A Semi-supervised Bacterial Heuristic Feature Selection Algorithm for High-Dimensional Classification with Missing Labels

Hong Wang[1], Yikun Ou[1], Yixin Wang[1], Tongtong Xing[1], Lijing Tan[2]*
[1] College of management, Shenzhen University, Shenzhen, China
[2] Department of Business Management, Shenzhen Institute of Information Technology, Shenzhen, China

Abstract. Feature selection is a crucial method for discovering relevant features in high-dimensional data. However, most studies primarily focus on completely labeled data, ignoring the frequent occurrence of missing labels in real-world problems. To address high-dimensional and label missing problems in data classification simultaneously, we proposed a semi-supervised bacterial heuristic feature selection algorithm. To track the label missing problem, a *k*-nearest neighbor semi-supervised learning strategy is designed to reconstruct the missing labels. Additionally, the bacterial heuristic algorithm is improved using hierarchical population initialization, dynamic learning, and elite population evolution strategies to enhance the search capacity for various feature combinations. To verify the effectiveness of the proposed algorithm, three groups of comparison experiments based on eight datasets are employed, including two traditional feature selection methods, four bacterial heuristic feature selection algorithms, and two swarm-based heuristic feature selection algorithms. Experimental results demonstrate that the proposed algorithm has obvious advantages in terms of classification accuracy and selected feature numbers.

Keywords. Semi-supervised feature selection; Bacterial colony optimization; High-dimensional classification

1 Introduction

The dimensionality of the data, which consists of many features, is one of the most influential aspects of the classification model's effectiveness[1]. Based on feature properties, instances can be categorized into their respective classes. However, redundant, irrelevant, and noisy features in high-dimensional data will hamper classification accuracy[2, 3], e.g., medical or clinic data classification [4, 5]. Particularly, it is challenging to distinguish between representative and meaningless features without prior knowledge[6]. On the other hand, due to statistical norms and personal errors, data

classification in real life often faces missing labels and loses more valid samples problems, which ultimately reduces the accuracy and robustness of the classification model[5, 7, 8]. To reduce the feature dimensionality and improve the classification performance in classification, feature selection (FS)[9] is recommended to collect more relevant feature subsets from the original data space. As a tool for optimizing data space, FS can make classification less complicated and improve the precision of classification models[10].

FS methods can be categorized as filter, wrapper, or embedded based on various feature evaluation criteria[11]. Filter methods use specific statistical metrics, such as information gain[12] and fisher score[13], to evaluate the performance of the created feature subsets, while wrapper methods use learning algorithms, such as *k*-nearest neighbor[14], naive bayes[15], and linear discriminant analysis[16]. Embedded approaches, such as the least absolute shrinkage and selection operator[17] and ridge regression[18], embedded FS into the training process for the learner. Filter methods typically execute faster than wrapper methods, but they cannot frequently achieve a higher degree of classification precision[19]. In addition, the process of designing embedded methods is intricate and necessitates plenty of prior experience[20]. Since the high-dimensional classification problem with only partial labels is already a hard task, this research investigates wrapper-based FS methods to ensure a higher accuracy while avoiding the increasing of classification difficulties.

Wrappers seek to find the best subset from feature space according to one predetermined performance assessment. However, it is realistic to select all possible subsets of features measured by wrappers in high dimensional classification problems because of the computational cost. Recently, wrapper based on population-based algorithms have been wildly developed without the necessity of evaluating all possible subsets. Tran[21] proposed the first variable-length particle swarm optimization representation for FS, enabling particles to have different and shorter lengths, which improves the performance of the algorithm. Considering the convergency of population, Song[22] proposed a variable-size cooperative coevolutionary strategy to optimize the searching population which employs the idea of “divide and conquer” in the cooperative coevolutionary approach. However, since wrapper-based FS methods do not perform feature filtering in advance, the searching space of them is the whole data space[23]. This means that in high-dimensional classification tasks, their search space is very large, so they have to use a heuristic strategy like random search to reduce the cost

of computation[24]. Nevertheless, classic heuristic strategy wrapper-based FS methods, such as particle swarm optimization based FS[25], differential evolution based FS[26], and genetic algorithm based FS[27], do not account for all potential feature combinations[28] .

In recent years, bacteria-based algorithms such as bacterial foraging optimization (BFO)[29] have been used to design FS methods to resolve combinatorial difficulties due to global searching capability for control and optimization[30]. However, the intricate structure of BFO limits its computation efficiency. To achieve efficient classification results, bacterial colony optimization (BCO) [31] with a new bacterial life cycle was proposed and laid the foundation for the following research on bacterial-based FS algorithms and applications[6, 28, 30, 32, 33]. The majority of those research offer algorithmic enhancements in terms of weight setting, parameter optimization, and learning strategy optimization. Nonetheless, in actual applications, the integrity of data itself is a significant element influencing the efficiency of FS, particularly the problem of incomplete sample labels, which is the most common and complicated task. This study focuses on developing an enhanced bacterial-based FS approach with a semi-supervised learning strategy to address the high-dimensional medical data classification with partial labels.

According to the integrity of data labels, learning tasks can be categorized into supervised learning, semi-supervised learning, and unsupervised learning [33]. In the supervised task, the training data has complete label information, whereas, in semi-supervised learning, the label information is only available in part. Unsupervised learning means that the analyzed data does not contain labels[34]. In the absence of prior knowledge, the accuracy of supervised learning is generally higher than that of semi-supervised learning. Nevertheless, the cost of getting complete labeled data is extremely high in practical medical data collection. Moreover, unsupervised learning is usually used to disclose the initial pattern of unlabeled data[35]. Therefore, to address the high-dimensional classification difficulty and label missing limitations in medical data, this paper investigated semi-supervised medical data classification problems and optimized the classification model by learning from partially labeled data to classify unlabeled data into the correct class.

Semi-supervised learning has been widely studied in different fields, in the human activity recognition (HAR) problem, Chen[36] designed a semi-supervised deep learning model that is useful in solving the problem of imbalanced distribution of labeled data over classes from multimodal wearable sensory data. In video semantic recognition

problems, Luo[37] proposed a novel semisupervised feature selection method to learn the optimal graph, which aims to upgrade the performance of video semantic recognition. Since the research mentioned above are based on multi-modal data, it makes more sense to employ deep learning or graph machine learning to overcome the problem of missing data labels. Despite the fact that these methods are effective for multimodal data, they incur substantial computational costs. Frequently, for a single mode of data, it is not necessary to use overly complex techniques. In contrast, feature selection methods based on wrapper need less computation, hence they are more suitable for single-type data. In terms of wrapper based FS methods, certain representative semi-supervised classification algorithms, such as the ensemble SVM based semi-supervised FS[38] and the rough set based semi-supervised FS[39], have demonstrated the ability of ensemble learning to solve label missing problems. Nevertheless, these methods rely on the ensembled classifiers to choose the best subset by voting on the results, which increases the computational cost marginally. As a straightforward and easy-to-use technique, *K*-nearest neighbors (KNN) based semi-supervised learning[40] offers great promise for improving the classification effect with missing labels. A number of studies utilize semisupervised KNN. Zhang[38] demonstrated that the introduction of semi-supervised learning with K-Nearest Neighbors (KNN) can enhance the available training sample size, provided that K is held constant. However, Mehta[41] discovered that the magnitude of *K* would impact the efficiency of the algorithm. The precision of the results was enhanced by his use of an exhaustive procedure to determine a suitable value of *K* for solving the problem. Nevertheless, in partially labeled data, different learning densities of KNN may lead to biased results. When the value of *K* is small, the model learning may not be comprehensive; When the value of *K* is large, the operation cost may increase. In other words, the selection of the *K* value is a key issue to be explored. Therefore, this study attempts to develop a new semi-supervised KNN learning approach that allows for the selection of *K* and can be combined with bacterial-based FS to form an effective classification method.

In this study, we propose a semi-supervised bacterial heuristic feature selection (SHBFS) algorithm for medical data classification mentioned earlier, i.e., label incomplete and high-dimensional redundant features. The main contributions of this research are as follows:

- A new self-adjusted semi-supervised feature selection approach is proposed to solve the classification problems with missing labels and high-dimensional

redundant features using a two-step self-training mechanism and an improved bacterial heuristic method.

- The strategies of hierarchical population initialization, dynamic learning, and elite population evolution are proposed to enhance the capacity of the bacterial heuristic algorithm in searching for various feature combinations.
- The proposed semi-supervised bacterial heuristic feature selection algorithm is studied to be superior in addressing label incomplete and high-dimensional classification tasks in comparison to several state-of-the-art semi-supervised FS algorithms.

The rest of this paper is organized as follows: Section 2 gives the background of bacterial-based feature selection methods and some related works on these topics. The proposed method is introduced in Section 3. In Section 4, the experimental configuration is given. The experiments and analyses of the results are provided in Section 5. The final section presents the conclusions and a description of future work.

2 Related Work

The life cycle of the searching algorithm of the proposed bacterial-based FS approach in this study is inspired by BCO. Thus, this section briefly introduces its main principle and reviews bacterial-based feature selection methods. More details are as follows.

2.1 Bacterial Colony Optimization

The life cycle of the BFO is a triple nested loop structure, which brings enormous computational complexity to solve high-dimensional problems. BCO simplifies the life cycle according to specific rules to address this computational drawback. Similar to BFO, BCO contains reproduction and elimination-dispersal processes. However, the chemotaxis steps in BCO are simplified as running and tumbling processes. The conditional controlling rules are used to cope with the traditional triple nested loop structure to improve algorithm efficiency. The pseudocode of BCO is shown in Pseudo-code 1.

- ***Running process:*** The running process is designed to speed convergence to the optimal position as Eq.(1).

$$\theta_i^t = \theta_i^{t-1} + r_i \cdot (gbest - \theta_i^{t-1}) + (1 - r_i) \cdot (pbest_i - \theta_i^{t-1}) \tag{1}$$

where r_i shows the learning coefficient randomly generated between [0,1], *gbest* is the best position in the current bacterial colony, and $pbest_i$ represents the individual optimal position during the chemotaxis process. In addition, communication schemes

such as dynamic neighbor and group-oriented learning can be embedded into the running process.

- ***Tumbling process:*** The tumbling process avoids being trapped in the local optimum and explores more potential solution spaces. As shown in Eq.(2), a random direction vector Δ_i is generated between [-1, 1] for the *i*th bacterium. While the randomly generated direction is correct, i.e., current fitness is improved, the bacterium will continue to exploit in the same direction.

$$\theta_i^t = \theta_i^{t-1} + r_i \cdot (gbest - \theta_i^{t-1}) + (1 - r_i) \cdot (pbest_i - \theta_i^{t-1}) + C(i) \cdot \frac{\Delta_i}{\sqrt{\Delta_i^T \cdot \Delta_i}} \tag{2}$$

The reproduction and elimination mechanisms in BCO are consistent with those in BFO[29]. For the reproduction operation, half of the population with better performance is used to replace the remaining half with poor performance, while the elimination of BCO is realized by assigning the bacterium a new and random position within the search space. It can be formulated as:

If $P < Ped$, then

$$\theta^t = lp + rand \times (up - lp) \tag{3}$$

Otherwise,

$$\theta^t = \theta^t \tag{4}$$

where Ped is a constant to determine the probability of the *i*th bacterium being assigned in a new position and *up* and *lp* are the upper and the lower boundaries of the search space, respectively. In this study, all BFO or BCO-based FS are referred to as bacterial-based FS, and the pertinent research reviews are detailed in the following section.

Pseudo-code 1. Bacterial Colony Optimization (BCO)

```
Input: original data
Initialization: P (Population), MaxIt (Max iterations), C (Chemotaxis step size)
While the maximum iterations are not satisfied do
    For each bacterium do
        Chemotaxis process (refer to Eq.(1))
        Fitness Evaluation
        If Previous fitness < Current fitness
            Tumbling process (refer to Eq.(2))
        End//Alternative mechanisms
        If the reproduction condition is satisfied do
            Reproduction process (refer to article[29])
        End
        If elimination-dispersal conditions are satisfied do
            Elimination-dispersal according to Eqs.(3)-(4)
```

15: | | **End**
16: | **End**
17: **End**//Life-cycle
18: **Output:** Optimal position with the best fitness

2.2 Bacterial Heuristic Feature Selection Methods

In recent years, research on bacterial-based FS algorithm improvements and applications has gradually increased. The process of bacterial-based FS consists primarily of the following steps[42]: i) input the original data, ii) randomly search different features to form a small subset, iii) use the subset to train the classifier, and with the classification results to guide the bacteria search, iv) output the optimal fitness of the current iteration, v) loop steps ii~iv until the maximum number of iterations is reached, and output the final optimal fitness.

