Natural Heuristic Algorithms to Solve Feature Selection Problem

Yu-Cai Wang, Jie-Sheng Wang *, Jia-Ning Hou, Yu-Xuan Xing

Abstract—**In most data mining tasks, feature selection is an essential pre-processing stage. Select the most important attributes to reduce the dimension of the data set, thus improving the accuracy of the classification. Natural heuristic algorithms are widely used in encapsulated feature selection. Based on the wrapper feature selection method, 7 natural heuristic algorithms are used to solve feature selection problems and perform performance comparison, which include Slime Mold Algorithm (SMA), Whale Optimization Algorithm (WOA), Harris Hawks Optimization Algorithm (HHO), Marine Predator Algorithm (MPA), Butterfly Optimization Algorithm (BOA), Cuckoo Search (CS) and Firefly Algorithm (FA). At the same time, performance tests are carried out on 21 standard UCI data sets to verify the performance of various algorithms, and the convergence curves and accuracy boxplots of 7 natural heuristic algorithms on 21 data sets are given. The simulation results were evaluated according to the mean and standard deviation of fitness, the number of selected features, and the running time, with the optimal value in bold. By comparing the comprehensive performance indexes, MPA obtained the highest average fitness value in most data sets (16 data sets), followed by FA (6 data sets). SMA obtained the best performance and finds the minimum eigenvalues (20 data sets) in multiple data sets and has an advantage in computing time.**

Index Terms—**feature selection, natural heuristic algorithm, KNN classifier, performance evaluation**

I. Introduction

With the development and upgrading of computer science and information technology, many data sets have been produced, the complexity and diversity of data have also increased, the original feature dimension has become higher and higher, and the training space and time complexity of classification model has also increased. High-dimensional data has disadvantages, such as data redundancy and long modeling time, which makes data analysis very difficult [1]. At this point, more excellent and efficient feature selection algorithms are needed to rationalize the processing of many original data and screen out more practical features with classification algorithms. Based on the above factors, feature selection is more concerned by academic circles. Feature selection techniques are knowledge discovery tools that understand problems by analyzing the most relevant features, with the aim of building better classifiers by listing the important features, which also helps reduce the computational burden. The high correlation of features often results in multiple equal-optimal features, which leads to the instability of the traditional feature selection methods, thus reducing the reliability of the selected features. Stability is the robustness of feature preference to the perturbation of training samples. When evaluating feature selection performance, the high stability and classification accuracy of the feature selection algorithm are very important. Feature selection algorithms help to understand the relationship between features and target variables, reduce the computational requirements to solve specific problems, effectively reduce dimension in high-dimensional data sets with fewer observations than features, help to improve predictive performance for solving specific problems, and in terms of cost efficiency and time efficiency has been improved. Feature selection is considered to be an immediate solution to this problem. The aim is to determine which features to use in the classification task without significantly reducing the prediction accuracy of the classifier [2]. As an effective data pre-treatment method, feature selection technology can eliminate irrelevant and redundant features, reduce the dimension of data set, enhance the generalization ability of the model, and reduce excessive fitting [3].

For the initial test results, feature selection is mainly studied on supervised learning algorithm. However, there are also many problems in practical problems. The selection task of test result calculation is often carried out in the case of not knowing the category. For these factors, supervised test result and unsupervised test result algorithm come into being. In 1988, Siedlecki et al. described two types of feature selection algorithms, supervised and unsupervised [4], in which both markers and information of training data could be understood under supervised conditions. At this stage, the primary task of test results is to select the sub-feature set from the initial feature set, so as to maximize the accuracy of the training classifier, or to minimize the feature dimension on the premise of ensuring the accuracy. In 1992, Doak et al. described two types of feature selection algorithms [5]. Later, Belkin et al. proposed a learning algorithm based on manifold in 2003 [6]. The visibility of manifold space is high and data display is easy. The commonly used methods include principal component analysis. In 2003, Zhu et al. proposed a semi-supervised learning method [7] based on the selection

Manuscript received July 24, 2022; revised November 11, 2022. This work was supported by the Basic Scientific Research Project of Institution of Higher Learning of Liaoning Province (Grant No. LJKZ0293), and the Project by Liaoning Provincial Natural Science Foundation of China (Grant No. 2019-ZD-0031).

Yu-Cai Wang is a doctoral candidate of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, P. R. China (e-mail: 1275934857@qq.com).

Jie-Sheng Wang is a professor of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, P. R. China (Corresponding author, phone: 86-0412-2538355; fax: 86-0412-2538244; e-mail: wang_jiesheng@126.com).

Jia-Ning Hou is a postgraduate student of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, P. R. China (e-mail: 19994905806@qq.com).

Yu-Xuan Xing is a postgraduate student of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, P. R. China (e-mail: 805790141@qq. com).

of supervised and unsupervised characteristics and considering advantages and disadvantages, thus reducing the classification cost and improving the performance of the algorithm. Based on the selection mechanism, feature selection can be divided into three categories: filter, wrapper and embedded. Filters can be divided into information gain (IG) [8], gain ratio (GR) [9], mutual information (MI) [10], Laplacian score [11], Fisher score (FS) [12], etc. Wrapper feature selection is based on the performance of the final model to be used. In other words, a wrapper selects a subset of a particular model that best fits its performance. In terms of the final model performance, the wrapper is superior to the filter, but its disadvantage is that it is computation-intensive and requires multiple training of the model. The most representative packaging feature selection method is based on LVM. For large data sets, the search space is very large, so complete search is not feasible. Therefore, meta-heuristic algorithms are widely used in feature selection with the form of wrapper. Finally, the embedded feature selection algorithm performs feature selection in the training process, inheriting the advantages of filter and wrapper feature selection methods. (1) There is an interaction between the selected learning algorithm and the feature. (2) Can record dependencies at a lower cost than wrappers, so they do not need an iterative set of evaluation features. In fact, the embedded approach uses machine learning methods to find the final subset [13].

In recent years, feature selection has become more diverse and diversified, with more suitable search strategies and more comprehensive evaluation criteria springing up. Many algorithms from other fields have also been introduced into feature selection. At the same time, algorithm fusion research has become more popular, such as the synthesis of filters, search strategies, filters and wrappers, evaluation criteria, unsupervised feature selection and supervised feature selection. Heuristic algorithm is a highly coupled calculation method between the problem and the problem to be solved, which can provide a feasible solution according to the calculation method that people can accept, the size of space, etc., under normal circumstances, the deviation range between the possible solution and the optimal solution cannot be predicted in advance. This technique can reduce computational complexity by sacrificing computational accuracy. In this paper, seven different heuristic algorithms (SMA, WOA, HHO, MPA, BOA, CS and FA) are compared on 21 typical UCI data sets based on the encapsulation selection mechanism so as to find the relative optimal algorithm in different data sets. The structure of the paper is arranged as follows. The second section introduces seven new natural heuristic algorithms (SMA, WOA, HHO, MPA, BOA, CS and FA). In section 3, S-type transfer function, K-nearest neighbor classifier (KNN) and fitness function are introduced, and the feature selection architecture based on natural heuristic algorithm is presented. The fourth section carries on the experiment simulation and the result analysis. Finally, the conclusion of the paper is given.

II. Natural Heuristic Algorithm

A. Slime Mold Algorithm (SMA)

As SMA was proposed by Professor Li and Mirjalili et al. in 2020. In this algorithm, it mainly simulates the foraging and diffusion behavior of slime mold, and simulates the adaptive weight generated by positive and negative propagation waves to form the optimal connection path. The main operators of SMA include approaching food (odor index), wrapping food and acquiring food.

(1) Approaching to the food

Slime molds use smells in the environment to get closer to food. Its convergence behavior is expressed mathematically, and Eq. (1) is used to simulate its contraction mode.

$$\overrightarrow{X_{(t+1)}} = \begin{cases} \overrightarrow{X_b}(t) + \overrightarrow{v_b} \cdot \left(\overrightarrow{W} \cdot \overrightarrow{X_A}(t) - \overrightarrow{X_B}(t) \right) & r < p \\ \overrightarrow{v_C} \cdot \overrightarrow{X_{(t)}} & r \geq p \end{cases} \tag{1}$$

where, $\overrightarrow{v_b}$ is between $[-a, a]$; $\overrightarrow{v_C}$ is between $[-1,1]$; the current iteration number is denoted by t ; represents the individual position with the best adaptation found at present is denoted by $\overrightarrow{X_b}$; the location of the slime mold is denoted by $\overrightarrow{X}$; the weight of the slime mold is denoted by $\overrightarrow{W}$; two individuals randomly selected from the slime molds are devoted by $\overrightarrow{X_A}$ and $\overrightarrow{X_B}$; r is between $[0,1]$; the threshold for controlling exploration and exploitation is denoted by p ; p is shown in Eq. (2); $\overrightarrow{v_C}$ is shown in Eq. (3); a is shown in Eq. (4); b is shown in Eq. (5); $\overrightarrow{W}$ is shown in Eq. (6).

$$p = \tanh |S(i) - BF| \tag{2}$$

$$\overrightarrow{v_C} = rand[-b, b] \tag{3}$$

$$a = \arctan h\left(-\left(\frac{FEs}{Max\_FEs}\right)+1\right) \tag{4}$$

$$b = 1 - \left(\frac{FEs}{Max\_FEs}\right) \tag{5}$$

$$\overrightarrow{W}((SI(FEs)) = \begin{cases} 1 + r \cdot \log\left(\frac{BF - S(i)}{BF - WF} + 1\right), condition \\ 1 - r \cdot \log\left(\frac{BF - S(i)}{BF - WF} + 1\right), others \end{cases} \tag{6}$$

where, $i = 1, 2, 3, ..., n$; $S(i)$ is the fitness value indicated by $\overrightarrow{X}$; the best adaptive value currently obtained is denoted by BF ; FEs is the quantity currently assessed; Max_FEs is the maximum quantity assessed; the weight of slime molds in positive and negative feedback mechanisms is denoted by $\overrightarrow{W}$; Condition $S(i)$ represents the top half of the ranking in the population; the value suitable for $\overrightarrow{X}$ is denoted by $S(i)$; the sequence of adaptive values in ascending order is denoted by SI ; r represents the random number in $[0,1]$; the best adaptive value obtained in the current iteration is denoted by BF ; the worst fit obtained during the current iteration is denoted by WF .

