algorithm with crossover scheme for feature selection problems, *Knowl.-Based Syst.*, **154** (2018), 43–67. https://doi.org/10.1016/j.knosys.2018.05.009

14. X. Wang, F. Wang, Q. He, Y. Guo, A multi-swarm optimizer with a reinforcement learning mechanism for large-scale optimization, *Swarm Evol. Comput.*, (2024), 101486. https://doi.org/10.1016/j.swevo.2024.101486

15. S. Huang, Z. Wang, Y. Ge, F. Wang, A coevolutionary estimation of distribution algorithm based on dynamic differential grouping for mixed-variable optimization problems, *Expert Syst. Appl.*, **245** (2024), 123122. https://doi.org/10.1016/j.eswa.2023.123122

16. S. Li, F. Wang, Q. He, X. Wang, Deep reinforcement learning for multi-objective combinatorial optimization: A case study on multi-objective traveling salesman problem, *Swarm Evol. Comput.*, **83** (2023), 101398. https://doi.org/10.1016/j.swevo.2023.101398

17. R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, IEEE, (1995), 39–43. https://doi.org/10.1109/MHS.1995.494215

18. B. Wei, X. W. Xia, F. Yu, Y. L. Zhang, X. Xu, Multiple adaptive strategies based particle swarm optimization algorithm, *Swarm Evol. Comput.*, **57** (2020), 100731. https://doi.org/10.1016/j.swevo.2020.100731

19. F. Wang, Q. He, S. Li, Solving combinatorial optimization problems with deep neural network: A survey, *Tsinghua Sci. Technol.*, **29** (2024), 1266–1282. https://doi.org/10.26599/TST.2023.9010076

20. M, Dorigo, C. G. Di, Ant colony optimization: a new meta-heuristic, in *IEEE Congress on Evolutionary Computation*, IEEE, (1999), 1470–1477. https://doi.org/10.1109/CEC.1999.782657

21. J. Wang, Y. Zhang, M. Hong, H. He, S. Huang, A self-adaptive level-based learning artificial bee colony algorithm for feature selection on high-dimensional classification, *Soft Comput.*, **26** (2022), 9665–9687. https://doi.org/10.1007/s00500-022-06826-1

22. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

23. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Software*, **69** (2014), 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

24. X. S. Yang, Firefly algorithms for multimodal optimization, in *International Symposium on Stochastic Algorithms*, Springer, (2009), 169–178. https://doi.org/10.1007/978-3-642-04944-6_14

25. D. Karaboga, An idea based on honey bee swarm for numerical optimization, in *Technical Report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department*, (2005), 1–10.

26. S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems, *Adv. Eng. Software*, **114** (2017), 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002

27. E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, BGSA: binary gravitational search algorithm, *Nat. Comput.*, **9** (2010), 727–745. https://doi.org/10.1007/s11047-009-9175-3

28. S. Mirjalili, S. M. Mirjalili, X. S. Yang, Binary bat algorithm, *Neural Comput. Appl.*, **25** (2014), 663–681. https://doi.org/10.1007/s00521-013-1525-5

29. M. Abdel-Basset, D. El-Shahat, I. El-henawy, V. H. C. de Albuquerque, S. Mirjalili, A new fusion of grey wolf optimizer algorithm with a two-phase mutation for feature selection, *Expert Syst. Appl.*, **139** (2020), 112824. https://doi.org/10.1016/j.eswa.2019.112824

30. F. Wang, X. Wang, S. Sun, A reinforcement learning level-based particle swarm optimization algorithm for large-scale optimization, *Inf. Sci.*, **602** (2022), 298–312. https://doi.org/10.1016/j.ins.2022.04.053

31. K. Chen, B. Xue, M. Zhang, F. Zhou, Correlation-guided updating strategy for feature selection in classification with surrogate-assisted particle swarm optimization, *IEEE Trans. Evolut. Comput.*, (2022), 1015–1029. https://doi.org/10.1109/TEVC.2021.3134804