In recent years, bacterial heuristic FS has many applications, including health care, recommendation, recognition, and model training[6, 30, 43, 44]. To improve the classification effect, bacterial-based FS has been improved in many ways. One improvement way is weight setting, Wang *et al.*[6] developed a weighted strategy to control the probability of different features being selected to enhance the accuracy. The other improvement way is population optimization, which can be further subdivided into position updates and population updates. For position updates, Chen *et al.*[43] incorporated chaotic mechanisms into the chemotaxis and position-updating stages of bacterial populations to increase their adaptability. For population updates, some studies divided the bacteria into multiple groups to do different jobs under the control of different modified population updating strategies to improve the searching efficiency[32]. Furthermore, learning strategy optimization is also a common and useful improvement way. For example, Kaur *et al.*[45] investigated a multiobjective BFO to improve bacterial learning ability and improve the convergence speed of the algorithm. Wang *et al.* designed an adaptive attribute learning strategy to enhance the information communication ability among bacteria[30].

In summary, bacterial-based FS research focuses mostly on algorithm enhancement and the application of various situations. However, the combined effect of missing labels and high-dimensional redundant features poses significant challenges for optimizers (including bacterial heuristic algorithms) in FS, as the search space of FS problems expands exponentially and the proportion of incomplete data increases synchronously. Therefore, improving the effectiveness and efficiency of bacterial

heuristic algorithms while considering semi-supervised learning methods and data dimension reduction simultaneously is worth studying. Therefore, in this study, we focus on the development of a semi-supervised feature selection approach based on bacteria optimization to solve classification problems with missing labels and high-dimensional redundant features.

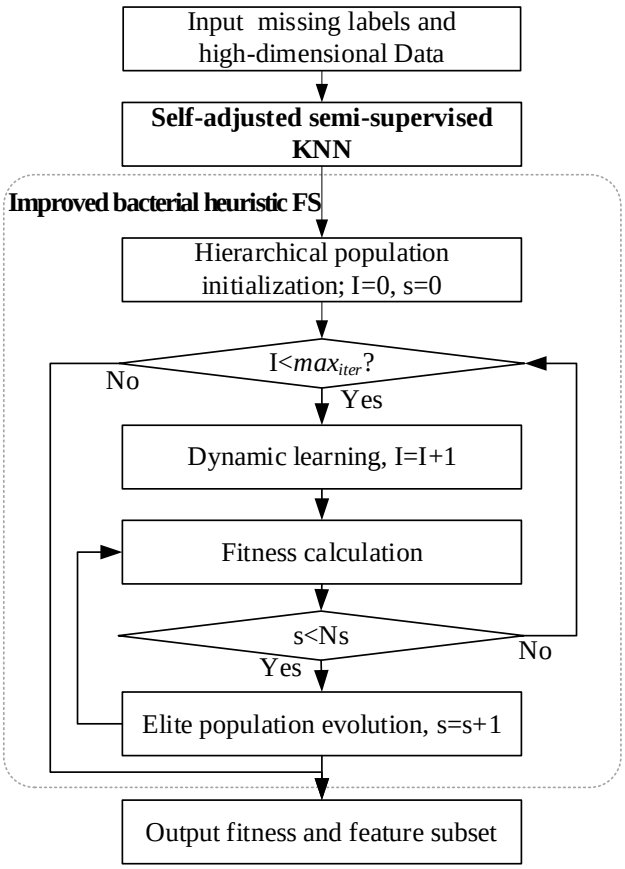


Figure 1 Overall structure of the proposed SHBFS approach.

3 The Proposed Approach

This section presents the proposed SHBFS approach to solve classification problems with missing labels and high-dimensional redundancy features. Figure 1 shows the structure of SHBFS. From the figure, we can find that the SHBFS consists of two main parts. On the one hand, a self-adjusted semi-supervised KNN strategy is presented for solving the problem of missing labels. On the other hand, an improved bacterial heuristic method for FS is presented to address the feature redundant problem, including three improvements: hierarchical population initialization, dynamic learning, and elite population evolution strategy. The hierarchical population initialization is used to obtain informative searching positions for bacteria to accelerate population

convergence. Dynamic learning increases the searching variety of the algorithm by adaptively changing the searching step length of bacteria. Finally, an elite population evolution strategy is employed to enhance the ability of bacteria to escape from the local optimum.

3.1 Self-adjusted Semi-supervised KNN

The proposed self-adjusted semi-supervised KNN is a two-step self-training method consisting of *K*-value determination and label construction. Furthermore, to make the semi-supervised learning method more adaptive to datasets of different sizes of datasets, the *K* value is adjusted using Eq.(5):

$$K = \begin{cases} 1:1:NS, NS < 10 \\ 1:\left[\lg(NS)\right]:\left[10*\lg(NS)\right], 10 \le NS \end{cases} \tag{5}$$

where *NS* is the number of data samples, which means that the *K* value linearly increases when the datasets have smaller samples, while the logarithmic function is employed for the datasets with larger sample sizes. The primary process of the self-adjusted semi-supervised KNN is illustrated as follows:

Step 1. *K*-Value Determination

The samples with labeled class and unlabeled class are separately saved in dataset ***L*** and dataset ***U***. As mentioned previously, a self-adjusted semi-supervised KNN is presented to label the data with no assignment in categories and find the best *K* value for classification. This step is to determine a *K* value for the label reconstruction using the labeled samples in ***L***, provided in Pseudo-code 2.

Pseudo-code 2. Determination of the *K* value

```
Input: L (L is used to save the labeled samples in the original dataset)
  For each K obtained by Eq.(5)
    For each running time
      Randomly divide L into two subsets
      /* half with saved labels and half with removed labels */
      Use the labeled subset to predict the labels of the unlabeled subset by KNN
      Record the accuracy of label prediction on each running time
  End
  Record the K value and the average accuracy of label prediction of each K
End
Output: Best (K value with the maximum average accuracy)
```

Pseudo-code 3. Label reconstruction

01: **Input:** labeled dataset ***L*** and unlabeled dataset ***U***, *best*
02: **For** each unlabeled sample in dataset U
03: | Use the samples from dataset L to predict the label by KNN (using *Kbest*)
04: | Assign the predicted label to the unlabeled sample and move it from ***U*** to ***L***
05: **End**
06: **Output**: the updating dataset L

Step 2. Label Reconstruction

The label reconstruction is provided in Pseudo-code 3. First, self-adjusted semi-supervised KNN is used to predict the labels for the samples from dataset ***U***. Then, the newly labeled samples are moved from dataset ***U*** to dataset ***L***. With increasing ***L*** in the space size, the self-training step can increase the learning efficiency of the training model.

3.2 Hierarchical Population Initialization

In BCO, the population is initialized randomly in a feasible space. However, addressing feature selection with high-dimensional features might make the bacterial colony fall into a poor searching position due to the high uncertainty of population initialization. As a result, more effort will be taken to jump out from their original position, which brings redundant computational complexity. To solve this problem, we develop a hierarchical population initialization strategy to enable the bacteria to start at relatively good positions and further accelerate the convergence speed of the population. In contrast to the aforementioned variable-size cooperative coevolutionary technique, the hierarchical population initialization does not use multi-population for searching. Instead, it uses the idea of the proposed feature hierarchical division strategy to reconstruct a smaller search space before each search. The hierarchical population initialization consists of three steps. The details are as follows.

Step 1. Feature Ranking and Filtering

Initially, symmetrical uncertainty (SU)[21] ranking is performed on the original features according to Eqs.(6)-(9). In this step, the correlations between features and classes are ranked, and the features' relevance significance is ordered from highest to lowest. After ranking, the worst 10 percent of features with significance below the mean are eliminated.

- *Symmetrical uncertainty* (SU)***:*** The SU index has been widely used in traditional FS methods based on information theory. SU measures the uncertainty between feature variables $f \in F$ with the label signals $l \in L$ given in Eqs.(6)-(9) based on

Shannon information entropy. In those formulas, $p(f)$ is the prior probability for all values of *f*, and $p(f \mid l)$ is the posterior probability of *f* given *l*.

$$SU(f,l) = 2\frac{H(f) - H(f \mid l)}{H(f) + H(l)} \tag{6}$$

$$H(l) = -\sum_{i=1}^{N} p(l_i)\log_2 p(l_i) \tag{7}$$

$$H(l) = -\sum_{l=1}^{N} p(l_i)\log_2 p(l_i) \tag{8}$$

$$H(f \mid l) = -\sum_{j=1}^{N} p(l_j)\sum_{i=1}^{N} p(f_i \mid l_j)\log_2 p(f_i \mid l_j) \tag{9}$$

where $H(f)$ and $H(l)$ are the entropy of feature variable *f* and label signal *l*, respectively. *N* is the number of observation samples $x \in X$. $SU(f,l)$ evaluates the correlation between features *f* and label signals *l*. A larger $SU(f,l)$ indicates a higher significance of the feature *f* to labels *l*. That means feature *f* has a more robust capability to discriminate the labels, and feature *f* needs to be selected into the feature subset.

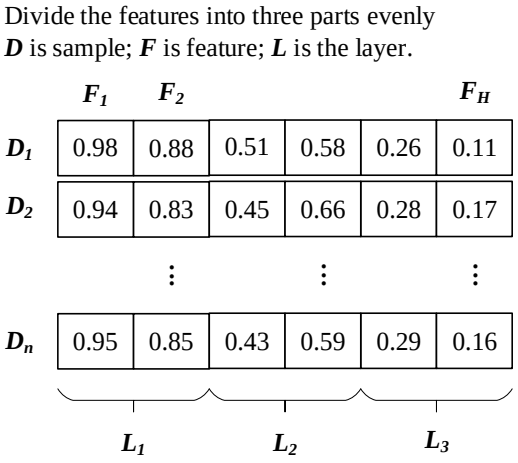


Figure 2 Feature hierarchical division.

Step 2. Feature Hierarchical Division

As shown in Figure 2, it is assumed that the numbers of SU are significant in the box. According to their significance, the sorted features will be divided evenly into three layers, L_1, L_2, and L_3. After this, 80% of the feature dimension will be selected randomly from the L_1 set, 15% from the L_2 set, and 5% from the L_3 set to form a searching position for the bacteria. This strategy can exclude subpar features and shrink the search space when dealing with high-dimensional features.

Step 3. Feature Weight Updating

Assume that the feature size is H, and each feature of the i_{th} bacterium is denoted as f_i. Define the current fitness as $fit(f_i)$ and the historical fitness as $Fit(f_i)$. In this paper, we adopt a weight mechanism[6] to evaluate the performance of features. The rules are as follows: If $fit(f_i) < Fit(f_i)$, then the performance weight pf_i will be increased by Eq.(12). Otherwise, pf_i will be decreased by Eq.(13).

Given that, after completing the aforementioned procedure, there are still unselected features in each feature layer. To increase these features' probability of being selected in the future, we defined the unselected weight (*Uweight*) of f_i as uf_i, and $Uweight = \{uf_1, uf_2, ..., uf_H\}$. Then, the weight of each unselected feature will be updated by Eq.(10) after Step 2. In each feature selection process, if one feature has been selected repeatedly in each search, then its uf_i will be decreased by Eq.(11).

$$uf_i = \begin{cases} uf_i + 0.01uf_i & , \quad uf_i \in L_1 \\ uf_i + 0.001uf_i & , \quad uf_i \in L_2 \\ uf_i + 0.0001uf_i & , \quad uf_i \in L_3 \end{cases} \tag{10}$$

$$uf_i = uf_i - (\max(uf_i) - \min(uf_i)) \tag{11}$$

$$pf_i = pf_i + \frac{|Fit(f_i) - fit(f_i)|}{Fit(f_i)} \tag{12}$$

$$pf_i = pf_i \text{-} Fit(f_i) * |Fit(f_i) - fit(f_i)| \tag{13}$$

where f_i is each feature of the ith bacterium, uf_i is the unselected weight, pf_i is the performance weight, $\{L_1, L_2, L_3\}$ are layers obtained in Step 2, $fit(f_i)$ is the current fitness of f_i, *and* $Fit(f_i)$ is the historical fitness.