(2) Packages of food

The relationship between venous width and food concentration was mathematically simulated by Eq. (6). The parameters r in Eq. (6) are used to simulate the uncertainty of venous contraction mode. At higher concentrations of food, weight in nearby areas increased. At lower concentrations of

food, the area loses weight and the slime molds continue to explore other areas. The evolution formula is shown in Eq. (7):

$$\overrightarrow{X}^{*}=\begin{cases} rand\cdot(UB-LB)+LB, & rand<z \\ \overrightarrow{X_b}(t)+\overrightarrow{\upsilon_b}\cdot\left(\overrightarrow{W}\cdot\overrightarrow{X_A}(t)-\overrightarrow{X_B}(t)\right), & r<p \\ \overrightarrow{\upsilon_C}\cdot\overrightarrow{X_{(t)}}, & r\geq p \end{cases} \tag{7}$$

The upper and lower limits of the search range are represented by UB and LB; $rand$ and r represent random values of $[0,1]$; $rand$ is used to control whether random update is selected; r is used to determine whether to enter the exploration and exploitation phase.

B. Whale Optimization Algorithm (WOA)

WOA was proposed by Mirjalili et al in 2016. It is inspired by the behavior of whales chasing prey. When hunting other fish, whales can do two things, one is to surround the prey, and one is to soak in the net. A round is when all the whales move on to the other whales. A bubble net is a way for a whale to swim in a circle through the water and then blow bubbles to expel prey. In each swim, the whale randomly selects these two movements. As they surround their prey, they randomly decide whether to swim to the best place. The whale optimization algorithm has the following steps.

(1) Initially determine the whale population size as X, and randomly generate the position of a whale. Then the parameter a, A, C, l, p and Max_Iter of WOA algorithm are initialized.

(2) The fitness of each whale is calculated and compared to determine the most suitable individual, which is defined as X^{*}.

(3) Enter the algorithm loop.

If $p<0.5$ and $|A|<1$, each whale updates the current position as shown in Eq. (8); Otherwise, update the individual whale position as shown in Eq. (9).

$$\overrightarrow{X}(t+1)=\overrightarrow{X^{*}}(t)-\overrightarrow{A}\cdot\overrightarrow{D} \tag{8}$$

$$\overrightarrow{X}(t+1)=\overrightarrow{X_{rand}}-\overrightarrow{A}\cdot\overrightarrow{D} \tag{9}$$

$$\overrightarrow{A}=2\vec{a}\cdot\vec{r}-\vec{a} \tag{10}$$

$$\overrightarrow{D}=|\overrightarrow{C}\cdot\overrightarrow{X_{rand}}-\overrightarrow{X}| \tag{11}$$

where, $\overrightarrow{A}$ represents the convergence factor; the distance between the individual and the optimal whale position is denoted by $\overrightarrow{D}$. It can be calculated from Eq. (10) and (11) respectively. $\vec{r}$ is between $[0,1]$; as the number of iterations increases, $\vec{a}$ will decrease from 2 to 0; the location of a random whale in the current population is denoted by $\overrightarrow{X_{rand}}$.

If $p\geq 0.5$, then each whale is calculated according to Eq. (12).

$$\overrightarrow{X'}(t+1)=\overrightarrow{D'}\cdot e^{bl}\cdot\cos(2\pi l)+\overrightarrow{X^{*}}(t) \tag{12}$$

$$\overrightarrow{D'}=|\overrightarrow{X^{*}}(t)-\overrightarrow{X}(t)| \tag{13}$$

where, the distance to the food for whale i is denoted by $\overrightarrow{D'}$; l is a random value between $[-1,1]$; b is the helical constant.

(4) Evaluate the whale population again and find out the global best individual whale and its location.

(5) If the termination condition of WOA is met, it will stop. Otherwise, go to Step 2.

(6) Output the global optimal solution X^{*}.

C. Harris Hawks Optimization Algorithm (HHO)

HHO was a new meta-heuristic algorithm proposed by Heidari et al in 2019 [14]. In the HHO algorithm, eagle is used to represent the candidate solution, and the optimal solution is called the prey. Harris eagles try to use their powerful eyes to track their prey and then pounce on what they find. HHO algorithm is mainly composed of three parts: exploration stage, transformation stage between exploration and exploitation stage, and exploitation stage.

(1) Exploration phase

A Harris eagle perches randomly in a certain area and finds prey by two strategic means, as shown in Eq. (14).

$$X(\tau+1)=\begin{cases} X_{rand}(\tau) & \\ -r_1|X_{rand}(\tau)-2r_2X(\tau)|, & q\geq 0.5 \\ [X_{rabbit}(\tau) & \\ -X_m(\tau)]-r_3[l^b+r_4(u^b-l^b)], & q<0.5 \end{cases} \tag{14}$$

where, individual positions in the current iteration and the next iteration are denoted by $X_{(\tau)}$ and $X_{(\tau+1)}$; the number of iterations is denoted by τ; the individual position selected at random is denoted by $X_{rand}(\tau)$; prey position is denoted by $X_{rabbit}(\tau)$; $r_1\sim r_4$ and q are random numbers between $[0,1]$; q is used for the strategy to be adopted in random selection; $X_m(\tau)$ is the average individual position, as shown in Eq. (15).

$$X_m(\tau)=\frac{1}{M}\sum_{k=1}^{M}X_k(\tau) \tag{15}$$

where, the position of the k-th individual in the population is denoted by $X_k(\tau)$, the size of the population is denoted by M.

(2) Transformation stage between exploration and exploitation

The HHO algorithm relies on escape energy of prey, and escape energy is expressed in Eq. (16).

$$E=2E_0\left(1-\frac{\tau}{T}\right) \tag{16}$$

where, the initial energy of prey is denoted by E_0; the number of iterations is denoted by τ; the maximum number of iterations is denoted by T. When $|E|\geq 1$, the exploration phase will be entered; When $|E|<1$, it will enter the exploitation phase.

(3) Exploitation stage

Definition: r is a random number between $[0,1]$, which is used to choose different exploitation strategies.

1) When $0.5\leq|E|<1$ and $r\geq 0.5$, adopt a soft siege strategy shown in Eq. (17).

$$X(\tau+1)=\Delta X(\tau)-E\,|JX_{rabbit}(\tau)-X_{(\tau)}| \quad (17)$$

$$\Delta X(\tau)=X_{rabbit}(\tau)-X(\tau) \quad (18)$$

where, the difference of the current position of the prey individual is denoted by $\Delta X(\tau)$; J is a random number between $[0,2]$.

2) When $|E|<0.5$ and $r\geq 0.5$, the hard siege method is adopted to update the position, as shown in Eq. (19).

$$X(\tau+1)=X_{rabbit}(\tau)-E\,|\Delta X(\tau)| \quad (19)$$

3) When $0.5\leq|E|<1$ and $r<0.5$, the progressive soft enveloping method is adopted to update the position, as shown in Eq. (20).

$$X(\tau+1)=\begin{cases}Y, f(Y)<f(X(\tau))\\ Z, f(Z)<f(X(\tau))\end{cases} \quad (20)$$

$$Y=X_{rabbit}(\tau)-E\,|JX_{rabbit}(\tau)-X(\tau)| \quad (21)$$

$$Z=Y+S\times LF(2) \quad (22)$$

where, the fitness function is denoted by $f(\cdot)$; a random vector in two dimensions is denoted by s ; $LF(\cdot)$ is Levy's flight mathematical expression.

4) When $|E|<0.5$ and $r<0.5$, the position of Eq. (20) is updated using the asymptotic hard enveloping method, where Y is shown in Eq. (23).

$$Y=X_{rabbit}(\tau)-E\,|JX_{rabbit}(\tau)-X_m(\tau)| \quad (23)$$

D. Marine Predator Algorithm (MPA)

MPA is put forward by Faramarzi etc. [15] in 2020, Marine predator algorithm was proposed by Faramarzi et al. [15] in 2020. MPA algorithm mainly simulates the process of survival of the fittest in the ocean, modeling the predation behavior of Marine predators.It will randomly generate solutions as prey. A solution based on the performance of the fitness function can be divided into two parts, respectively elite prey and predator. Elite predators can supervise the search and find prey based on the location information of prey. The optimization process of MPA is mainly divided into three stages.

(1) Phase one

For phase one, this happens early in the iteration of optimization. Maintaining the current position is the best strategy for the predator at this stage. When $Iter<Max\_Iter/3$:

$$\overrightarrow{stepsize_i}=\overrightarrow{R_B}\otimes(\overrightarrow{Elite_i}-\overrightarrow{R_B}\otimes\overrightarrow{Pery_i}), i=1,\ldots,n \quad (24)$$

$$\overrightarrow{\Pr ey_i}=\overrightarrow{Pery_i}+P.\vec{R}\otimes\overrightarrow{stepsize_i} \quad (25)$$

where, $\overrightarrow{stepsize}$ represents the moving step at this stage; $\overrightarrow{R_B}$ represents Brown walk random vector; $P=0.5$; $\otimes$ is the term by term multiplication operator to represent the movement of prey; R is a random value in $[0,1]$; $\overrightarrow{Elite}$ is an elite matrix of top predators; the current iteration number is denoted by $Iter$; the maximum number of iterations is denoted by Max_Iter.

(2) Phase two

This strategy occurs in the middle of the iteration, and the population is divided into two parts. At this time, both the prey and the predator are looking for the prey. The prey carries out the Levy-motion, which is responsible for the development of the algorithm in the search space, while the predator carries out the Brow motion, which is responsible for the exploration of the algorithm in the search space.

When $\frac{1}{3}Max\_Iter<Iter<\frac{2}{3}Max\_Iter$:

Position change of the first half:

$$\overrightarrow{stepsize_i}=\overrightarrow{R_L}\otimes(\overrightarrow{Elite_i}-\overrightarrow{R_L}\otimes\overrightarrow{\Pr ey_i}), i=1,\ldots,n/2 \quad (26)$$

$$\overrightarrow{\Pr ey_i}=\overrightarrow{Pery_i}+P.\vec{R}\otimes\overrightarrow{stepsize_i} \quad (27)$$

Position change of the latter half:

$$\overrightarrow{stepsize_i}=\overrightarrow{R_B}\otimes(\overrightarrow{R_B}\otimes\overrightarrow{Elite_i}-\overrightarrow{\Pr ey_i}), i=n/2,\ldots,n \quad (28)$$

$$\overrightarrow{\Pr ey_i}=\overrightarrow{Elite_i}+P.CF\otimes\overrightarrow{stepsize_i} \quad (29)$$

where, $\overrightarrow{R_L}$ represents the random vector of Levy motion and CF is the adaptive function of the predator's moving step.

$$CF=(1-\frac{Iter}{Max\_Iter})^{(2\frac{Iter}{Max\_Iter})} \quad (30)$$

(3) Stage three

This stage occurs at the late stage of iteration and should focus on improving the local development of the algorithm. Levy-motion is the best predator strategy.