32. M. Ghosh, R. Guha, R. Sarkar, A. Abraham, A wrapper-filter feature selection technique based on ant colony optimization, *Neural Comput. Appl.*, **32** (2020), 7839–7857. https://doi.org/10.1007/s00521-019-04171-3

33. Y. Zhang, S. Cheng, Y. Shi, D. W. Gong, X. Zhao, Cost-sensitive feature selection using two-archive multi-objective artificial bee colony algorithm, *Expert Syst. Appl.*, **137** (2019), 46–58. https://doi.org/10.1016/j.eswa.2019.06.044

34. R. Salgotra, U. Singh, S. Singh, G. Singh, N. Mittal, Self-adaptive salp swarm algorithm for engineering optimization problems, *Appl. Math. Modell.*, **89** (2021), 188–207. https://doi.org/10.1016/j.apm.2020.08.014

35. I. Aljarah, M. Mafarja, A. A. Heidari, H. Faris, Y. Zhang, S. Mirjalili, Asynchronous accelerating multi-leader salp chains for feature selection, *Appl. Soft Comput.*, **71** (2018), 964–979. https://doi.org/10.1016/j.asoc.2018.07.040

36. G. I. Sayed, G. Khoriba, M. H. Haggag, A novel chaotic salp swarm algorithm for global optimization and feature selection, *Appl. Intell.*, **48** (2018), 3462–3481. https://doi.org/10.1007/s10489-018-1158-6

37. M. A. Tawhid, A. M. Ibrahim, Improved salp swarm algorithm combined with chaos, *Math. Comput. Simulat.*, **202** (2022), 113–148. https://doi.org/10.1016/j.matcom.2022.05.029

38. A. Choura, H. Hellara, M. Baklouti, O. Kanoun, Comparative study of different Salp Swarm Algorithm improvements for feature selection applications, in *2021 International Workshop on Impedance Spectroscopy (IWIS)*, IEEE, (2021), 146–149. https://doi.org/10.1109/IWIS54661.2021.9711897

39. A. E. Hegazy, M. A. Makhlouf, G. S. El-Tawel, Improved salp swarm algorithm for feature selection, *J. King Saud Univ.-Comput. Inf. Sci.*, **32** (2020), 335–344. https://doi.org/10.1016/j.jksuci.2018.06.003

40. A. D. Tang, T. Han, D. W. Xu, H. Zhou, L. Xie, An improved salp swarm algorithm using Gaussian distribution estimation strategy, *J. Syst. Eng. Electron.*, **44** (2022), 2229–2240. https://www.sys-ele.com/CN/Y2022/V44/I7/2229

41. J. S. Liu, M. M. Yuan, F. Zuo, Global search-oriented adaptive leader salp swarm algorithm, *J. Control. Decis.*, **36** (2021), 2152–2160. https://doi.org/10.13195/j.kzyjc.2020.0090

42. H. Zhang, Z. Cai, X. Ye, M. Wang, F. Kuang, H. Chen, et al., A multi-strategy enhanced salp swarm algorithm for global optimization, *Eng. Comput.*, **38** (2022), 1177–1203. https://doi.org/10.1007/s00366-020-01099-4

43. G. Yavuz, Diversified position update equation-based SSA with refreshing-gap strategy for global optimization, *J. Comput. Sci.*, **60** (2022), 101597. https://doi.org/10.1016/j.jocs.2022.101597

44. R. A. Ibrahim, A. A. Ewees, D. Oliva, M. Abd Elaziz, S. Lu, Improved salp swarm algorithm based on particle swarm optimization for feature selection, *J. Amb. Intell. Hum. Comput.*, **10** (2019), 3155–3169. https://doi.org/10.1007/s12652-018-1031-9