3.3 Dynamic learning

In BCO, the chemotaxis step length of bacteria is governed by a set of fixed values denoted by $C(i)$. However, the lack of variation in step lengths may trap the bacteria within the same search space. On a long-term basis, the diversity of feature subsets will decline. Therefore, in SHBFS, the running process is the same as that in BCO, while the tumbling process is improved by employing a dynamic learning strategy to increase the search variety.

Specifically, a dynamic learning strategy is proposed by adopting an adaptive chemotaxis step length changing strategy, which is denoted as aC[46], and a step length communication strategy dC. Eqs.(14)-(15) show that aC is affected by the bacterial size S, where $S = \{1, 2, \ldots, i,\ i \in N^+\}$. Define the current fitness as fit, the upper bound of the

step length C^{ub}, and the lower bound of the step length C^{lb}. ∂ is the disturbance factor. As the iteration proceeds, the disturbance effect of ∂ on aC will become small. In addition, the larger the *fit* is, the larger the value of aC will be. The step length can be changed dynamically by aC.

$$\partial = |(1-\frac{i}{s})\times(C^{ub}-C^{lb})+C^{lb}| \tag{14}$$

$$aC = \frac{fit}{fit+\partial} \tag{15}$$

There is no information communication among the bacteria in BCO. To enhance the convergence speed and improve the search capability, this paper presents a step length communication strategy. Let dC be the step length after communication, and its size is $S\times D$, where $D=\{1,\ 2,\ \ldots,\ d,\ d\in N^{+}\}$ is the dimension of bacteria. Eq.(16) shows the communication process of i_{th} bacteria in the t_{th} iteration.

$$dC^{t} = 0.01\times aC + c_{pi}\cdot R_{pi}\cdot(pbest_i-\theta_i^t)+c_{gi}\cdot R_{gi}\cdot(gbest_i-\theta_i^t) \tag{16}$$

where θ_i^t is the current position of the bacterium, c_p and c_g are the constant learning factors, and R_p and R_g are the random disturbance terms. The R_p and R_g are confined to [0,1]. The step length size in SBHFS learns from the best population record of individual bacteria (*pbest*) and the best population record of the bacteria (*gbest*). For this, the bacteria will prefer to learn from the record with a larger position excursion. After updating the step length dC, bacterial population tumbling is conducted using Eq.(17).

$$\theta_i^t = \theta_i^{t-1} + dC^{t-1}\cdot\frac{\Delta_i}{\sqrt{\Delta_i^T\cdot\Delta_i}}\cdot O_q \tag{17}$$

where Δ_i is a random direction vector generated between [-1, 1] for the ith bacterium. Due to varying data sizes, the range of bacteria location change is greater in large samples than in small samples. So, the $O_q=\{O_1,O_2,q=1,2.\}$ has been proposed in this paper to adjust the offset of the bacterial position. In tumbling process, q=1; in the swimming process, q=2. The setting of O_1, O_2 is in Section 4.3. Defined the number of features of the whole dataset is H, and the selected features subset size is D. If $H<D$, $O_1=O_2=1$.

After tumbling, the feature subset is formed by Eq.(18), where $[\bullet]$ represents the rounding operator. The performance of the feature subset is measured by a classifier. Thus, we adopt the confusion matrix[47] as the evaluation metric, and the fitness is the error rate which will be updated by Eq.(19).

$$\left\{\left[\theta_{i1}^{t}\right],\ldots,\left[\theta_{id}^{t}\right]\right\} \tag{18}$$

$$fit = \frac{FP + FN}{TP + TN + FP + FN} \tag{19}$$

where *FP* is false positive result, *FN* is false negative result, *TP* is true positive result, and *TN* is the true negative result. *fit* is the current fitness. *Fit* is defined as the historical fitness. The current best fitness is *fpbest,* and the historical best fitness is *fgb*est. The main process of dynamic learning is shown in Pseudo-code 4.

Pseudo-code 4. Dynamic learning

```
Input: fit, Pweight, Uweight, fpbest, and fgbest
For each bacterium
    Running process by Eq.(1) // Running
    Update guiding factor aC by Eqs.(14)-(15)
    Update step length dC by Eq.(16)
    Update position by Eq.(17) // Tumbling
    Get feature subset by Eq.(18)
    Update fit by Eq.(19) and update Uweight by Eq.(11)
    If fit < fpbest
        fpbest= fit
        Update Pweight by Eq.(12)
    Else
        Update Pweight by Eq.(13)
    End
    If fpbest < fgbest
        fgbest = fpbest
    End
    Swimming by Pseudo-code 5.
End
Output: fit and feature subset
```

3.4 Elite Population Evolution

In most bacterial based methods, the population will randomly undergo dispersal-elimination. This means that the new searching position of bacteria could be good or bad. The bad searching position may waste the search time. To make population evolution more meaningful, this paper designed an elite population evolution mechanism using *Pweight* and *Uweight* values aforementioned in Section 3.2, which is to guide bacteria to conduct reproduction and dispersal-elimination.

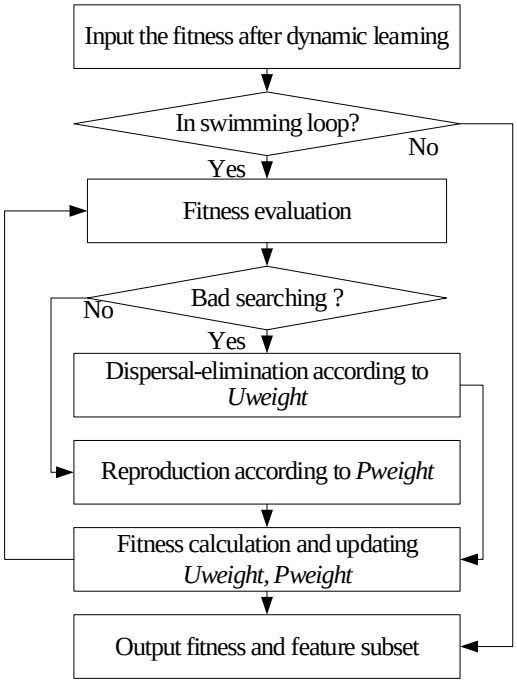


Figure 3 The elite population evolution mechanism.

In SBHFS, either reproduction or dispersal elimination will be conducted per iteration. The elite population evolution mechanism is proposed to determine which operation is executed, as depicted in Figure 3. After dynamic learning, the bacteria will perform a swimming loop as BFO until they meet the threshold *Ns* (see Table 2). In the swimming loop, each bacterium will first undergo a fitness evaluation to determine its performance. The *errTre* is defined as the performance threshold. If the fitness exceeds *errTre*, it will be counted in *bT*. When *bT* is larger than half of *Ns*, we can simply regard this bacterium as performing a bad search, and dispersal-elimination is conducted based on the *Uweight* matrix. Otherwise, bacteria will reproduce based on the *Pweight* matrix. Next, we calculated the fitness of the new bacteria and updated two weights (*Pweight, Uweight*). Finally, repeat the preceding steps until the end of the loop. The main process of the elite population evolution mechanism is given in Pseudo-code 5.

The following is a description of the reproduction and dispersal-elimination processes: Assume the bacteria dimension is *D*. In dispersal-elimination, the seldomly selected features are identified first by ranking the features according to their *Uweight*. The new searching position for bacteria is then determined by randomly selecting *D* features from the top half of the seldomly selected features. In reproduction, the features are initially ranked by their *Pweight* to identify the highest-performing features of the search history. Then, the half population of bacteria with poor performance will be gradually replaced

by the dimensions of the highest-performing features. The overall Pseudo-code of SBHFS is given in Pseudo-code 6, here we analyze the computation time of feature selection (lines 3 to 13) of SBHFS. Suppose that there are *S* bacterium in the population, the max iteration time is *I*, the original number of features is *D*, the swimming time is *M* and $M \ll I$.

Pseudo-code 5. Elite population evolution mechanism

```
01: Input: fit, Fit, Pweight, Uweight, fpbest, and fgbest
02: For each bacterium
03:     While doing swimming loop
04:         If fit < Fit
05:             Fit = fit
06:             If bT > control threshold// it means bad effect searching
07:                 Do dispersal-elimination
08:                 Rank F by Uweight, and save as SF// SF is the sorted features
09:                 Θ= randomly selecting D features from the top half of SF
10:             Else// D is the bacteria dimension
11:                 Do reproduction
12:                 For r = 1:Sr// Sr is the reproduction size of bacteria
13:                     Sort the position Θ of bacteria
14:                     Sort features F by Pweight, and save as SF
15:                     Θ_{r+Sr} = SF(Sr)
16:                 End
17:             End
18:         End
19:         Get feature subset by Eq.(18)
20:         Update fit by Eq.(19) and update Uweight by Eq.(11)
21:         If fit < fpbest
22:             fpbest = fit
23:             Update Pweight by Eq.(12)
24:             If fpbest < fgbest
25:                 fgbest = fpbest
26:             End
27:         Else
28:             Update Pweight by Eq.(13)
29:         End
30:     End
31: End
32: Output: fit and feature subset
```

First, we analyze the initialization part. The feature ranking by *SU* and feature weighting is the main time consumption step. The time complexity of calculating the *SU* scores and weight for features are both *O(D)*, which are related to the number of features. Thus, the time complexity of the initialization part is $O(D)+O(D) \cong O(D)$. We further analyze the time computation of the main loop of Pseudo-code 6 from lines 4 to 13. At iteration *I*, if the dynamic learning step in line 6 is conducted, the time complexity of

this step is $O(SN_I)$, according to the Pseudo-code 4, where N_I is the selected features at iteration I and $N_I \le D$. In the elite population evolution part in line 11 of Pseudo-code 6, if the evolution choice is dispersal-elimination, the time complexity is $O(SM)$. If the evolution choice is reproduction, the time complexity is $O(SM)+O(s) \cong O(SM)$, where s is the population to be updated and $s \le S$ (Usually, s is half of the population). Therefore, in the worst case for one iteration I, the complexity of SBHFS is $O(D)+O(SN_I)+O(SM) \cong O(SN_I)$. Since N_I denotes the number of selected features at iteration I and N_I is smaller than or equal to D, the time complexity of SBHFS at iteration I is smaller than or equal to $O(SD)$. Thus, we can reach the conclusion that the time complexity of the main loop of SBHFS during the I iterations is not larger than $O(ISD)$.

Pseudo-code 6. SBHFS

```
01: Input: missing labels and high-dimensional data, number of selected features: D
02: Semi-supervised learning: labeling by Pseudo-codes 2-3
03: Initialization: parameters setting follows Section 4.3;
    Population initialization follows Section 3.2. iteration: I=0; swimming: s=0;
04: While I < max iterations
05:     I=I+1;
06:     Dynamic learning by Pseudo-code 4
07:     Obtain fitness and feature subset
08:     If s <Ns // Ns is the number of swimming times
09:         s=s+1;
10:         Elite population evolution by Pseudo-code 5
11:         Obtain fitness and feature subset
12:     End
13: End
14: Output: fitness and feature subset
```

4 Experimental Configuration

In this section, detailed information on the datasets, benchmark methods, and experimental design is given.

4.1 Datasets

In this paper, we verified the proposed method on different datasets, consisting of five high-dimensional microarray datasets and three benchmark datasets[6]. The description of the selected dataset is given in Table 1. # Features defines the number of original features, # Instances denotes the number of samples, and the number of classes is given in # Class. # Smallest Class is the size of the Class with the fewest instances, whereas # Largest Class is the size of the Class with the most instances. Among these

datasets, Colon, SRBCT, DLBCL, Leukemia-AllAML (LA), and Central Nervous System (CNS) are datasets with the highest number of features up to 7129. All feature values in those five datasets are normalized within [0,1]. Besides, the number of instances relative to the number of features in the last five datasets is considerably lower. Furthermore, all datasets are significantly imbalanced. These traits present FS and classification with formidable challenges. Since the proposed method is intended to handle missing label data, the original data will be transformed into partially labeled data, as described in Section 4.3.