When $Iter>\frac{2}{3}Max\_Iter$:

$$\overrightarrow{stepsize_i}=\overrightarrow{R_L}\otimes(\overrightarrow{R_L}\otimes\overrightarrow{Elite_i}-\overrightarrow{\Pr ey_i}), i=1,\ldots,n \quad (31)$$

$$\overrightarrow{\Pr ey_i}=\overrightarrow{Elite_i}+P.CF\otimes\overrightarrow{stepsize_i} \quad (32)$$

where, the dot product of $\overrightarrow{R_L}$ and the elite matrix simulates the Levy motion of the predator. Again, the movement of the predator was simulated as an update of the prey position.

E. Butterfly Optimization Algorithm (BOA)

BOA was proposed by Arora et al. in 2019. It is inspired by the survival and reproduction behavior of butterflies in nature. Butterflies rely on their senses to identify food sources. In the BOA, if every butterfly can produce an odor, the odor will spread out. The smell of each butterfly is related to its fitness. That is, as the position of the butterfly changes, so does its fitness. When a butterfly smells something else, it moves toward it, in what's called a "global search." The other is that when a butterfly doesn't feel more fragrant than it does, it flies at random, in what's called a "local search." Fragrance is indicated by stimulus intensity, which is shown in Eq. (33).

$$f=cI^{\alpha} \quad (33)$$

where, the sensory factor is denoted by c ; stimulus intensity is denoted by I ; α is a power; I has to do with the fitness of butterflies.

In the global search, the butterfly will move towards the optimal solution g^*, as shown in Eq. (34).

$$x_i^{t+1} = x_i^t + (r^2 * g^* - x_i^t) * f_i \quad (34)$$

where, the solution vector of the i -th butterfly in the t -th iteration is denoted by x_i^t ; the optimal solution so far is denoted by g^* ; the fragrance of the i -th butterfly is denoted by f_i ; r is between $[0,1]$.

The local search stage can be expressed as Eq. (35).

$$x_i^{t+1} = x_i^t + (r^2 * x_j^t - x_k^t) * f_i \quad (35)$$

k -th and j -th butterflies are randomly selected from the space of solutions are denoted by x_k^t and x_j^t . Both kinds of searches occur during the butterfly's foraging, which can be solved by p . This method can be used for the transformation of global search and local search. Each iteration is based on the comparison of r and p randomly generated by Eq. (35) to consider whether global or local search is needed, $r = rand(0,1)$.

F. Cuckoo Search Algorithm (CS)

In 2009, Yang et al. proposed a CS algorithm with relatively few parameters, relatively simple operation, easy implementation and strong optimization ability. To simulate the living habits of cuckoos, the CS algorithm assumes the following ideal conditions:

(1) Each cuckoo lays one egg at a time and randomly selects a nest to store it in.

(2) During the searching for nests, the nest with the best eggs is saved for the next generation.

(3) There are a certain number of nests available, and let's say the probability of other eggs being found in the nest is p , $p \in [0,1]$. If other exotic eggs are found, the owner will build a new nest.

According to the above three state assumptions, the location and route of rhododendron optimization search are modified, and the calculation is shown in Eq. (36).

$$x_i^{t+1} = x_i^t + \alpha \oplus L(\lambda), i = 1, 2, ..., n \quad (36)$$

where, the nest position of the i -th bird's nest in t -th generation is denoted by x_i^t , a point-to-point multiplication is denoted by $\oplus$, the step size control quantity is denoted by α , the random search path is denoted by $L(\lambda)$, and the random step size adopts the $Levy$ distribution.

$$L(s, \lambda) s^{-\lambda}, (1 < \lambda \le 3) \quad (37)$$

where, the random step obtained by $Levy$ flight is denoted by s .

The CS algorithm steps are as follows.

Step 1: Firstly, determine the initialization of the objective function $f(X)$, where $X = (x_{1,...,} x_d)^T$, and randomly generate the initial position $X_i (i = 1, 2, ..., n)$ of n nests, set p , Max_Iter , and the population size, etc.

Step 2: Select the fitness function and determine the corresponding objective function value of each nest position, so as to get the best value.

Step 3: Eq. (36) updates the position and situation of other nests and records the value of the best function of the previous generation.

Step 4: Compare the existing position function value with the previous generation of the best function value, if it is better, adjust the current optimal value.

Step 5: After position update, compare $r \in [0,1]$ and p with random numbers. If $r > p$, change x^{t+1} randomly; if not, keep it unchanged. The best position of the bird's nest was denoted as y_i^{t+1} .

Step 6: If the maximum number of iterations or minimum error requirement is not reached, return to Step 2; If not, proceed to the next step.

Step 7: Output the global optimal location.

G. Firefly Algorithm (FA)

FA is a swarm intelligence optimization algorithm proposed by Yang based on the phototaxis among fireflies [16]. Fireflies that emit relatively strong light attract their neighbors that emit less light, and at the same time, the intensity decreases as the distance increases. Each firefly has a sensing range. If it can't sense a brighter firefly partner, it flies randomly. Firefly brightness is represented by objective function value, and firefly position is represented by feasible solution. The process of firefly approaching to the brightest firefly is to complete the optimization process of the treatment function. The disadvantage of FA is that the time is relatively long, and the accuracy is not high. The four elements of FA are luminescence intensity, the distance between two fireflies, appeal and firefly location update.

(1) Luminous intensity

If the continuous optimization problem to be solved is a minimization problem, the calculation of the luminous intensity I_i of the firefly at the spatial position x_i is shown in Eq. (38). If the maximum problem is solved, the luminous intensity I_i of the firefly is shown in Eq. (39), where the value of the target function for x_i is denoted by g $f(x_i)$.

$$I_i = \frac{1}{f(x_i)} \quad (38)$$

$$I_i = f(x_i) \quad (39)$$

(2) The distance between two fireflies

The distance r_{ij} between the i -th firefly and the j -th firefly is shown in Eq. (40), where $x_{i,k}$ is the k -th component of the space coordinate x_i of the i -th firefly, and d represents the dimension of the problem.

$$r_{ij} = \| x_i - x_j \| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \quad (40)$$

(3) Attractive

The attraction of a firefly is shown in Eq. (41), where r represents the distance between the firefly and another point, β_0 is a constant to represent the maximum attraction, and γ represents light absorption coefficient.

$$\beta(r) = \beta_0 e^{-\gamma r^2} \quad (41)$$

(4) Firefly location update

As shown in Eq. (42), α represents the random term coefficient, and $rand$ is between $[0,1]$.

$$x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2}(x_j - x_i) + \alpha(rand - 0.5) \quad (42)$$

With the progress of time, many scholars have improved the firefly algorithm. For example, Fister et al. used consternation to represent small fireflies to improve performance and avoid any stagnation in the searching process [17].

III. Natural Heuristic Algorithms to Solve Feature Selection Problem

A. S-type Transfer Function

The transfer function is one of the most efficient ways to convert a continuous algorithm to binary because it can convert continuity to discrete without changing the original structure of the algorithm. Most scholars use S-type transfer functions to transform continuous optimization algorithm into binary version. Therefore, this paper carries out research based on S-type transfer function. Table 1 gives the mathematical expressions of the s-type transfer function family. Fig.1 shows the schematic of the transfer functions.

B. K Nearest Neighbor Classifier (KNN)

K-nearest neighbor is a common multivariate classification algorithm. Its basic principles are described as follows. When most of the K closest samples belong to a certain type in a specific feature space, then the sample also belongs to that type. In the experiment, KNN is used to classify tasks, calculate the Euclidean distance D_E between the training data set and testing data set, and determine K types of the nearest samples, as shown in Eq. (43).

$$D_E = \sqrt{\sum_{i=1}^{K}(Train_{Fi} - Test_{Fi})^2} \quad (43)$$

C. Fitness Function

Feature selection is a kind of binary optimization problem, in which feature subset is composed of 1 and 0, represented by binary vector, where 1 represents feature selected and 0 represents feature not selected. The goal of feature selection is to select as few features as possible while maintaining high classification accuracy. So feature selection is a relatively complex multi-objective problem. In this paper, the fitness function shown in Eq. (44).

$$fitness = h_1 \gamma_R(D) + h_2 \frac{|M|}{|N|} \quad (44)$$

where, the classification error rate corresponding to the feature subset selected by the classifier is denoted by $\gamma_R(D)$, the number of features selected is denoted by $|M|$, the total number of features is denoted by $|N|$, h_1 and h_2 are the two weight coefficients reflecting the classification rate and length of the subset, satisfying $h_1 + h_2 = 1$. Due to the need for an accurate classification model, the classification accuracy is assigned with a high inertial weight. In this paper, h_1 and h_2 are set to 0.99 and 0.01 respectively.

D. FS Architecture Based on Natural Heuristic Algorithm

Feature selection, as an important stage in classification problems, is a challenging process in dealing with high-dimensional searching spaces. The goal of FS is to select the least representative feature set from the original feature set on the premise of ensuring the classification accuracy of the classifier. In other words, FS is a dimension reduction process, in which the classification accuracy of the classifier is used to verify the effectiveness of dimension reduction. FS architecture based on natural heuristic algorithm is shown in Fig. 2.

IV. Simulation Experiments and Result Analysis

A. Selection of Experimental Datasets

Twenty-one data sets are selected from UCI for classification research. These data sets have different instances, feature numbers and categories. The advantages and disadvantages of different natural heuristic algorithms in various data sets can be seen from different perspectives. Table 2 lists the details of these data sets. In this study, the KNN algorithm was used to calculate the classification accuracy in the fitness function, because KNN was proved to be faster and simpler, where K=5. The experiments were repeated 30 times with different random seeds. In addition, to prevent excessive fitting, a 50% cross validation method is adopted. The data set is divided into training sets and test sets.

Table 1. The S-type transfer functions

S-type transfer function	Equation
S1	$T(x) = \frac{1}{1+e^{-2x}}$
S2	$T(x) = \frac{1}{1+e^{-x}}$
S3	$T(x) = \frac{1}{1+e^{(-\frac{x}{2})}}$
S4	$T(x) = \frac{1}{1+e^{(-\frac{x}{3})}}$

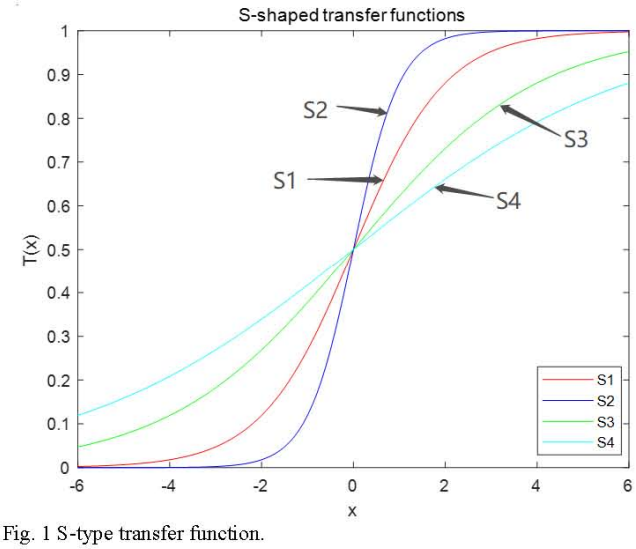


Fig. 1 S-type transfer function.