45. M. Qaraad , S. Amjad, N. K. Hussein, M. A. Elhosseini, Large scale salp-based grey wolf optimization for feature selection and global optimization, *Neural Comput. Appl.*, **34** (2022), 8989–90149. https://doi.org/10.1007/s00521-022-06921-2

46. A. H. Gandomi, X. S. Yang, Chaotic bat algorithm, *J. Comput. Sci.*, **5** (2014), 224–232. https://doi.org/10.1016/j.jocs.2013.10.002

47. M. J. Zhang, H. Zhang, X. Chen, J. Yang, A grey wolf optimization algorithm based on Cubic mapping and its application, *Comput. Eng. Sci.*, **43** (2021), 2035–2042. http://manu46.magtech.com.cn/ces/EN/Y2021/V43/I11/2035

48. S. Mahdavi, S. Rahnamayan, K. Deb, Opposition based learning: A literature review, *Swarm Evol. Comput.*, **39** (2018), 1–23. https://doi.org/10.1016/j.swevo.2017.09.010

49. H. R. Tizhoosh, Opposition-based learning: a new scheme for machine intelligence, in *International Conference on International Conference on Computational Intelligence for Modelling* IEEE, (2005), 695–701. https://doi.org/10.1109/CIMCA.2005.1631345

50. F. Yu, Y. X. Li, B. Wei, X. Xu, Z. Y. Zhao, The application of a novel OBL based on lens imaging principle in PSO, *Acta Electron. Sin.*, **42** (2014), 230–235. https://doi.org/10.3969/j.issn.0372-2112.2014.02.004

51. L. Yang, C. Yang, Particle swarm optimization with simulated binary crossover, in *2014 Fifth International Conference on Intelligent Systems Design and Engineering Applications*, IEEE, (2014), 710–713. https://doi.org/10.1109/ISDEA.2014.161

52. P. Subbaraj, P. N. Rajnarayanan, Optimal reactive power dispatch using self-adaptive real coded genetic algorithm, *Electr. Power Syst. Res.*, **79** (2009), 374–381. https://doi.org/10.1016/j.epsr.2008.07.008

53. Q. Zhu, Q. Lin, J. Chen, P. Huang, A gene-level hybrid crossover operator for multiobjective evolutionary algorithm, in *2015 Second International Conference on Soft Computing and Machine Intelligence (ISCMI)*, IEEE, (2015), 20–24. https://doi.org/10.1109/ISCMI.2015.25

54. E. Emary, H. M. Zawbaa, A. E. Hassanien, Binary grey wolf optimization approaches for feature selection, *Neurocomputing*, **172** (2014), 371–381. https://doi.org/10.1016/j.neucom.2015.06.083

55. D. E. Goldberg, J. H. Holland, Genetic algorithms and machine learning, *Mach. Learn.*, **3** (1988), 95–99. https://doi.org/10.1023/A:1022602019183

56. M. Taradeh, M. Mafarja, A. A. Heidari, H. Faris, I. Aljarah, S. Mirjalili, et al., An evolutionary gravitational search-based feature selection, *Inf. Sci.*, **497** (2019), 219–239. https://doi.org/10.1016/j.ins.2019.05.038

57. S. Arora, P. Anand, Binary butterfly optimization approaches for feature selection, *Expert Syst. Appl.*, **116** (2019), 147–160. https://doi.org/10.1016/j.eswa.2018.08.051

58. M. M. Mafarja, S. Mirjalili, Hybrid whale optimization algorithm with simulated annealing for feature selection, *Neurocomputing*, **260** (2017), 302–312. https://doi.org/10.1016/j.neucom.2017.04.053

59. M. A. Awadallah, A. I. Hammouri, M. A. Al-Betar, M. S. Braik, M. Abd Elaziz, Binary Horse herd optimization algorithm with crossover operators for feature selection, *Comput. Biol. Med.*, **141** (2022), 105152. https://doi.org/10.1016/j.compbiomed.2021.105152

AIMS Press