Table 1 Datasets for feature selection

Datasets	# Features	# Instances	# Class	#Smallest Class	# Largest Class
Australian	15	690	2	222	468
German	24	1000	2	300	700
Ionosphere	33	351	2	38	313
Colon	1999	62	2	22	40
SRBCT	2308	83	4	13	35
DLBCL	5469	77	2	25	75
leukemia-ALLAML (LA)	7129	72	4	23	49
Central Nervous System (CNS)	7129	60	2	21	39

4.2 Comparison Methods

The proposed SBHFS is measured and compared with six recently widely recognized bioinspired wrapper FS algorithms, denoted as benchmark methods. The parameters of the comparison algorithms are shown in Table 2.

Table 2 The settings of parameters for benchmark methods

Algorithms	**Parameter settings**
SBHFS	$errTre = 0.6$, $c_p = 0.0015$, $c_g = 0.0015$, $Ns = 4$ $C_{ub} = 0.15$, $C_{lb} = 0.05$, O_1=1000, O_2=100
ACBFO	$Nre = 5$, $Ned = 2$, $Nc = 10$, $Ns = 4$, $alpha = 0.2$, Ped =0.25, $d_{\text{attract}} = h_{\text{repellant}} = 0.1$, $w_{\text{attract}} = w_{\text{repellant}} = 0.2$
ISEDBFO	$Nre = 5$, $Ned = 2$, $Nc = 10$, $Ns = 4$, $alpha = 0.2$, Ped =0.25, $d_{\text{attract}} = h_{\text{repellant}} = 0.1$, $w_{\text{attract}} = 5$, $w_{\text{repellant}} = 10$
SMA	$z = 0.3$, $r \in [0,1]$, $b = [0,1]$
MOBIFS	$Nre = 4$, $Ned = 2$, $Nc = 200$, $Ns = 4$, $alpha = 0.2$, Ped =0.25
BMRFO	$T(x) = \left\| x / \sqrt{1+x^2} \right\|$, $S = 2$, r, r_1, r_2, $r_3 \in [0,1]$
IBFA	$\gamma = 1$, $\beta_0 = 1$, $\alpha = 0.5 - 0.5(t / MaxIt)$

The adaptive chemotaxis bacterial foraging optimization algorithm (ACBFO)[42], improved swarming and elimination-dispersal bacterial foraging optimization

algorithm (ISEDBFO)[42], and multiobjective bacteria-inspired algorithm (MOBIFS)[48] are three recently proposed BFO variants for FS which have good performances. ACBFO proposed an adaptive chemotaxis strategy, and ISEDBFO adopts a hyperbolic tangent function and a roulette technique to improve the search effects of BFO in FS. MOBIFS is an effective multiobjective BFO algorithm that handles FS issues using four information exchange mechanisms. The Slime mold algorithm (SMA)[49], Binary manta ray foraging optimization (BMRFO)[50]and Improved binary butterfly algorithm (IBFA)[51] are three other bioinspired algorithms that have good performance in FS. SMA imitates the slime mold's foraging behavior and introduces the composite mutation strategy and restart strategy. BMRFO is a manta ray heuristic algorithm for FS problem solving that uses a rational transfer function. IBFA uses a new dynamic mutation operator to increase the diversity of the searching population.

Except for the abovementioned six bioinspired benchmark algorithms, to better verify the effectiveness of SBHFS, we designed two more groups of comparison experiments: comparisons with standard BFO and BCO and comparisons with semi-supervised methods.

- *Comparison with standard BFO and BCO*. Based on the basic bacterial evolutionary framework, the SBHFS has been developed with some efficient strategies. This comparison intends to evaluate the enhanced performance of the proposed approach compared with the standard bacterial heuristic algorithm.
- *Comparisons with semi-supervised methods*. The proposed self-adjustment semi-supervised learning method is being evaluated in this experimental group. First, the original datasets are randomly divided into training and test sets. Moreover, the training set is further divided into a labeled subset and an unlabeled subset. Finally, two KNN-based semi-supervised labeling techniques, semi-supervised KNN (SSKNN)[38] and the best K semi-supervised KNN (BKSKNN)[41], are selected to execute on eight incompletely labeled datasets with the SBHFS.

4.3 Experimental Design

In this study, all experiments were performed on a PC with Windows 10, an Intel Core i7-7700, at 3.6 GHz, with 8 GB RAM, and the Windows 10 operating system. Moreover, for all algorithms, the population size is set to 30, and the number of maximum iterations (max_{iter}) is set to 100. All experiments were run independently 30 times. Due to the facility to implement KNN, this paper uses KNN as the learning algorithm to assess the classification performance after FS as in the literature[38, 41]. In

each dataset, 70% of the samples from each class were randomly selected as the training set and the remaining 30% as the testing set. To simulate partially labeled data, this paper divides the training set into half labeled samples and half unlabeled samples (see Section3.1). According to the previous experiments[6], only a small subset of tenths of the features provides the ideal solution. When the number of features for the last five datasets in Table 1 is less than 50, it is possible to attain high classification accuracy. The desired number of features (*Fno.*) is therefore varied between 1 and 10 for the first three datasets (with reduced feature subset size) and between 5 and 50 for the remaining datasets. The parameters of all benchmark methods are given in Table 2.

For evaluation metrics (Eqs.(20)-(25)), the classification error rate (denoted as *Err.*), true positive rate (*TPR*), true negative rate (*TNR*), precision(*Pre*), Gmean (*GM*), and F1 score (*F1*) are used to assess the feature selection results[52]. The effectiveness of the feature selection approaches can be fully reflected by these evaluation metrics. The performance of the classification result on imbalanced data is assessed using the error rate and G-Means. The *TNR* measures a method's capacity to isolate true positive samples (minority samples) from all other samples, whereas the *TPR* measures a method's ability to isolate negative samples (majority samples) from all other samples. The precision gauges a method's capacity to distinguish genuine positive samples from all other positive samples (including true positives and false positives). A thorough evaluation of TPR and precision performance is provided by the F1-score.

$$Err = \frac{FP + FN}{TP + TN + FP + FN} \tag{20}$$

$$TPR = \frac{TP}{TP + FN} \tag{21}$$

$$TNR = \frac{TN}{TN + FP} \tag{22}$$

$$Pre = \frac{TP}{TP + FP} \tag{23}$$

$$GM = \sqrt{TPR + TNR} \tag{24}$$

$$\mathrm{F1} = 2 * \frac{Pre * TPR}{Pre + TPR} \tag{25}$$

Additionally, the Wilcoxon rank sum test[53] was performed on each approach. It is marked as "=" when the p-value is greater than 0.05, meaning there is no significant difference under the significance level of 5%. If the p-value is less than 0.05, the

recommended method is considered more significant than the comparison algorithms and marked as "+". Otherwise, it is marked as "-" if not.

5 Experimental Results and Analyses

This section gives the comparison results and analysis of the three experimental groups. First, the improvement of the proposed bacterial heuristic optimization algorithm is proven by making comparisons with standard BFO and BCO for feature selection. Next, the enhanced semi-supervised method is verified and discussed with two KNN-based semi-supervised methods. Finally, the effectiveness of the overall proposed SBHFS for tracking incomplete data classification is demonstrated. In Table 3-6, the value in bold represents the best value for the current indicator. When the p-value is "=", there is no significant difference between algorithms. Therefore, the evaluation index score corresponding to the p-value will not be bolded.

5.1 Comparisons with standard BFO and BCO

This comparison aims to verify the effectiveness of the proposed three strategies in BHFS, including hierarchical population initialization, dynamic learning, and elite population evolution. Table 3 shows the comparison results among the proposed bacterial heuristic optimization algorithm for FS (BHFS) and BFO for FS (BFOFS) and BCO for FS (BCOFS). The rows of Ave. and Std. show the average and standard deviation classification metrics of 30 independent runs, respectively. The rows of *p* show the significance values obtained Wilcoxon rank sum test.

From the specific data, the feature numbers of these algorithms are consistently unchanged. This is because the controlling strategies for BHFS, BFOFS, and BCOFS are the same (see Section 4.3). Consequently, there is no difference in the significance of *Fno*.

On the whole, excluding *Fno*, BHFS obtains significantly better results in 92 out of 96 cases versus BFOFS while achieving statistically similar performance in 4 cases. Since the proposed three strategies of BHFS are the improvements of the BFOFS and BCOFS, and they are also the key modules to compose the BHFS, where each strategy is interlinked. This result proves that BHFS is better than BFOFS and BCOFS, which reflects that our improvements are effective.

From the comparison between BFOFS and BCOFS, we can find that the classification results of BCOFS perform better than those of BFOFS. This demonstrates that the improved life cycle model in BCOFS performs better than the triple nested loop

structure, in which the optimization capability is further enhanced. Moreover, compared with BFOFS and BCOFS, BHFS achieves significantly better performance in 86 out of 96 cases while obtaining statistically similar results in 10 cases. In particular, it almost achieves the best classification error rate on eight datasets. Except for the German dataset with the 26.9% classification error rate on average, BHFS on other datasets has achieved an accuracy rate of more than 90%, even the 100% accuracy achieved on LA and DLBCL, two microarray datasets. Thus, it is evident that BHFS outperforms both BFOFS and BCOFS. The primary reason is that BHFS has further developed the life cycle with the three proposed strategies, which improve the algorithm's search ability to locate the optimal space in the population initialization step, increase the probability of individual learning in the chemotaxis stage, and enhance the quality of population evolution in the reproduction and dispersal-elimination stages.

Table 4 illustrates the average calculation time for feature selection and classification in each run. Compared to all bacterial-based methods, BHFS achieves a superior classification effect with less computational complexity. Throughout the iteration period, the computing time of the BFO algorithm increases exponentially due to its nested structure. However, the life cycle enhancement offered by BCO streamlines this procedure, hence reducing the computing cost dramatically. Inspired by BCO, BHFS modifies the parts of population updating based on BCO so that reproduction and dispersal-elimination operations are carried out just one at a time, and the algorithm is additionally programmed with a rule to instantly stop iterating when the ideal solution occurs repeatedly.

Table 3 The results of the comparisons with standard BFO and BCO

Methods	Dataset	Colon							SRBCT						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
BHFS	Ave.	**0.038**	27.380	**0.917**	**0.988**	**0.956**	**0.951**	**0.971**	**0.069**	27.440	**0.977**	**0.932**	**0.936**	**0.951**	**0.926**
	Std.	0.011	15.095	0. 028	0.011	0.015	0.015	0.009	0.044	15.040	0.015	0.046	0.038	0.034	0.048
BFOFS	Ave.	0.321	27.500	0.477	0.800	0.731	0.606	0.761	0.472	28.889	0.842	0.543	0.548	0.657	0.520
	Std.	0.076	15.138	0.190	0.070	0.082	0.116	0.054	0.128	15.366	0.042	0.124	0.147	0.095	0.125
	P	+	=	+	+	+	+	+	+	=	+	+	+	+	+
BCOFS	Ave.	0.116	27.500	0.757	0.958	0.877	0.848	0.913	0.150	27.500	0.950	0.854	0.855	0.896	0.841
	Std.	0.022	15.138	0.118	0.059	0.055	0.044	0.016	0.056	15.138	0.017	0.072	0.078	0.049	0.074
	P	+	=	+	=	+	+	+	+	=	+	+	+	+	+
Methods	Dataset	Ionosphere							German						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
BHFS	Ave.	**0.055**	12.909	**0.976**	**0.684**	**0.816**	**0.811**	**0.722**	**0.269**	10.383	**0.317**	**0.919**	0.753	0.526	**0.827**
	Std.	0.015	6.927	0.028	0.143	0.110	0.072	0.055	0.015	5.468	0.092	0.048	0.023	0.076	0.011
BFOFS	Ave.	0.136	15.100	0.936	0.307	0.359	0.494	0.300	0.343	11.983	0.317	0.827	0.726	0.489	0.770
	Std.	0.031	10.005	0.036	0.263	0.154	0.204	0.180	0.028	7.229	0.108	0.095	0.023	0.090	0.032
	P	+	=	+	+	+	+	+	+	=	+	+	=	=	+
BCOFS	Ave.	0.076	16.000	0.967	0.554	0.689	0.725	0.597	0.291	12.000	0.291	0.898	0.744	0.485	0.811
	Std.	0.011	9.950	0.026	0.165	0.088	0.088	0.064	0.010	7.211	0.147	0.076	0.027	0.123	0.016
	P	+	=	+	+	+	+	+	+	=	=	=	=	=	+
Methods	Dataset	CNS							LA						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
BHFS	Ave.	**0.093**	27.420	**0.837**	**0.942**	**0.925**	**0.884**	**0.931**	**0.000**	27.440	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	Std.	0.013	15.066	0.046	0.023	0.019	0.020	0.010	0.000	15.069	0.000	0.000	0.000	0.000	0.000
BFOFS	Ave.	0.428	27.500	0.350	0.723	0.690	0.482	0.703	0.136	27.500	0.886	0.853	0.945	0.867	0.895