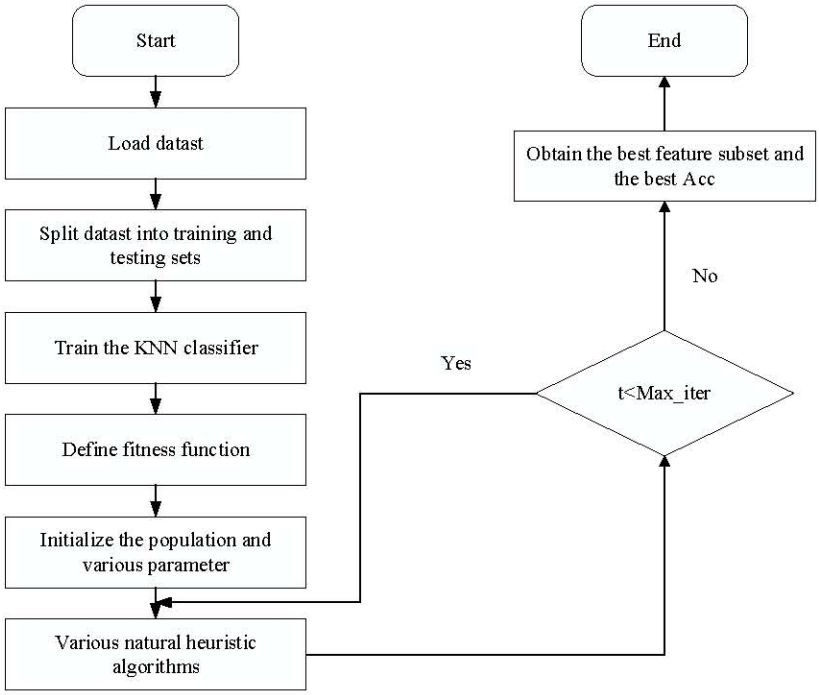


Fig. 2 FS architecture based on natural heuristic algorithm.

TABLE 2. 21 DATA SETS USED IN THE SIMULATION EXPERIMENTS

Number	Datasets	Features	Instances	Classes
1	Algerian Forest Fires	12	244	2
2	Clean1	167	476	2
3	Climate Model Simulation Crashes	18	540	2
4	Connectionist Bench	60	208	2
5	Diabetic Retinopathy Debrecen	20	1151	2
6	Electrical Grid Stability Simulated	14	10000	2
7	Forest type mapping	27	326	4
8	Heart	13	303	2
9	Im	270	1000	20
10	Ionosphere	34	351	2
11	Page Blocks Classification	10	5473	5
12	Parkinson Disease	754	756	2
13	Pima Indians Diabetes	8	768	2
14	Planning Relax	13	182	2
15	QSAR biodegradation	41	1055	2
16	Seeds	7	210	3
17	Semeion	256	1593	2
18	Spambase	57	4601	2
19	Waveform	21	5000	3
20	Wine	13	178	3
21	Zoo	17	101	6

In the first iteration, 80% of the eigenvectors were used for training and the remaining 20% for testing. Next, 20% feature vectors are used for testing, and the remaining 80% feature vectors are used for training set. Repeat the process until all eigenvectors are tested. Finally, average statistical measurements were collected over 30 separate runs and displayed as final results. All experiments used MATLAB R2018b, running on Intel Core i5-9300H machine, CPU 2.40 GHz, RAM 8GB, Windows 10 operating system. In this study, the population of each algorithm was set as 10, the maximum iteration number was set as 100, and the common parameters of the seven algorithms remained consistent. The dimension of the searching space is the same as the total number of features. According to previous studies by scholars [18], the classifier has the best classification performance when the hyper parameter h_1 is set to 0.99.

B. Feature Selection Performance Evaluation

When evaluating and interpreting the results of a feature selection problem, there are metrics. These measures were fitness value, classification accuracy, and average selection size. Eq.(45)-(50) show the calculation methods of average classification accuracy, mean value of selected feature number, mean value of fitness and standard deviation in turn.

$$Mean\_accuracy = \frac{1}{30}\sum_{i=1}^{30} Accuracy_i \tag{45}$$

where, $Mean\_accuracy$ represents the average classification accuracy obtained, where the algorithm is run independently for 30 times. $Accuracy_i$ represents the classification accuracy rate obtained by each run. The classification accuracy is calculated as follows:

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} match(Pl_i, Al_i) \tag{46}$$

where, the number of test set points is denoted by N; Pl_i is the class label of prediction class data point i; Al_i is the actual class in the labeled data, that is, the reference class

label of i, and $match(Pl_i, Al_i)$ is a comparison discriminant function. If $Pl_i == Al_i$, $match(Pl_i, Al_i) = 1$ or $match(Pl_i, Al_i) = 0$.

$$Mean\_feature = \frac{1}{30}\sum_{i=1}^{30} feature_i \tag{47}$$

where, the average value of selected feature number obtained by independent operation of the algorithm for M times is denoted by $Mean\_feature$, and $feature_i$ is the value of selected feature number obtained by each operation.

$$Mean\_fitness = \frac{1}{30}\sum_{i=1}^{30} fitness_i \tag{48}$$

where, the average fitness of M times of independent operation of the algorithm is denoted by $Mean\_fitness$, and f_i is the best fitness of each operation. The fitness value is shown in Eq. (49).

$$fitness = 0.99*(1-Accuracy) + 0.01*\frac{|Selected\ features \quad Count|}{|Total\ features \quad Count|} \tag{49}$$

where, $Accuracy$ is the classification accuracy rate.

$$Std\_fitness = \sqrt{\frac{1}{30}\sum(fitness_i - Mean\_fitness)^2} \tag{50}$$

where, the standard deviation of fitness value is denoted by $Std\_fitness$, $fitness_i$ is the fitness value obtained at the i-th time,

C. Simulation of Feature Selection Based on Natural Heuristic Algorithms

Table 3-5 compares the simulation results of 7 natural heuristic algorithms for 21 different UCI data sets. Table 3 shows the 21 data mean fitness and standard deviation of accuracy under 7 natural heuristic algorithms. Table 4 and Table 5 respectively show the average number of selected features and running time. In the above tables, the best results are highlighted with deepen. In Table 3, MPA achieved the highest average fitness value in most data sets (16 data sets), followed by FA (6 data sets), and SMA achieved an optimal performance. In Table 4, SMA wins by a wide margin, finding the minimum eigenvalues on all 20 datasets. In Table 5, SMA has an advantage in operation time. By drawing the convergence curves and the accuracy box plots of the optimal classification accuracy calculated by KNN classifier, the difference between the seven different natural heuristic algorithms can be shown more intuitively and vividly. The convergence curves of the 7 natural heuristic algorithms on 21 data sets are shown in Fig. 3, where the abscissa represents the iteration times of the algorithm, and the ordinate represents the average accuracy value of each algorithm after 30 times of independent execution. MPA has the best convergence effect in most cases, followed by FA and CS respectively. The boxplot of the accuracy values is shown in Fig. 4.

TABLE 3. 21 DATA MEAN FITNESS AND STANDARD DEVIATION OF ACCURACY

Datasets	Measure	SMA	WOA	HHO	MPA	BOA	CS	FA
Algerian	AVG	0.0037	0.0098	0.0159	**0.0016**	0.0171	0.0024	0.0024
	STD	**0.0064**	0.0114	0.0126	0.0152	0.0146	0.0186	0.0179
Clean1	AVG	0.0837	0.0807	0.0782	**0.0398**	0.0892	0.0583	0.0690
	STD	0.0144	0.0142	0.0131	0.0203	**0.0123**	0.0168	0.0131
Climate	AVG	0.0549	0.0617	0.0641	0.0376	0.0660	0.0421	**0.0369**
	STD	0.0118	0.0156	0.0145	0.0122	0.0107	**0.0100**	0.0110
Connectionist	AVG	0.1149	0.1238	0.1326	**0.0464**	0.1279	0.0835	0.1095
	STD	0.0339	0.0342	0.0311	0.0409	0.0329	**0.0277**	0.0316
Diabetic	AVG	0.2928	0.2957	0.3032	**0.2777**	0.3021	0.2821	0.2801
	STD	**0.0084**	0.0119	0.0134	0.0157	0.0102	0.0159	0.0154
Electrical	AVG	**0.0013**	0.0590	0.1212	**0.0013**	0.0643	0.0885	0.0172
	STD	**0.0000**	0.0911	0.1037	0.1015	0.0735	0.0316	0.0375
Forest	AVG	0.1014	0.1016	0.1063	**0.0900**	0.1118	0.0946	0.0915
	STD	0.0086	0.0080	**0.0074**	0.0097	0.0086	0.0091	0.0092
Heart	AVG	0.2199	0.2246	0.2382	0.1875	0.2297	0.1909	**0.1866**
	STD	**0.0232**	0.0252	0.0341	0.0240	0.0245	0.0378	0.0329
Im	AVG	0.2140	0.1966	0.1925	**0.1506**	0.2092	0.1737	0.1850
	STD	**0.0110**	0.0130	0.0140	0.0196	0.0113	0.0123	0.0115
Ionosphere	AVG	0.0314	0.0452	0.0570	**0.0110**	0.0690	0.0464	0.0492
	STD	0.0206	0.0201	0.0316	0.0231	0.0167	0.0193	**0.0162**
Page	AVG	0.0458	0.0484	0.0497	0.0440	0.0497	0.0452	**0.0436**
	STD	**0.0025**	0.0030	0.0026	0.0031	0.0028	0.0037	0.0033
Parkinson	AVG	0.1827	0.2010	0.1842	**0.1504**	0.2137	0.2235	0.2188