	Std.	0.120	15.138	0.166	0.110	0.068	0.125	0.079	0.064	15.138	0.131	0.069	0.062	0.076	0.050
	P	+	=	+	+	+	+	+	+	=	+	+	+	+	+
	Ave.	0.211	27.500	0.633	0.867	0.830	0.734	0.844	0.046	27.500	0.914	0.973	0.962	0.942	0.967
BCOFS	Std.	0.051	15.138	0.131	0.098	0.047	0.059	0.043	0.037	15.138	0.100	0.034	0.043	0.053	0.027
	P	+	=	+	+	+	+	+	+	=	+	+	+	+	+
Methods	Dataset	Australian							DLBCL						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
BHFS	Ave.	**0.359**	5.614	**0.741**	**0.434**	**0.450**	**0.563**	**0.438**	**0. 002**	27.400	**0.999**	**0.998**	**1.000**	**0.999**	**0.999**
	Std.	0.042	2.297	0.076	0.069	0.047	0.034	0.040	0.005	15.069	0.000	0.007	0.000	0.004	0.004
	Ave.	0.489	7.471	0.556	0.457	0.326	0.485	0.374	0.225	27.500	0.587	0.844	0.855	0.692	0.848
BFOFS	Std.	0.049	4.195	0.118	0.159	0.032	0.039	0.065	0.108	15.138	0.225	0.083	0.079	0.166	0.073
	P	+	=	+	+	+	+	+	+	=	+	+	+	+	+
	Ave.	0. 407	7.500	0.689	0.393	0.380	0.518	0.384	0.057	27.500	0.883	0.965	0.961	0.921	0.962
BCOFS	Std.	0.037	4.183	0.065	0.052	0.038	0.029	0.033	0.041	15.138	0.112	0.041	0.037	0.062	0.028
	P	+	=	+	+	+	+	+	+	=	+	+	+	+	+

Table 4 The average computation time (minutes) of BHFS, BFPFS, and BCOFS for each run

Datasets	Algorithms		
	BHFS	BFOFS	BCOFS
Australian	**4.566**	30.104	5.477
German	**4.667**	27.046	5.371
Ionosphere	**4.019**	23.034	4.182
Colon	**2.420**	22.398	3.360
SRBCT	**2.818**	25.946	3.963
DLBCL	**1.803**	29.433	4.388
leukemia-ALLAML (LA)	**2.833**	34.795	4.282
Central Nervous System (CNS)	**3.540**	30.153	6.104

5.2 Comparisons with semi-supervised methods

Since BHFS demonstrated its superiority and usefulness in Section 5.1, this section will evaluate the effectiveness of the proposed self-adjustment semi-supervised KNN strategy for BHFS. In the following context, we refer to BHFS with the self-adjusted semi-supervised KNN strategy as SBHFS. The compared two semi-supervised techniques based on KNN are as followed. One is the semi-supervised KNN (SSKNN)[38], which assigns the unlabeled sample's label to the label of the labeled sample that is closest to it. The best *K* semi-supervised KNN (BKSKNN)[41] is another comparative technique. By learning about their neighbors, BKSKNN also labels the unlabeled samples. Differently, BKSKNN has two steps as opposed to SSKNN, the first of which is to compute the accuracy of the labeling result of KNN using various *K* values. *K* is then set from 1 to 51. The process finds the best *K* with the highest level of labeling accuracy and then uses the best-labeled data to perform the subsequent procedure. These two semi-supervised learning approaches are embedded into BHFS for the comparison of the effectiveness of different semisupervised method, and they are recorded as BHFS-SSKNN and BHFS-BKSKNN, respectively. Table 5 shows the average, standard deviation classification metric and statistical test results of different semi-supervised learning approaches on benchmark datasets. Since all three methods are based on BHFS, their feature subset size control methodologies are identical (see Section 4.3). Consequently, the significance of *Fno.* does not change in the three

methods.

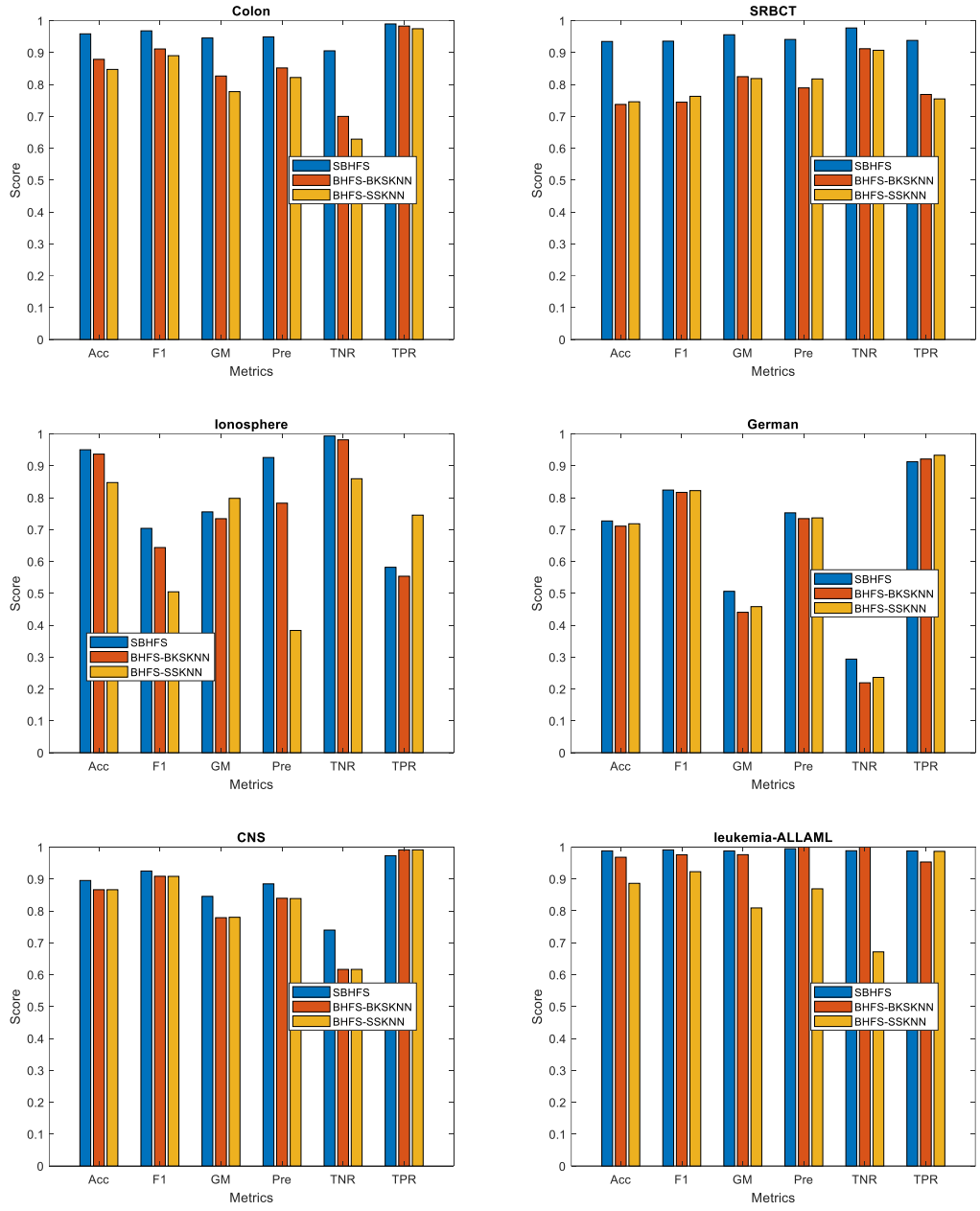
Table 5 The results of the comparisons with semi-supervised methods

Methods	Dataset	Colon							SRBCT						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
SBHFS	Ave.	**0.041**	27.320	**0. 906**	0.990	**0. 949**	**0.946**	**0.968**	**0.065**	27.400	**0. 977**	**0.938**	**0. 941**	**0.956**	**0.936**
	Std.	0.008	14.916	0. 030	0.016	0. 016	0.011	0.006	0.031	15.106	0. 011	0.029	0. 029	0.021	0.029
BHFS -SSKNN	Ave.	0.153	27.500	0. 629	0.975	0. 822	0.778	0.890	0.254	**27.300**	0. 907	0.755	0. 817	0.819	0.763
	Std.	0.046	15.138	0. 138	0.040	0. 054	0.082	0.030	0.046	14.818	0. 018	0.051	0. 044	0.042	0.052
	P	+	=	+	=	+	+	+	+	=	+	+	+	+	+
BHFS -BKSKNN	Ave.	0.121	27.500	0. 700	0.983	0. 852	0.827	0.912	0.263	27.500	0. 912	0.769	0. 789	0.825	0.745
	Std.	0.025	15.138	0. 105	0.035	0. 043	0.051	0.016	0.040	15.138	0. 012	0.040	0. 068	0.030	0.052
	P	+	=	+	=	+	+	+	+	=	+	+	+	+	+
Methods	Dataset	Ionosphere							German						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
SBHFS	Ave.	**0.050**	13.655	**0. 994**	0.582	**0. 926**	0.756	**0.704**	**0.273**	10.517	**0.294**	0.913	**0. 753**	0.506	**0.824**
	Std.	0.013	7.794	0. 005	0.110	0. 049	0.072	0.090	0.016	5.735	0.112	0.028	0.025	0.090	0.005
BHFS -SSKNN	Ave.	0.152	15.500	0. 860	**0.745**	0. 384	0.798	0.505	0.282	11.250	0.236	**0.933**	0.737	0.458	0.822
	Std.	0.016	7.634	0. 020	0.112	0. 035	0.055	0.048	0.010	6.398	0.095	0.049	0.019	0.076	0.011
	P	+	=	+	-	+	=	+	=	=	=	-	=	=	=
BHFS -BKSKNN	Ave.	0.063	14.182	0. 982	0.554	0. 783	0.734	0.644	0.289	11.583	0.219	0.921	0.734	0.441	0.817
	Std.	0.012	8.280	0. 008	0.103	0. 082	0.069	0.082	0.008	7.128	0.089	0.036	0.016	0.076	0.007
	P	+	=	+	=	+	=	=	+	=	=	=	+	=	+
Methods	Dataset	CNS							LA						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
SBHFS	Ave.	**0.104**	27.380	**0.740**	0.973	**0. 885**	**0.846**	**0.925**	**0.012**	27.500	**0.989**	0.988	**0.995**	**0.988**	**0.991**
	Std.	0.006	15.033	0.044	0.024	0.017	0.015	0.005	0.011	15.138	0.020	0.010	0.009	0.013	0.008
BHFS -SSKNN	Ave.	0.133	27.500	0. 617	**0.992**	0. 839	0.780	0.909	0.114	27.500	0. 671	0.987	0. 869	0.809	0.923
	Std.	0.029	15.138	0. 081	0.026	0. 027	0.052	0.019	0.044	15.138	0. 151	0.028	0. 058	0.082	0.029
	P	+	=	+	-	+	+	+	+	=	+	=	+	+	+
BHFS -BKSKNN	Ave.	0.133	27.400	0.617	**0.992**	0.840	0.779	0.909	0.032	27.500	1.000	0.953	1.000	0.976	0.976
	Std.	0.039	14.976	0.112	0.026	0.040	0.070	0.025	0.022	15.138	0.000	0.032	0.000	0.016	0.017
	P	+	=	+	-	+	+	=	=	=	=	=	=	=	=
Methods	Dataset	Australian							DLBCL						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
SBHFS	Ave.	0.350	5.871	**0. 771**	0.398	0. 454	0.553	0.423	**0.003**	27.480	**0. 997**	0.996	**0. 999**	**0.996**	**0.998**
	Std.	0.016	2.600	0. 017	0.043	0. 029	0.030	0.036	0.006	15.105	0. 011	0.008	0. 004	0.006	0.004
BHFS -SSKNN	Ave.	0.483	7.600	0. 419	**0.724**	0. 351	0.489	0.462	0.117	27.500	0. 550	1.000	0. 863	0.740	0.927
	Std.	0.058	2.716	0. 205	0.259	0. 081	0.135	0.154	0.021	15.138	0. 081	0.000	0. 022	0.053	0.012
	P	+	=	+	-	+	=	+	+	=	+	=	+	+	+
BHFS -BKSKNN	Ave.	**0.330**	6.308	0. 788	0.421	**0. 488**	**0.575**	**0.451**	0.052	27.500	0. 800	1.000	0. 936	0.892	0.966
	Std.	0.020	2.594	0. 021	0.050	0. 039	0.035	0.042	0.034	15.138	0. 131	0.000	0. 041	0.073	0.022
	P	-	=	=	=	+	+	+	+	=	+	=	+	+	+

SBHFS outperforms BHFS-SSKNN and BHFS-BKSKNN in the majority of classification evaluation metrics, demonstrating the efficacy of the self-adjusted semi-supervised KNN technique. Self-adjusted semi-supervised KNN will adaptively update the *K* value to find a better label for each sample from varying data sizes, whereas SSKNN will simply apply the fixed *K* value that limited the algorithm's performance.

Although compared with the BHFS-SSKNN, SBHFS obtains a lower error rate in all data cases. In other classification metrics, BHFS-SSKNN shows a better performance with true positive rate (TPR), indicating that the SSKNN method is more capable of correctly labeling the positive samples. While SBHFS can achieve significantly better or similar performance in true negative rate (TNR). This proves that using TPR or TNR metrics alone to judge the performance of algorithms is one-sided. Therefore, it is necessary to deeply analyze the algorithm effect through the remaining three comprehensive evaluation indicators (*Pre, GM, and F1*). The results demonstrate that

the *Pre* scores of SBHFS are 100 percent superior to that of BHFS-SSKNN, while the *GM* scores of SBHFS are superior in five out of eight datasets, and the *F1* scores are superior in half of the datasets. Thus, self-adjusted semi-supervised KNN does improve the performance of SBHFS.



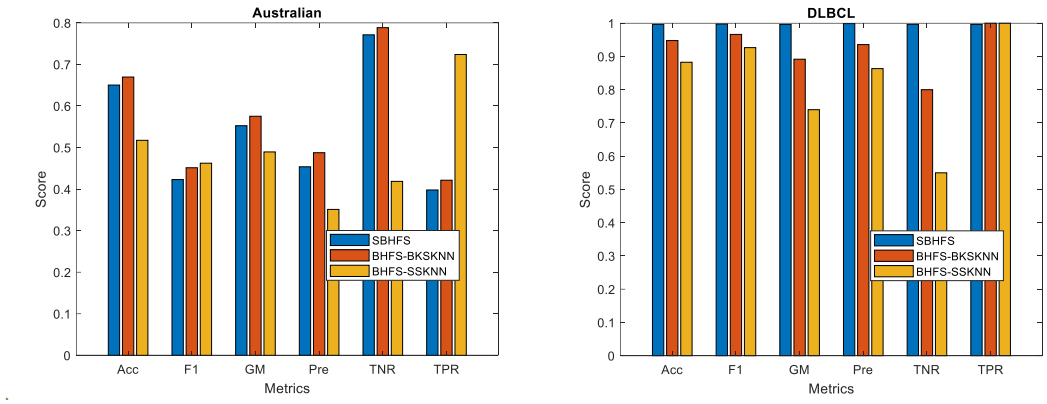


Figure 4 The bar chart of the comparison results with semi-supervised methods.

Figure 4 is the bar chart of the comparison results with semi-supervised methods. The horizontal axis corresponds to evaluation metrics. *Fno.* is excluded since the comparison algorithms use the same feature subset size control methods. The ordinate represents each algorithm's score, with larger values indicating superior performance. From Table 5 and Figure 4, we find that the SBHFS can achieve statistically significantly better classification performance on all high-dimensional datasets with the most different metrics. For benchmark datasets, the statistical significance is not as obvious. One reason is that in a low-dimensional space, the KNN-based semi-supervised learning method is less affected by the value of *K*. Therefore, we conclude that the proposed self-adjusted semi-supervised KNN mechanism is more suitable for high-dimensional data analysis.

5.3 Comparisons with benchmark methods

We further compare the proposed SBHFS with other bioinspired wrapper FS algorithms. Table 6 reports different evaluation metrics (i.e., Err., Fno., TPR, TNR, Pre, GM, F1) of SBHFS and benchmark methods on the test sets. In general, SBHFS achieves competitive results compared to the other six bioinspired feature selection methods, which means that SBHFS is superior to other bioinspired wrapper FS algorithms.

From the specific results, SBHFS performs best on five datasets (i.e., LA, CNS, Colon, SRBCT, and DLBCL) with the most evaluation metrics. However, on the Australian dataset, the proposed method does not perform best. Comparing the results of other algorithms reveals that the effect of SBHFS may be influenced by the sparsity of the features. To be specific, according to the results on the Australian dataset, we can see that smaller subsets produce better results and the number of effective features of the Australian dataset is about 1 or 2. Nevertheless, SBHFS sets the subset size to

integers between 1 and 10 when the total dataset contains fewer than 50 features (see Section 4.3). This increases the average size of the subset in each iteration which exceeds the effective feature size of the Australian. Except for this, based on the results of statistical significance tests on all datasets, SBHFS achieves considerably enhanced efficiency in 229 of 336 cases (39 cases with the '=' p-value are excluded), which illustrates that SBHFS performs well in most of the datasets, especially in high-dimensional ones.

Table 6 The result of the comparisons with benchmark methods

Methods	Indexes	Colon							SRBCT						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
SBHFS	Ave.	**0.041**	27.5	**0.92**	**0.982**	**0.957**	**0.949**	**0.968**	**0.064**	27.5	**0.979**	**0.937**	**0.94**	**0.956**	**0.932**
	Std.	0.015	15.138	0.038	0.015	0.021	0.02	0.012	0.036	15.138	0.012	0.042	0.033	0.029	0.044
ACBFO	Ave.	0.176	390.6	0.665	0.91	0.84	0.769	0.871	0.159	458.35	0.945	0.863	0.877	0.894	0.853
	Std.	0.028	15.364	0.055	0.031	0.026	0.035	0.022	0.03	19.594	0.011	0.028	0.025	0.025	0.028
	P	+	+	+	+	+	+	+	+	+	+	+	+	+	+
ISEDBFO	Ave.	0.153	232.45	0.679	0.938	0.848	0.791	0.888	0.162	246.4	0.944	0.859	0.869	0.891	0.842
	Std.	0.029	56.974	0.073	0.029	0.03	0.046	0.021	0.043	73.685	0.015	0.043	0.04	0.034	0.048
	P	+	+	+	+	+	+	+	+	+	+	+	+	+	+
SMA	Ave.	0.135	**4.3**	0.773	0.916	0.887	0.832	0.896	0.134	**7.833**	0.955	0.872	0.887	0.907	0.863
	Std.	0.035	1.302	0.074	0.06	0.031	0.039	0.031	0.035	5.834	0.011	0.034	0.038	0.025	0.038
	P	+	-	+	+	+	+	+	+	-	+	+	+	+	+
MOBIFS	Ave.	0.25	19.166	0.438	0.79	0.91	0.478	0.845	0.48	18.924	0.914	0.471	0.731	0.561	0.62
	Std.	0.108	2.104	0.489	0.065	0.069	0.402	0.067	0.117	3.621	0.083	0.073	0.326	0.218	0.079
	P	+	=	=	+	=	+	+	+	=	=	+	+	+	+
BMRFO	Ave.	0.179	34.75	0.666	0.905	0.838	0.765	0.866	0.129	216	**0.955**	0.888	0.897	0.914	0.874
	Std.	0.032	35.84	0.092	0.033	0.038	0.055	0.023	0.048	146.98	0.017	0.044	0.035	0.035	0.049
	P	+	=	+	+	+	+	+	+	+	-	+	+	+	+
IBFA	Ave.	0.221	988.3	0.571	0.886	0.801	0.695	0.837	0.19	1140.8	0.93	0.84	0.86	0.88	0.83
	Std.	0.022	20.63	0.04	0.027	0.018	0.029	0.019	0.02	21.71	0.01	0.02	0.02	0.02	0.02
	P	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Methods	Indexes	Ionosphere							German						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
SBHFS	Ave.	**0.062**	14.491	**0.963**	0.721	0. 750	0.829	0.716	**0.246**	10.967	**0.407**	**0.917**	0.777	**0.595**	**0.839**
	Std.	0.03	8.114	0.042	0.099	0.128	0.043	0.062	0.024	5.954	0.125	0.039	0.036	0.114	0.008
ACBFO	Ave.	0.104	4.85	0.976	0.754	0.949	0.855	0.834	0.288	4.5	0.351	0.867	0.758	0.542	0.807
	Std.	0.011	0.988	0.013	0.038	0.027	0.019	0.021	0.01	1.539	0.064	0.027	0.013	0.049	0.009
	P	+	-	=	=	-	-	-	+	-	+	+	+	+	+
ISEDBFO	Ave.	0.085	3.05	0.97	**0.817**	0.94	**0.89**	**0.873**	0.275	4.1	0.409	0.86	0.774	0.588	0.814
	Std.	0.008	0.51	0.009	0.022	0.017	0.012	0.014	0.01	0.553	0.059	0.021	0.014	0.041	0.007
	P	+	-	=	-	-	-	-	+	-	=	+	=	+	+
SMA	Ave.	0.091	**2.5**	0.961	0.816	0.925	0.884	0.864	0.305	**2.9**	0.332	0.851	0.749	0.513	0.793
	Std.	0.013	0.607	0.015	0.021	0.025	0.014	0.019	0.04	0.852	0.083	0.039	0.028	0.082	0.032
	P	+	-	+	-	-	-	-	+	-	+	+	+	+	+
MOBIFS	Ave.	0.677	11.732	0.881	0.118	0.738	0.321	0.203	0.29	9.252	0.497	0.737	**0.924**	0.601	0.82
	Std.	0.059	2.939	0.052	0.016	0.064	0.031	0.026	0.027	3.434	0.125	0.009	0.038	0.08	0.02
	P	+	=	+	+	=	+	+	+	=	=	+	-	=	+
BMRFO	Ave.	0.09	3.1	0.965	0.811	0.936	0.883	0.865	0.279	4.6	0.342	0.884	0.76	0.537	0.816
	Std.	0.009	0.968	0.012	0.023	0.02	0.012	0.014	0.015	1.875	0.101	0.038	0.021	0.087	0.011
	P	+	-	=	-	-	-	-	+	-	+	+	+	+	+
IBFA	Ave.	0.116	9.7	0.978	0.716	**0.951**	0.835	0.814	0.258	9.25	0.441	0.871	0.785	0.617	0.825
	Std.	0.009	1.809	0.009	0.02	0.019	0.013	0.016	0.016	1.743	0.042	0.011	0.013	0.031	0.01
	P	+	=	=	=	-	=	-	+	=	=	+	=	=	+
Methods	Indexes	CNS							LA						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
SBHFS	Ave.	0.082	**27.5**	**0.82**	**0.967**	**0.918**	**0.887**	**0.94**	**0.002**	27.48	**1**	**0.997**	**1**	**0.999**	**0.999**
	Std.	0.018	15.138	0.042	0.022	0.018	0.023	0.014	0.004	15.127	0	0.006	0	0.003	0.003
ACBFO	Ave.	0.38	1411.2	0.444	0.716	0.711	0.53	0.702	0.119	1394.7	0.942	0.853	0. 970	0.894	0.905
	Std.	0.028	37.047	0.085	0.04	0.035	0.061	0.025	0.011	35.534	0.029	0.011	0.012	0.016	0.009
	P	+	+	+	+	+	+	+	+	+	+	+	+	+	+
ISEDBFO	Ave.	0.38	731.8	0.487	0.695	0.724	0.495	0.689	0.121	647.3	0.936	0.858	0.969	0.892	0.905
	Std.	0.031	108.87	0.071	0.059	0.021	0.059	0.038	0.019	30.78	0.046	0.006	0.021	0.027	0.013
	P	+	+	+	+	+	+	+	+	+	+	+	+	+	+
SMA	Ave.	**0.031**	108.87	0.071	0.059	0.021	0.059	0.038	0.036	**3.6**	0.969	0.963	0.986	0.964	0.973
	Std.	0.067	1.353	0.112	0.07	0.052	0.093	0.056	0.028	2.683	0.047	0.034	0.021	0.031	0.021
	P	-	+	+	+	+	+	+	+	-	+	+	+	+	+