	STD	0.0206	0.0177	0.0243	0.0148	0.0183	**0.0048**	0.0066
Pima	AVG	0.2045	0.2060	0.2132	**0.1979**	0.2141	0.1985	**0.1979**
	STD	**0.0098**	0.0107	0.0125	0.0188	0.0131	0.0200	0.0182
Planning	AVG	0.2127	0.2249	0.2322	0.1911	0.2219	0.1915	**0.1832**
	STD	**0.0211**	0.0247	0.0221	0.0331	0.0247	0.0293	0.0321
QSAR	AVG	0.1380	0.1409	0.1432	**0.1086**	0.1465	0.1134	0.1195
	STD	**0.0101**	0.0104	0.0164	0.0120	0.0118	0.0127	0.0118
Seeds	AVG	0.0084	0.0269	0.0373	**0.0029**	0.0283	0.0053	0.0037
	STD	**0.0120**	0.0242	0.0317	0.0296	0.0238	0.0348	0.0309
Semeion	AVG	0.0166	0.0146	0.0141	**0.0071**	0.0168	0.0101	0.0115
	STD	**0.0022**	0.0032	0.0036	0.0028	0.0025	0.0029	**0.0022**
Spambase	AVG	0.1093	0.0995	0.1079	**0.0814**	0.1234	0.0855	0.0950
	STD	0.0084	0.0067	0.0155	**0.0042**	0.0148	0.0213	0.0194
Waveform	AVG	0.1818	0.1704	0.1768	**0.1599**	0.1986	0.1646	0.1607
	STD	0.0081	0.0062	**0.0054**	0.0063	0.0070	0.0099	0.0131
Wine	AVG	0.0525	0.0689	0.0852	0.0328	0.0730	0.0381	**0.0320**
	STD	0.0233	0.0370	0.0543	**0.0173**	0.0246	0.0477	0.0485
Zoo	AVG	0.0246	0.0295	0.0454	**0.0032**	0.0388	0.0034	0.0033
	STD	**0.0252**	0.0254	0.0349	0.0284	0.0267	0.0401	0.0284

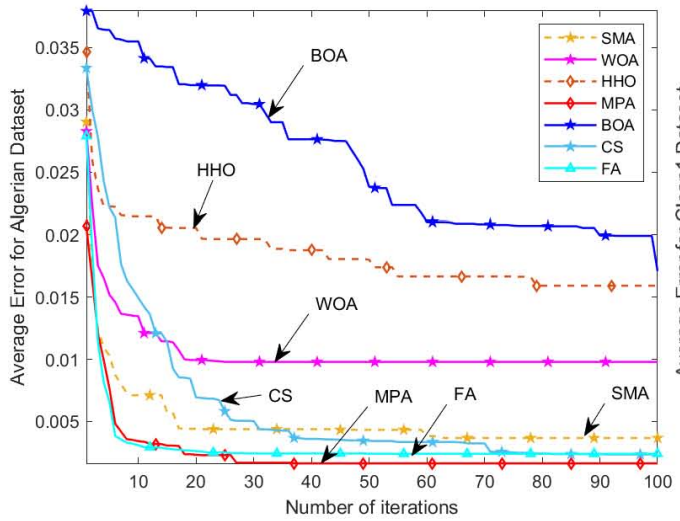
TABLE 4. Average NUMBER OF SELECTED FEATURES

Datasets	SMA	WOA	HHO	MPA	BOA	CS	FA
Algerian	**2.1000**	3.8000	2.8333	2.1333	3.4667	3.0667	3.1333
Clean1	**35.2000**	72.0333	69.7000	37.8000	57.0000	74.5333	79.7000
Climate	**4.1667**	7.2333	7.2333	6.1333	7.2667	7.2000	7.8000
Connectionist	**3.4667**	13.5667	13.2333	8.1333	14.1333	22.7000	24.6333
Diabetic	**3.5000**	6.5333	7.4333	4.5667	6.3667	6.0333	6.3667
Electrical	**1.0000**	3.5333	5.7333	**1.0000**	2.4333	4.2000	1.7333
Forest	**4.8000**	9.6333	10.3667	6.5333	9.8000	9.5667	9.6667
Heart	**3.0000**	4.4000	3.8000	3.7667	3.8333	4.2333	4.2333
Im	**108.0333**	145.8333	141.6000	110.3667	109.8333	128.6667	131.2000
Ionosphere	**2.4333**	3.0000	4.7667	3.7667	7.1333	8.5333	10.1000
Page	**2.7667**	3.8667	4.1667	3.2000	4.0000	3.3333	3.2333
Parkinson	**3.0667**	46.1333	15.6000	54.9000	131.0667	324.7000	366.1333
Pima	**2.9667**	3.6333	3.2000	3.0000	3.5333	3.1333	3.0333
Planning	**2.2000**	4.7667	4.7667	2.7333	3.4333	4.3000	4.2667
QSAR	**11.2000**	18.5667	17.5667	15.6000	15.2000	17.2667	19.1667
Seeds	**2.0000**	2.3000	2.4333	**2.0000**	2.7333	2.0333	2.0333
Semeion	111.1333	113.1667	105.1000	**83.9333**	110.6667	116.4667	126.6667
Spambase	**25.5333**	39.4333	37.9000	33.9000	28.1333	29.9667	30.5667
Waveform	**11.3667**	17.0333	16.3333	15.1333	12.0667	14.2333	14.3000
Wine	**3.3333**	3.8000	4.1333	3.4000	4.1667	4.1667	3.6667
Zoo	**4.9667**	7.6333	6.6667	5.1333	6.6000	5.4000	5.2000

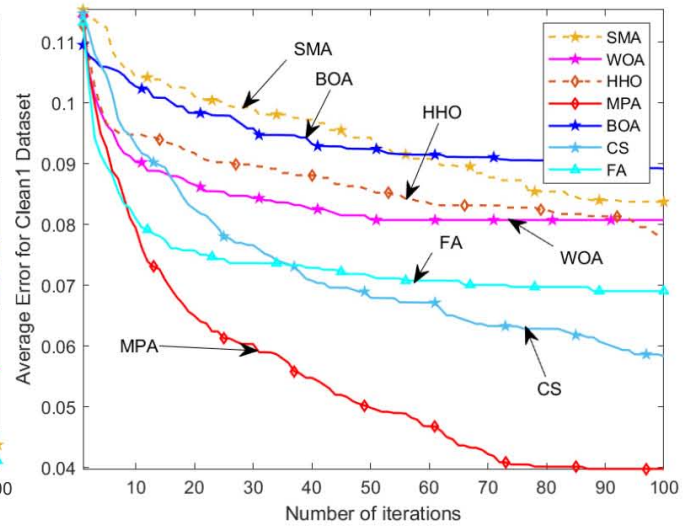
TABLE 5. Average CALCULATION TIME

Datasets	SMA	WOA	HHO	MPA	BOA	CS	FA
Algerian	**2.7937**	4.3368	7.0689	9.6928	4.8129	9.7582	4.1529
Clean1	**4.5682**	4.6438	7.6268	9.4702	4.7209	9.8446	22.8839
Climate	**4.0210**	6.3878	10.5751	14.5365	7.1445	14.8315	11.2090
Connectionist	**5.1002**	6.9053	10.6260	14.8330	7.1251	14.1597	31.5449

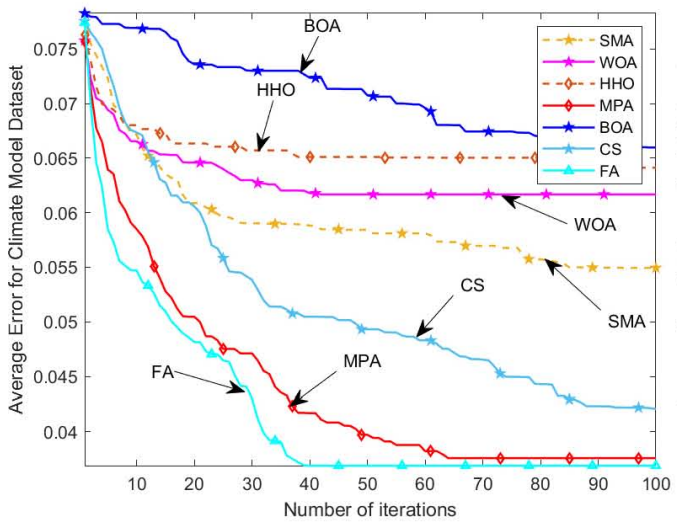
Diabetic	**3.1220**	4.7222	7.9507	10.3252	4.9786	10.1512	8.1394
Electrical	**5.1373**	17.9071	37.2978	23.8613	13.8781	38.5439	26.7692
Forest	**3.3340**	4.1264	6.7820	9.0311	4.4138	8.5376	10.1377
Heart	**2.4706**	3.8716	6.4230	8.4401	4.0582	8.4037	4.0536
Im	**5.5634**	10.4377	17.1450	17.5917	7.8739	19.9698	46.5381
Ionosphere	**2.3821**	3.7057	6.1142	8.5447	4.1482	8.0627	12.6725
Page	**4.5381**	8.4661	14.0972	17.6297	8.7565	18.4108	5.0826
Parkinson	**5.6109**	7.5516	11.2934	16.1952	14.9549	37.6721	92.1565
Pima	4.6303	7.7177	12.8742	16.6320	7.4865	16.5773	**2.3753**
Planning	**3.4716**	5.9837	9.7087	12.6162	6.2699	12.5517	5.1757
QSAR	**6.3173**	7.3760	12.0012	15.3495	7.8473	15.3607	31.8740
Seeds	2.5154	3.8918	6.4249	8.4547	3.9869	8.4461	**0.9993**
Semeion	**6.7993**	16.4154	25.8271	27.2894	15.4411	36.1279	88.1108
Spambase	**12.4977**	33.0733	53.6999	57.3135	19.7129	54.3682	116.1491
Waveform	**8.8461**	18.9628	31.3375	36.8335	13.7289	35.4683	28.9533
Wine	**2.7895**	3.9199	6.3622	8.5347	4.2136	8.4618	3.5489
Zoo	**3.3819**	3.8887	6.5454	8.4232	4.1947	8.3315	4.9387



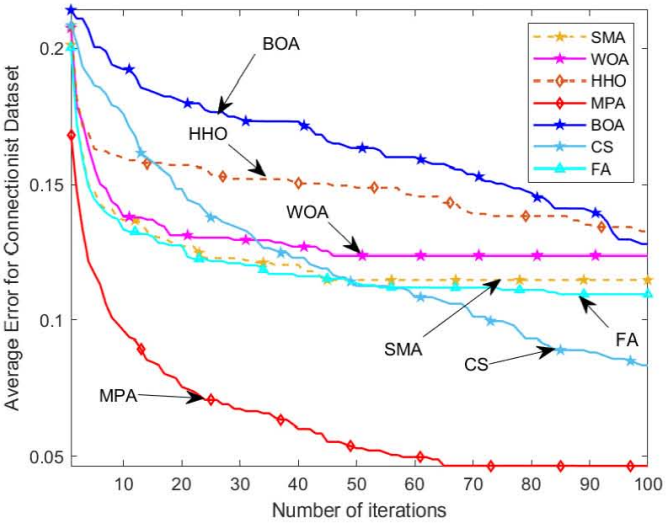
(a) Algerian



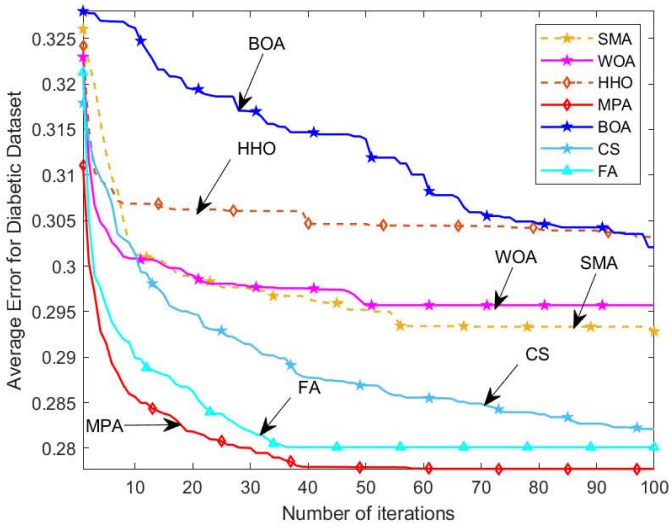
(b) Clean1



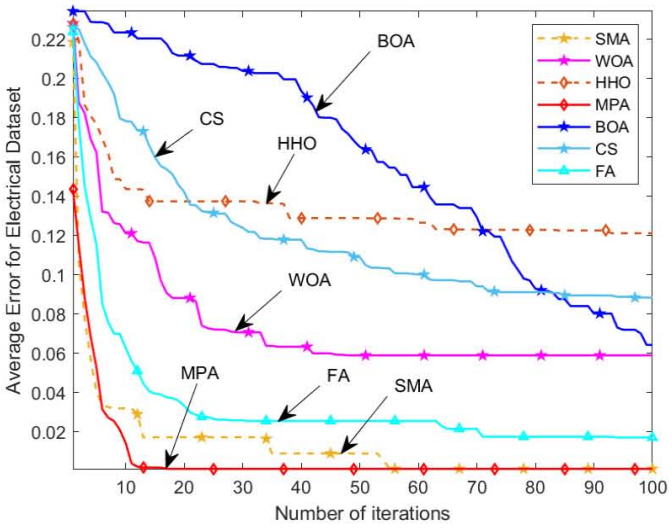
(c) Climate



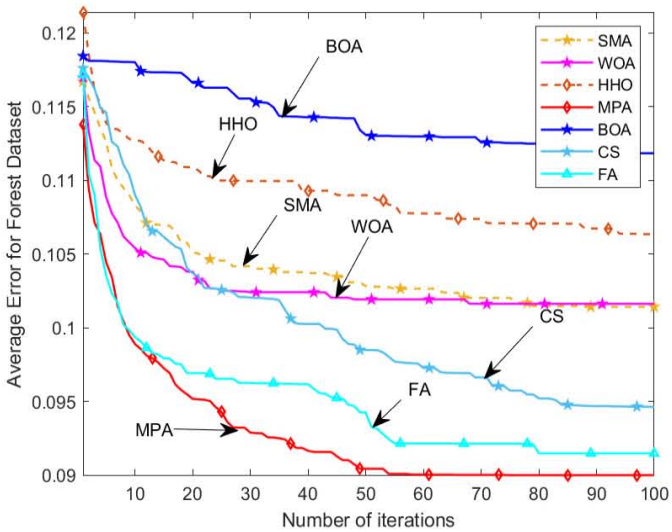
(d) Connectionist



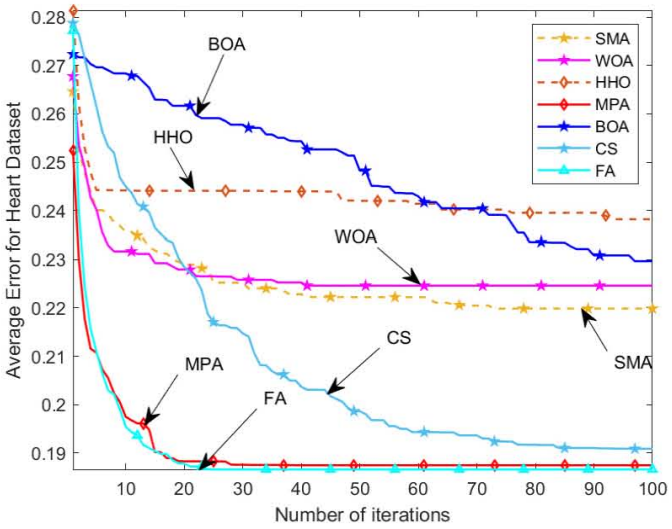
(e) Diabetic



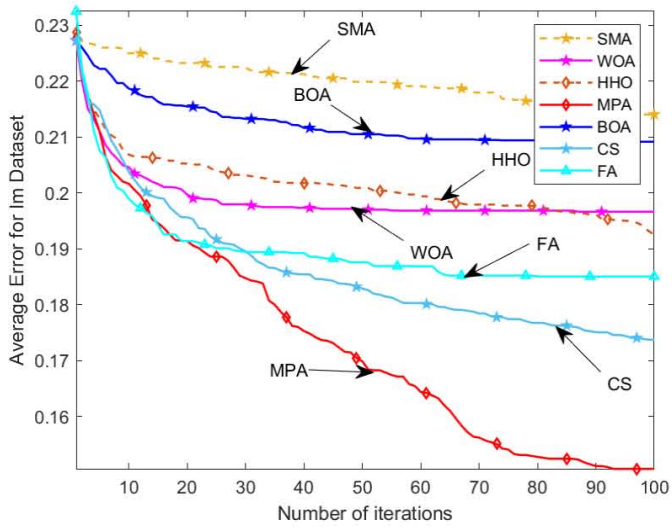
(f) Electrical



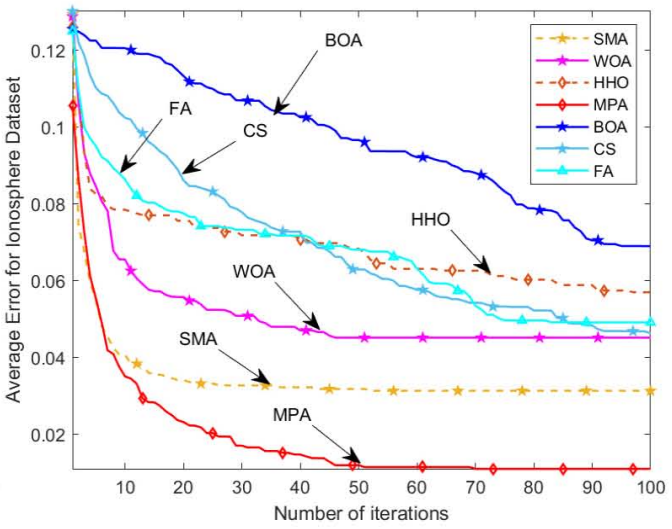
(g) Forest



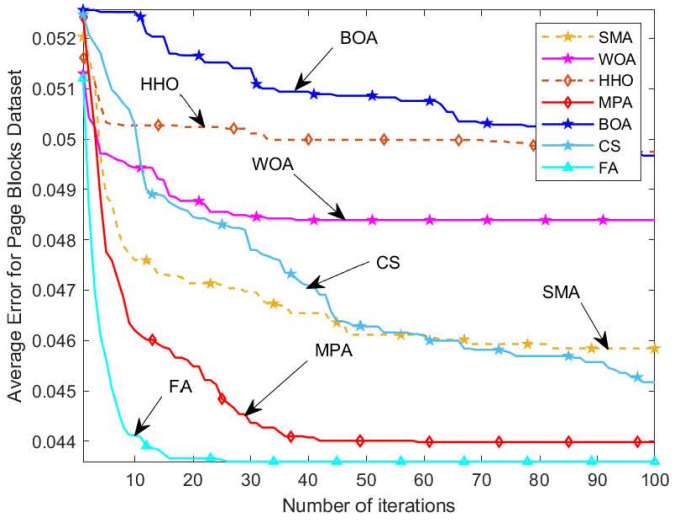
(h) Heart



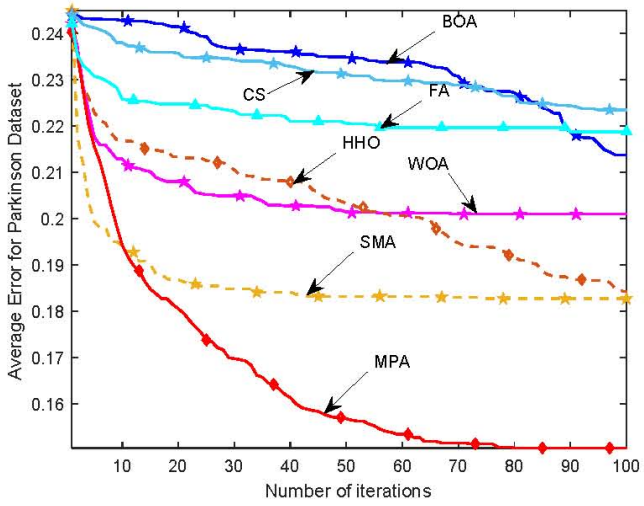
(i) Im



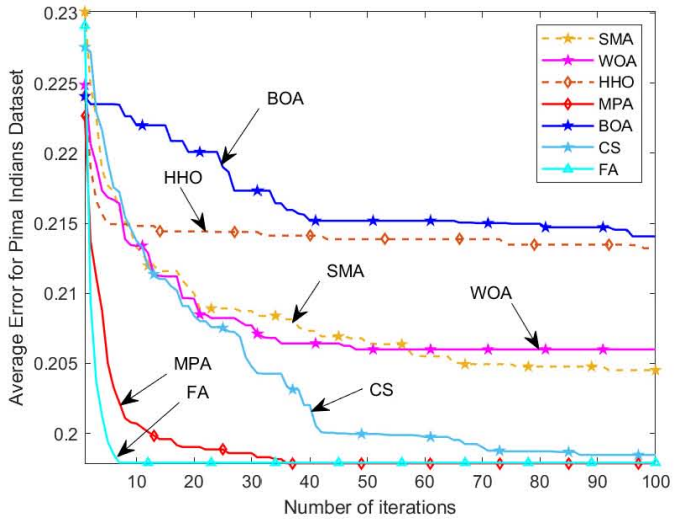
(j) Ionosphere



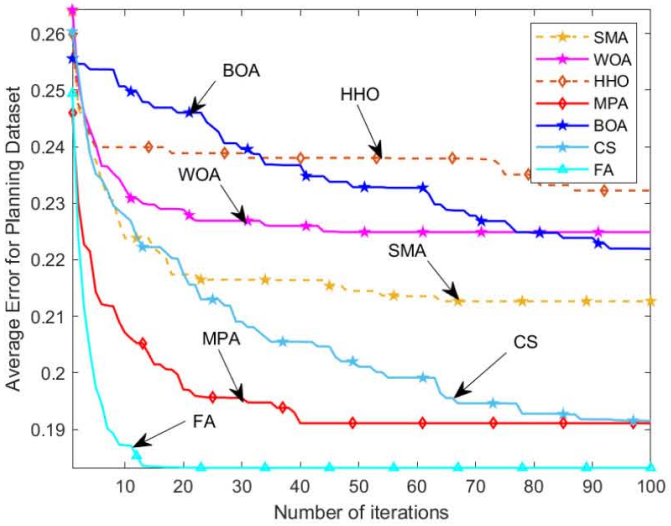
(k) Page