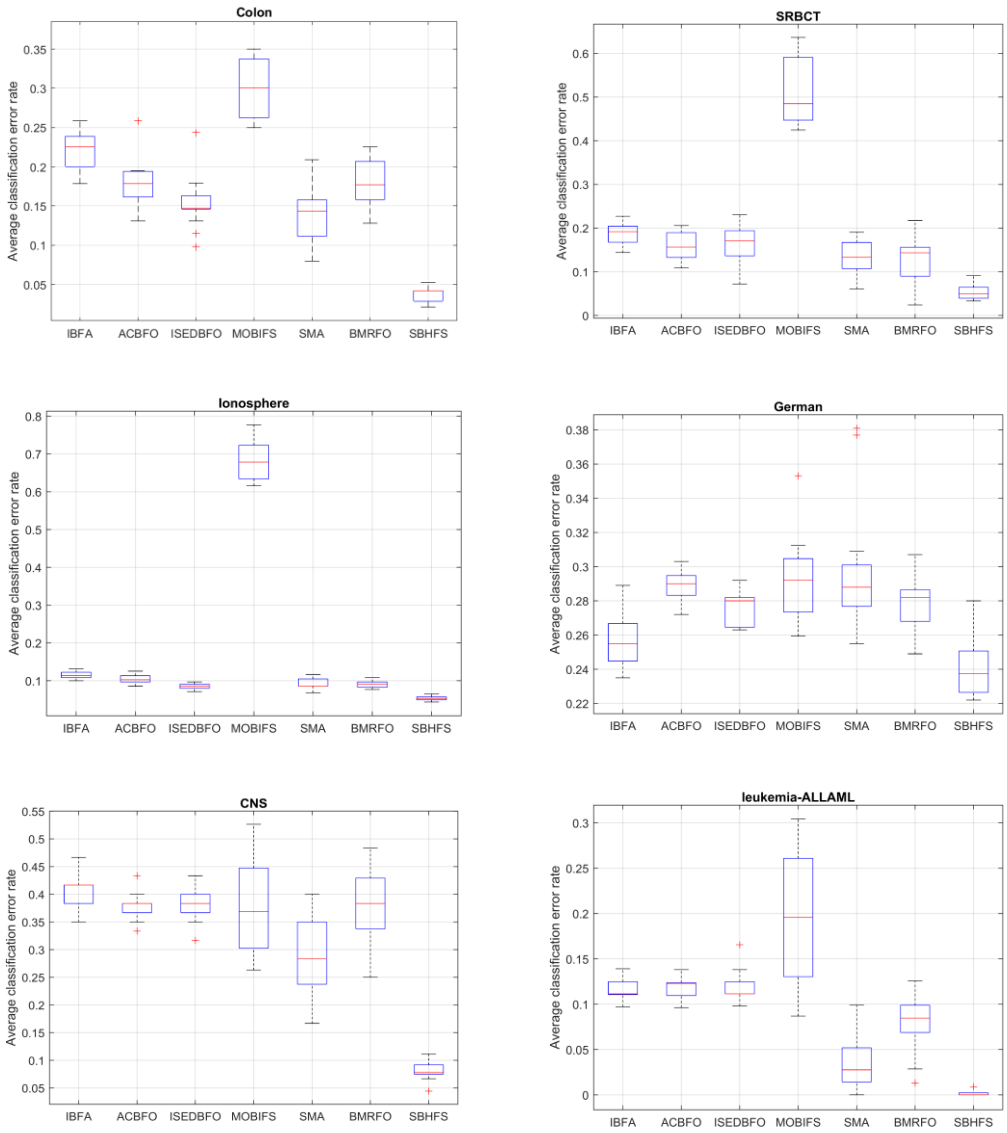
MOBIFS	Ave.	0.333	19.262	0.722	0.683	0.901	0.682	0.775	0.174	20.734	0.902	0.815	0.981	0.854	0.889
	Std.	0.144	2.305	0.357	0.101	0.089	0.285	0.095	0.094	4.515	0.18	0.082	0.029	0.123	0.057
	P	+	=	=	=	=	=	=	+	=	+	+	+	+	+
BMRFO	Ave.	0.38	358.3	0.458	0.701	0.717	0.518	0.699	0.081	16.35	0.925	0.916	0.967	0.917	0.938
	Std.	0.059	470.91	0.103	0.061	0.056	0.096	0.051	0.029	17.236	0.063	0.033	0.027	0.038	0.022
	P	+	+	+	+	+	+	+	+	+	+	+	+	+	+
IBFA	Ave.	0.404	3517.7	0.413	0.694	0.689	0.518	0.69	0.114	3492.3	0.948	0.86	0.972	0.901	0.91
	Std.	0.025	36.131	0.049	0.029	0.021	0.042	0.021	0.012	23.086	0.019	0.013	0.009	0.013	0.009
	P	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Methods	Indexes	Australian							DLBCL						
		Err.	Fno.	TNR	TPR	Pre.	GM	F1	Err.	Fno.	TNR	TPR	Pre.	GM	F1
SBHFS	Ave.	0.357	5.857	0.7	0.524	0.456	0.605	0.487	**0**	27.46	**1**	**1**	**1**	**1**	**1**
	Std.	0.024	2.474	0.03	0.034	0.032	0.025	0.03	0	15.124	0	0	0	0	0
ACBFO	Ave.	0.145	1	0.925	0.799	0.931	0.859	0.859	0.108	1084.6	0.798	0.924	0.934	0.852	0.927
	Std.	0	0	0	0	0	0	0	0.018	30.659	0.063	0.022	0.017	0.033	0.012
	P	-	-	-	-	-	-	-	+	+	+	+	+	+	+
ISEDBFO	Ave.	0.145	**1**	**0.925**	0.799	**0.931**	**0.859**	0.859	0.057	623.6	0.887	0.962	0.964	0.92	0.962
	Std.	0	0	0	0	0	0	0	0.015	127.109	0.048	0.017	0.015	0.025	0.01
	P	-	-	-	-	-	-	-	+	+	+	+	+	+	+
SMA	Ave.	0.145	**1**	**0.925**	0.799	0.93	**0.859**	0.859	0.097	**12.55**	0.81	0.936	0.939	0.864	0.935
	Std.	0	0	0	0	0	0	0	0.036	17.111	0.096	0.028	0.029	0.06	0.024
	P	-	-	-	-	-	-	-	+	-	+	+	+	+	+
MOBIFS	Ave.	0.448	2.556	0.734	0.393	0.625	0.537	0.481	0.22	19.633	0.937	0.784	0.966	0.874	0.865
	Std.	0.024	1.262	0.002	0.015	0.029	0.01	0.005	0.097	2.387	0.081	0.071	0.041	0.068	0.058
	P	-	=	-	-	-	-	=	+	=	+	+	+	+	+
BMRFO	Ave.	**0.144**	1.5	0.912	0.811	0.922	**0.859**	**0.861**	0.104	76.05	0.75	0.942	0.927	0.817	0.931
	Std.	0.002	1.235	0.031	0.028	0.021	0.001	0.006	0.039	81.127	0.112	0.042	0.031	0.089	0.028
	P	-	-	-	-	-	-	-	+	+	+	+	+	+	+
IBFA	Ave.	0.154	2.2	0.87	**0.826**	0.893	0.845	0.855	0.09	2684.9	0.81	0.94	0.94	0.87	0.94
	Std.	0.014	1.152	0.062	0.027	0.04	0.02	0.008	0.02	22.44	0.04	0.01	0.01	0.02	0.01
	P	-	=	=	-	=	=	-	+	+	+	+	+	+	+

From the perspective of the fundamental algorithm, SBHFS achieves notable significance in 150 out of 224 cases (25 cases with the '=' p-value are excluded) in comparison to four other bacterial based FS methods, i.e., ACBFO, ISEDBFO, SMA, and MOBIFS. The results demonstrate that the improvements in SBHFS are better than other bacterial based algorithms in this study. The proposed strategies optimize the searching ability of the algorithm achieving smaller classification error rates and better results on other evaluation indexes. Moreover, the superiority of SBHFS is more obvious on high-dimensional datasets (i.e., # Features > 1999). For example, compared with ISEDBFO, SBHFS achieves an 11.2% lower *Err.* on Colon (# Features = 1999) and 9.8% lower *Err.* on SRBCT (# Features = 2308). This demonstrates that the dimension redundancy capability of the suggested feature selection approach is satisfactory.

Furthermore, compared with BMRFO and IBFA, the SBHFS achieves superior performance in 79 out of 112 cases (14 cases with the '=' p-value are excluded). Specifically, compared to IBFA, SBHFS achieves 9.9% higher *F1*. on LA (# Features = 7129) and 1.4% higher *F1* on SRBCT (# Features = 2308). Compared to comparisons with bacterial-based algorithms, the effects between SHBFS and other bioinspired algorithms are better. This shows that the proposed modified bacterial-based FS algorithm is better for solving dimension reduction problems and that SBHFS can be used not only for high-dimensional datasets but also for some low-dimensional

datasets.

To verify the stability of SBHFS, the boxplot in Figure 5 shows the comparison results of the SBHFS with other bioinspired FS methods. According to the boxplot, except for the Australian dataset with the 35.7% average classification error rate, SBHFS has achieved the best accuracy rate compared with other algorithms on other datasets. Moreover, the median results show that SBHFS generally achieves lower error rates, and the width of boxes indicates that SBHFS is more stable than other comparison methods. This is due to the dynamic learning method that allows the bacterial population of each iteration to move closer to the optimal solution instead of searching randomly.



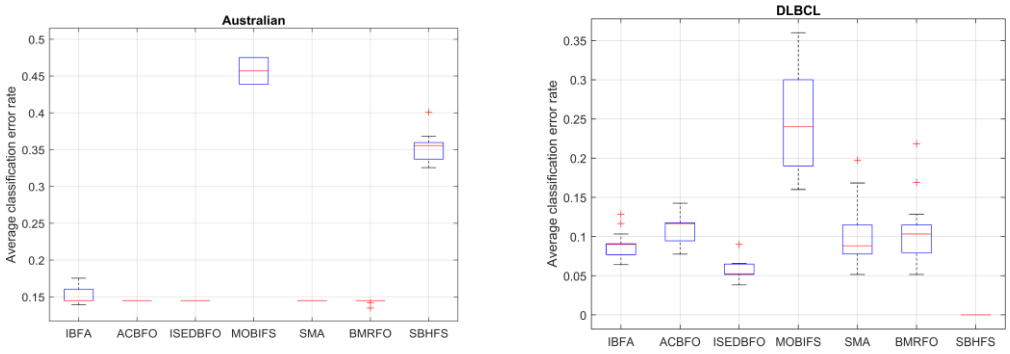


Figure 5 The box-line plot of comparison results with benchmark methods

6 Conclusions

This paper presents a semi-supervised bacterial heuristic feature selection algorithm (SBHFS) to address label incomplete and high-dimensional classification problems. The self-adjusted semi-supervised KNN strategy can reconstruct labels effectively with the help of the two-step self-training mechanism, and the improved bacterial heuristic method can enhance the searching precision by increasing feature selection variety, cooperating with hierarchical population initialization, dynamic learning, and elite population evolution strategies. To be specific, the hierarchical population initialization accelerates the convergence of the algorithm with the help of the *SU* feature ranking method and a proposed layer mechanism. Then, the dynamic learning strategy increases the diversity of the feature subset because it promotes the communication of searching bacteria. Furthermore, the proposed elite population evolution strategy changes the population update method of bacterial based algorithm and improved its optimization performance. The comparisons with the semi-supervised methods show that the proposed semi-supervised learning method is effective for labeling incomplete data, especially on high-dimensional datasets.