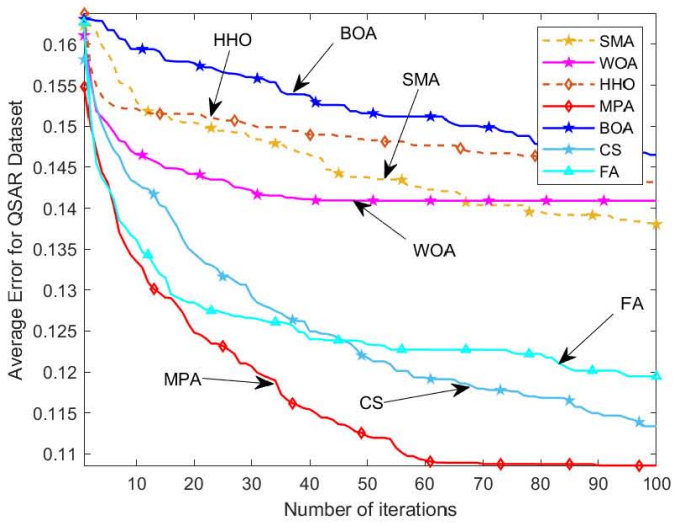
(l) Parkinson



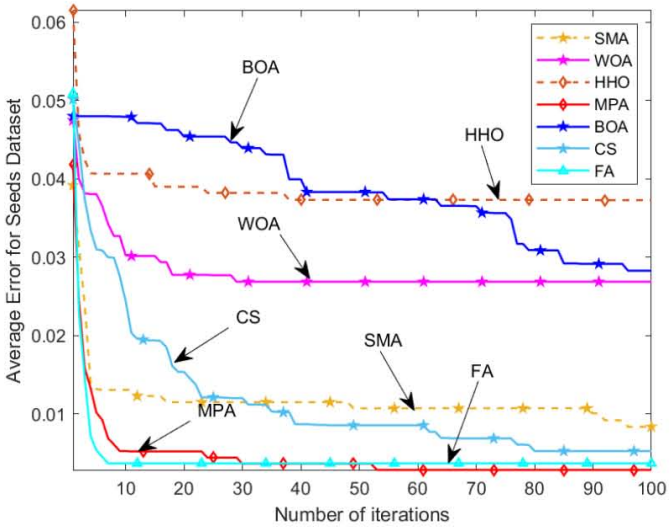
(m) Pima



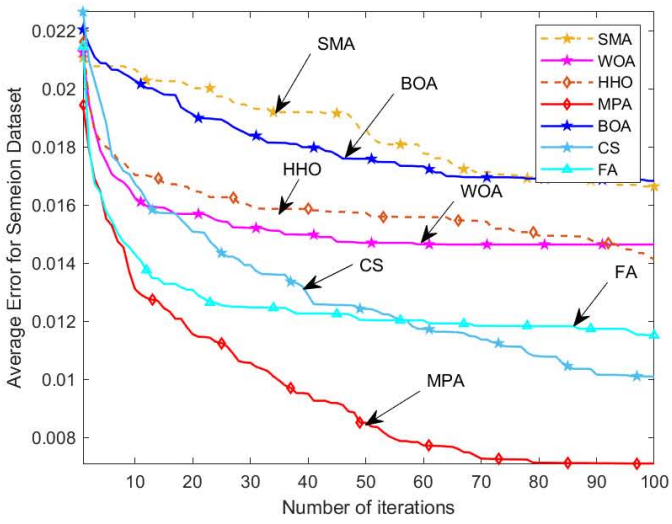
(n) Planning



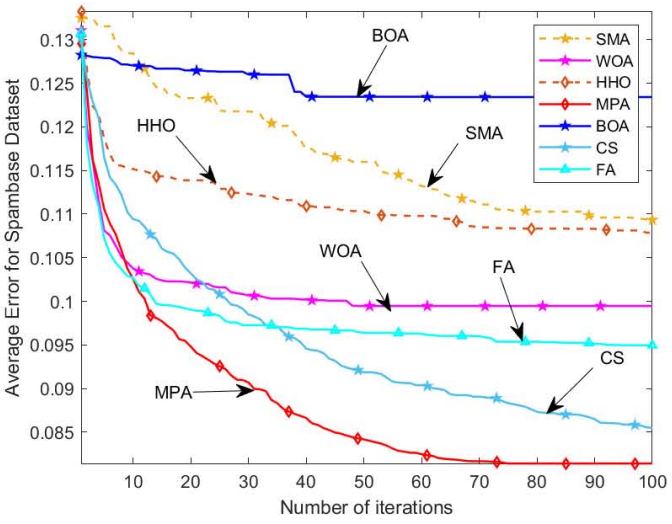
(o) QSAR



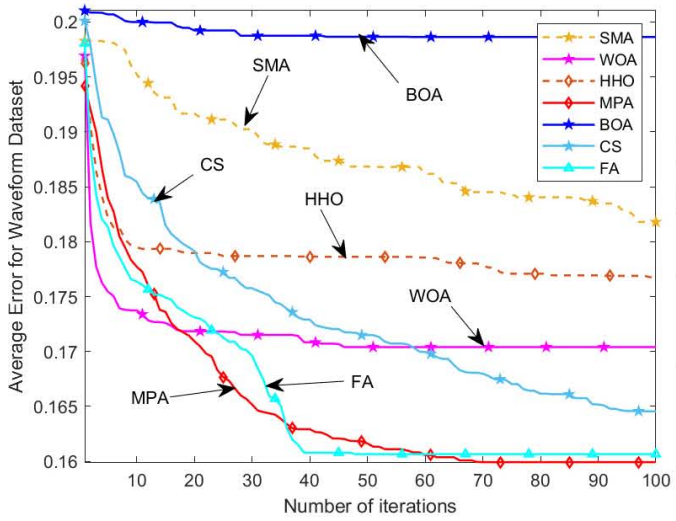
(p) Seeds



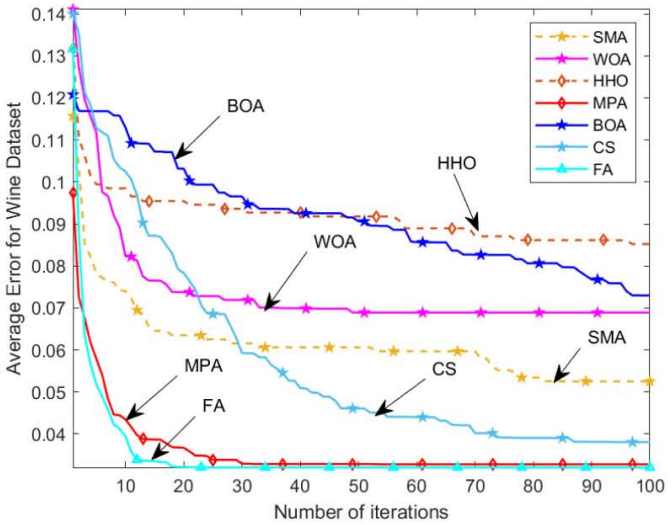
(q) Semeion



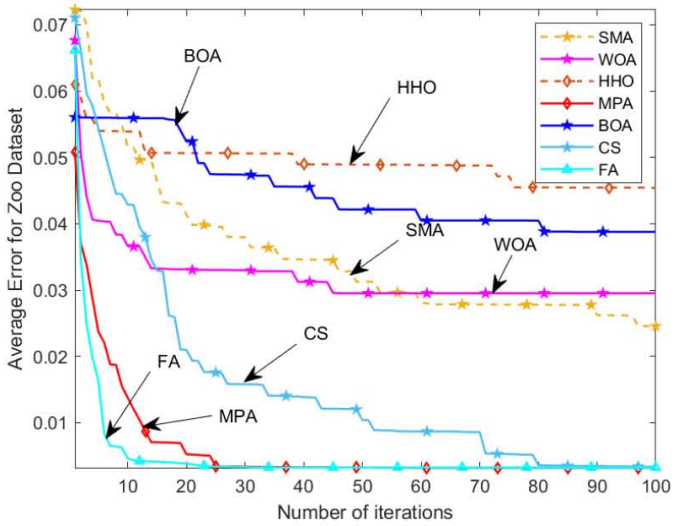
(r) Spambase



(s) Waveform

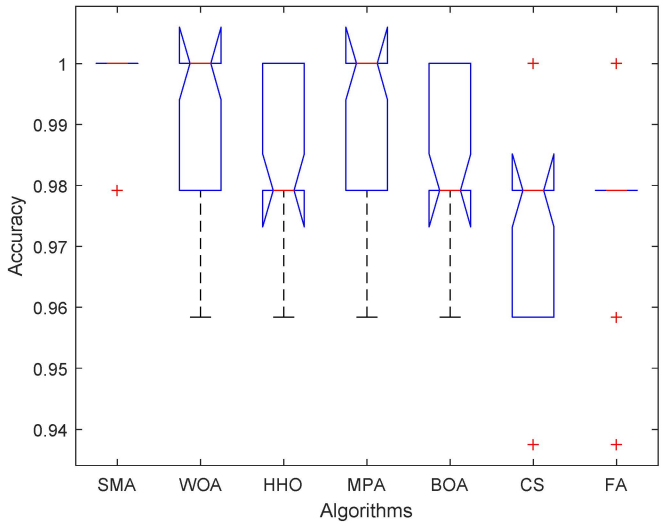


(t) Wine

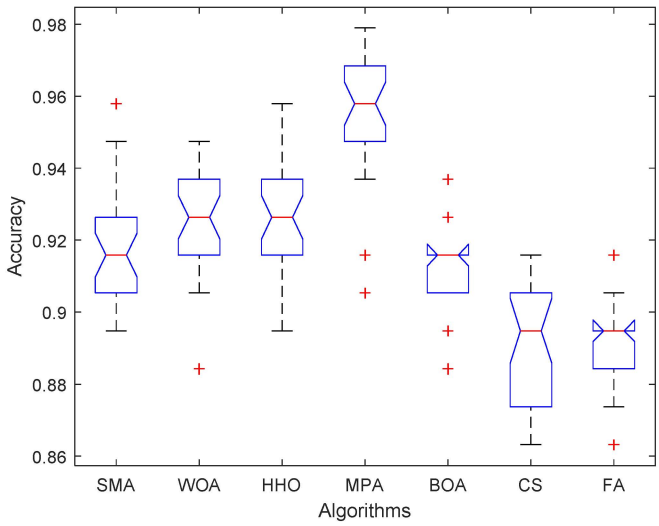


(u) Zoo

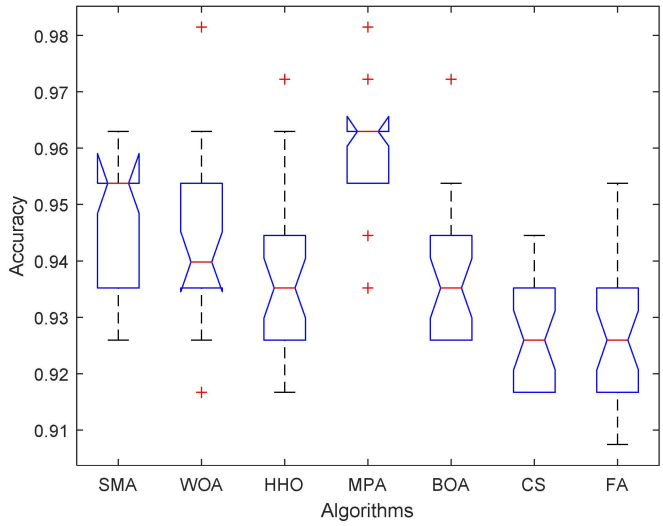
Fig. 3 Convergence curves of seven natural heuristic algorithms on 21 data sets.