Although the proposed SBHFS approach has shown promise in high-dimensional classification with missing labels, the proposed semi-supervised approach is based on the enhancement of the KNN semi-supervised technique, and the semi-supervised method based on other learners is not considered. This may limit the efficiency of the bacterial heuristic algorithm in FS for classification issues involving plenty of sparse features. Considering this information in feature selection may help bacterial heuristic algorithms achieve better results, although this is challenging to accomplish. In our future endeavors, we will consider this direction.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grants Nos. 71901152), Guangdong innovation team project "intelligent management and cross innovation"(2021WCXTD002), and Scientific Research Team Project of Shenzhen Institute of Information Technology (SZIIT2019KJ022).

References

1.W. Wang, X. Yang, X. Li, and J. Tang, Convolutional-capsule network for gastrointestinal endoscopy image classification, *Int. J. Intell. Syst.*, **2022**, *37*. (9). pp. 5796-5815.

2.D. Zhang, L. Yao, K. Chen, S. Wang, X. Chang, and Y. Liu, Making sense of spatio-temporal preserving representations for eeg-based human intention recognition, *IEEE Trans. Cybern.*, **2019**, *50*. (7). pp. 3033-3044.

3.P. Wang, B. Xue, J. Liang, and M. Zhang, Multiobjective differential evolution for feature selection in classification, *IEEE Trans. Cybern.*, **2021**

4.J. Piri, and P. Mohapatra, An analytical study of modified multi-objective harris hawk optimizer towards medical data feature selection, *Comput. Biol. Med.*, **2021**, *135*. pp. 104558.

5.S. Sreejith, H.K. Nehemiah, and A. Kannan, Clinical data classification using an enhanced smote and chaotic evolutionary feature selection, *Comput. Biol. Med.*, **2020**, *126*. pp. 103991.

6.H. Wang, X. Jing, and B. Niu, A discrete bacterial algorithm for feature selection in classification of microarray gene expression cancer data, *Knowledge-Based Syst.*, **2017**, *126*. pp. 8-19.

7.M. Liu, J. Zhang, C. Lian, and D. Shen, Weakly supervised deep learning for brain disease prognosis using mri and incomplete clinical scores, *IEEE Trans. Cybern.*, **2019**, *50*. (7). pp. 3381-3392.

8.Z. Chen, Y. Liu, Y. Zhang, R. Jin, J. Tao, and L. Chen, Low-rank sparse feature selection with incomplete labels for alzheimer's disease progression prediction, *Comput. Biol. Med.*, **2022**. pp. 105705.

9.H. Liu, and H. Motoda, Feature extraction, construction and selection: A data mining perspective (*Springer Science & Business Media*, **1998**, 1998).

10.M.M. Ahmed, and T. Palaniswamy, A novel TMGWO–SLBNC-based multidimensional feature subset selection and classification framework for frequent diagnosis of breast lesion abnormalities, *Int. J. Intell. Syst.*, **2021**, *37*. (3). pp. 2131-2162.

11.S. Liu, H. Wang, W. Peng, and W. Yao, A surrogate-assisted evolutionary feature selection algorithm with parallel random grouping for high-dimensional classification, *IEEE Trans. Evol. Comput.*, **2022**

12.S. Jadhav, H. He, and K. Jenkins, Information gain directed genetic algorithm wrapper feature selection for credit rating, *Appl. Soft Comput.*, **2018**, *69*. pp. 541-553.

13.L. Sun, T. Wang, W. Ding, J. Xu, and Y. Lin, Feature selection using fisher score and multilabel neighborhood rough sets for multilabel classification, *Inf. Sci.*, **2021**, *578*.

pp. 887-912.
14.N. Maleki, Y. Zeinali, and S.T.A. Niaki, A k-NN method for lung cancer prognosis with the use of a genetic algorithm for feature selection, *Expert Syst. Appl.*, **2021**, *164*. pp. 113981.
15.O. Gokalp, E. Tasci, and A. Ugur, A novel wrapper feature selection algorithm based on iterated greedy metaheuristic for sentiment classification, *Expert Syst. Appl.*, **2020**, *146*. pp. 113176.
16.F. Dornaika, and A. Khoder, Linear embedding by joint robust discriminant analysis and inter-class sparsity, *Neural Netw*, **2020**, *127*. pp. 141-159.
17.E. Hancer, B. Xue, and M. Zhang, A survey on feature selection approaches for clustering, *Artificial Intelligence Review*, **2020**, *53*. (6). pp. 4519-4545.
18.J. Fan, and R. Li, Comment: Feature screening and variable selection via iterative ridge regression, *Technometrics*, **2020**, *62*. (4). pp. 434-437.
19.X. Song, M.T. Liu, Q. Liu, and B. Niu, Hydrological cycling optimization-based multiobjective feature-selection method for customer segmentation, *Int. J. Intell. Syst.*, **2021**, *36*. (5). pp. 2347-2366.
20.H. Liu, M. Zhou, and Q. Liu, An embedded feature selection method for imbalanced data classification, *IEEE/CAA J. Autom. Sin.*, **2019**, *6*. (3). pp. 703-715.
21.B. Tran, B. Xue, and M. Zhang, Variable-length particle swarm optimization for feature selection on high-dimensional classification, *IEEE Trans. Evol. Comput.*, **2019**, *23*. (3). pp. 473-487.
22.X.-F. Song, Y. Zhang, Y.-N. Guo, X.-Y. Sun, and Y.-L. Wang, Variable-size cooperative coevolutionary particle swarm optimization for feature selection on high-dimensional data, *IEEE Trans. Evol. Comput.*, **2020**, *24*. (5). pp. 882-895.
23.J. González, J. Ortega, M. Damas, P. Martín Smith, and J.Q. Gan, A new multi-objective wrapper method for feature selection – accuracy and stability analysis for bci, *Neurocomputing*, **2019**, *333*. pp. 407-418.
24.T. Gangavarapu, and N. Patil, A novel filter–wrapper hybrid greedy ensemble approach optimized using the genetic algorithm to reduce the dimensionality of high-dimensional biomedical datasets, *Appl. Soft Comput.*, **2019**, *81*
25.B. Xue, M. Zhang, and W.N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, *IEEE Trans. Cybern.*, **2012**, *43*. (6). pp. 1656-1671.
26.R.N. Khushaba, A. Al Ani, and A. Al Jumaily, Feature subset selection using differential evolution and a statistical repair mechanism, *Expert Syst. Appl.*, **2011**, *38*. (9). pp. 11515-11526.
27.C.F. Tsai, W. Eberle, and C.Y. Chu, Genetic algorithms in feature and instance selection, *Knowledge-Based Syst.*, **2013**, *39*. pp. 240-247.
28.H. Wang, L. Tan, and B. Niu, Feature selection for classification of microarray gene expression cancers using bacterial colony optimization with multi-dimensional population, *Swarm Evol. Comput.*, **2019**, *48*. pp. 172-181.
29.K.M. Passino, Biomimicry of bacterial foraging for distributed optimization and control, *IEEE Control Syst. Mag.*, **2002**, *22*. (3). pp. 52-67.
30.H. Wang, B. Niu, and L. Tan, Bacterial colony algorithm with adaptive attribute

learning strategy for feature selection in classification of customers for personalized recommendation, *Neurocomputing*, **2021**, *452*. pp. 747-755.
31. B. Niu, and H. Wang, Bacterial colony optimization, *Discrete Dyn. Nat. Soc.*, **2012**, *2012*. pp. 1-28.
32. H. Wang, and B. Niu, A novel bacterial algorithm with randomness control for feature selection in classification, *Neurocomputing*, **2017**, *228*. pp. 176-186.
33. Q.Q. Pang, and L. Zhang, Semi-supervised neighborhood discrimination index for feature selection, *Knowledge-Based Syst.*, **2020**, *204*
34. W. Yang, K. Xia, T. Li, M. Xie, and F. Song, A multi-strategy marine predator algorithm and its application in joint regularization semi-supervised elm, *Mathematics*, **2021**, *9*. (3)
35. Y. Zhang, H.G. Li, Q. Wang, and C. Peng, A filter-based bare-bone particle swarm optimization algorithm for unsupervised feature selection, *Appl. Intell.*, **2019**, *49*. (8). pp. 2889-2898.
36. K. Chen, L. Yao, D. Zhang, X. Wang, X. Chang, and F. Nie, A semisupervised recurrent convolutional attention model for human activity recognition, *IEEE Trans. Neural Networks Learn. Syst.*, **2019**, *31*. (5). pp. 1747-1756.
37. M. Luo, X. Chang, L. Nie, Y. Yang, A.G. Hauptmann, and Q. Zheng, An adaptive semisupervised feature analysis for video semantic recognition, *IEEE Trans. Cybern.*, **2017**, *48*. (2). pp. 648-660.
38. D. Zhang, L. Jiao, X. Bai, S. Wang, and B. Hou, A robust semi-supervised svm via ensemble learning, *Appl. Soft Comput.*, **2018**, *65*. pp. 632-643.
39. K. Liu, X. Yang, H. Yu, J. Mi, P. Wang, and X. Chen, Rough set based semi-supervised feature selection via ensemble selector, *Knowledge-Based Syst.*, **2019**, *165*. pp. 282-296.
40. A. Calma, T. Reitmaier, and B. Sick, Resp-knn: A probabilistic k-nearest neighbor classifier for sparsely labeled data *International Joint Conference on Neural Networks (IJCNN)***2016**, pp, 4040-4047.
41. K. Mehta, A. Jain, J. Mangalagiri, S. Menon, P. Nguyen, and D.R. Chapman, Lung nodule classification using biomarkers, volumetric radiomics, and 3d cnns, *J Digit Imaging*, **2021**, *34*. (3). pp. 647-666.
42. Y.P. Chen, Y. Li, G. Wang, Y.F. Zheng, Q. Xu, J.H. Fan, and X.T. Cui, A novel bacterial foraging optimization algorithm for feature selection, *Expert Syst. Appl.*, **2017**, *83*. pp. 1-17.
43. M. Wang, and H. Chen, Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis, *Appl. Soft Comput.*, **2020**, *88*
44. Z. Zeng, L. Guan, W. Zhu, J. Dong, and J. Li, Face recognition based on svm optimized by the improved bacterial foraging optimization algorithm, *Int. J. Pattern Recognit Artif Intell.*, **2019**, *33*. (07). pp. 1956007.
45. M. Kaur, and S. Kadam, A novel multi-objective bacteria foraging optimization algorithm (MOBFOA) for multi-objective scheduling, *Appl. Soft Comput.*, **2018**, *66*. pp. 183-195.
46. J. Too, A.R. Abdullah, and N. Mohd Saad, A new co-evolution binary particle swarm optimization with multiple inertia weight strategy for feature selection, *Informatics*,

2019, *6*. (2)

47.W. Pei, B. Xue, L. Shang, and M. Zhang, Genetic programming for development of cost-sensitive classifiers for binary high-dimensional unbalanced classification, *Appl. Soft Comput.*, **2021**, *101*. pp. 106989.

48.B. Niu, W. Yi, L. Tan, S. Geng, and H. Wang, A multi-objective feature selection method based on bacterial foraging optimization, *Nat. Comput.*, **2019**, *20*. (1). pp. 63-76.

49.H. Jia, W. Zhang, R. Zheng, S. Wang, X. Leng, and N. Cao, Ensemble mutation slime mould algorithm with restart mechanism for feature selection, *Int. J. Intell. Syst.*, **2021**, *37*. (3). pp. 2335-2370.

50.K.K. Ghosh, R. Guha, S.K. Bera, N. Kumar, and R. Sarkar, S-shaped versus V-shaped transfer functions for binary manta ray foraging optimization in feature selection problem, *Neural Comput. Appl.*, **2021**, *33*. (17). pp. 11027-11041.

51.J. Zhang, B. Gao, H. Chai, Z. Ma, and G. Yang, Identification of DNA-binding proteins using multi-features fusion and binary firefly optimization algorithm, *BMC Bioinf.*, **2016**, *17*. (1). pp. 323.

52.A.D. Li, B. Xue, and M. Zhang, Multi-objective feature selection using hybridization of a genetic algorithm and direct multisearch for key quality characteristic selection, *Inf. Sci.*, **2020**, *523*. pp. 245-265.

53.B. Abdollahzadeh, F. Soleimanian Gharehchopogh, and S. Mirjalili, Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems, *Int. J. Intell. Syst.*, **2021**, *36*. (10). pp. 5887-5958.