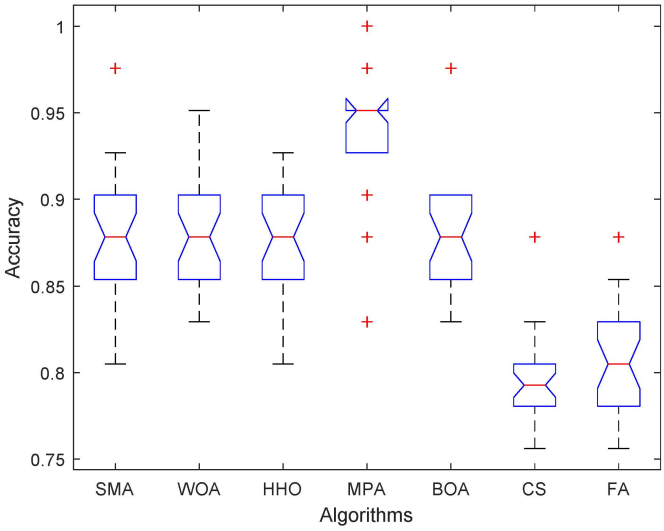
(a) Algerian



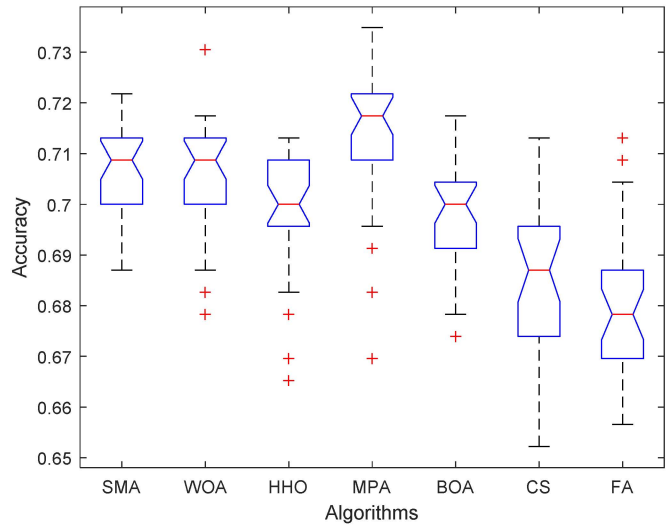
(b) Clean1



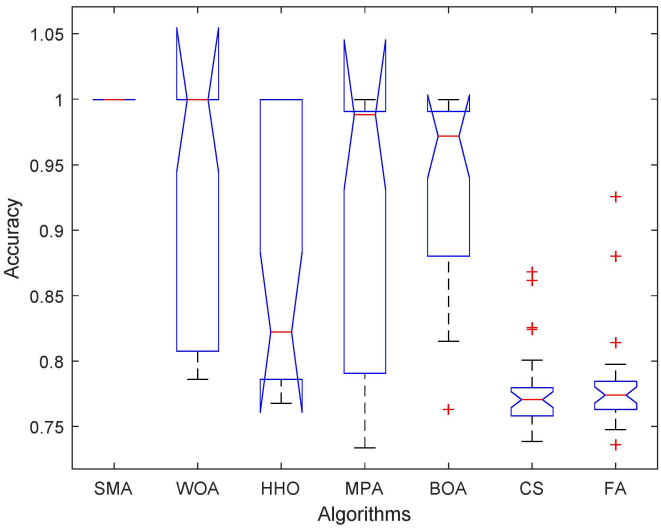
(c) Climate



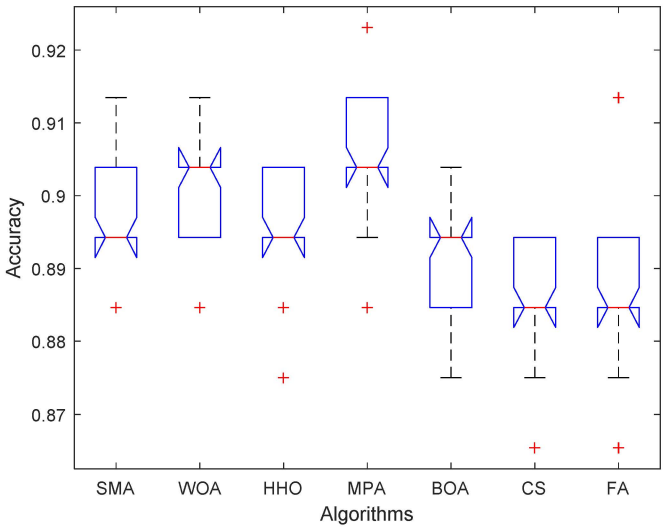
(d) Connectionist



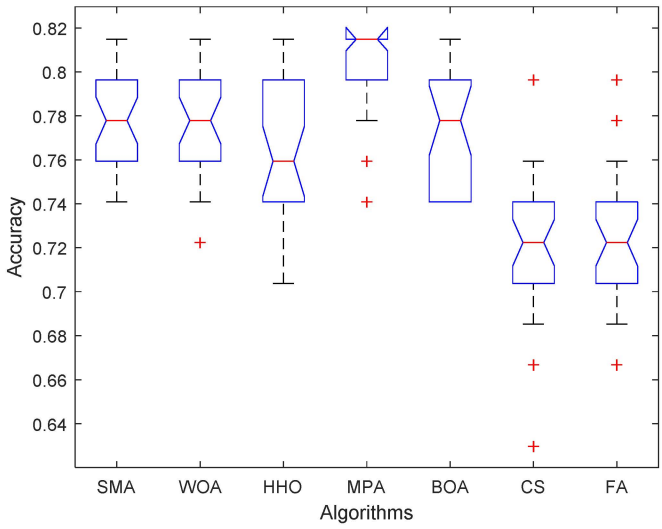
(e) Diabetic



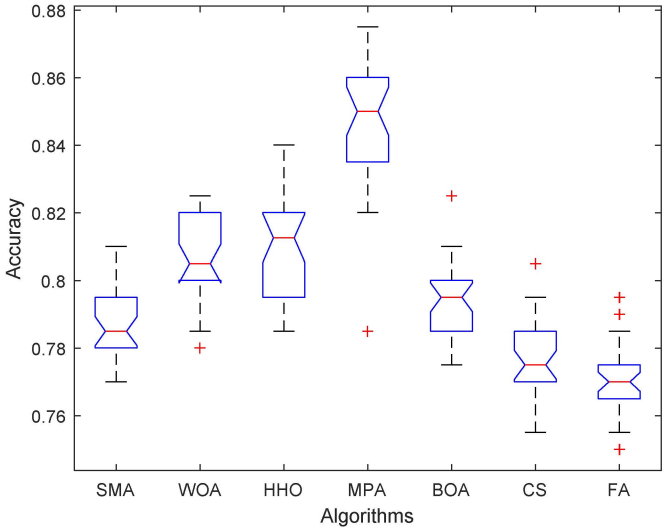
(f) Electrical



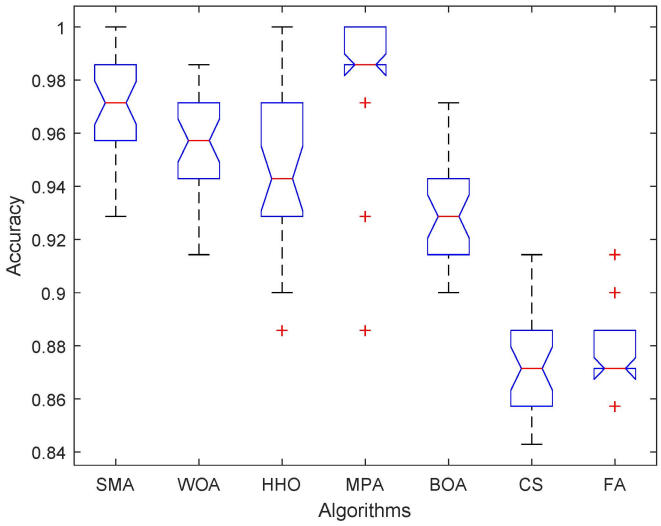
(g) Forest



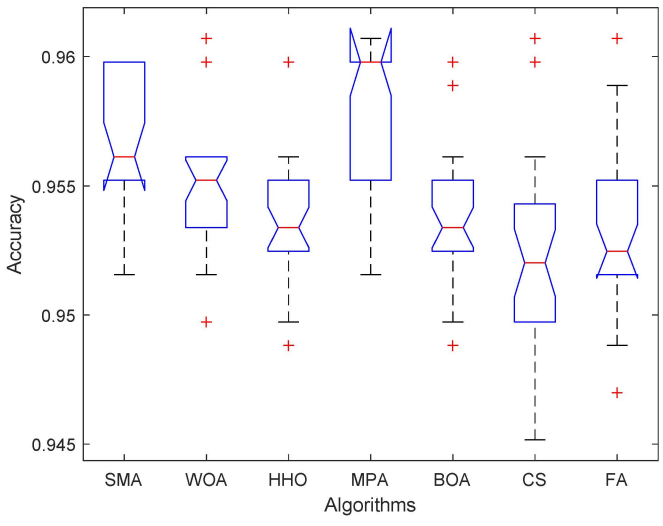
(h) Heart



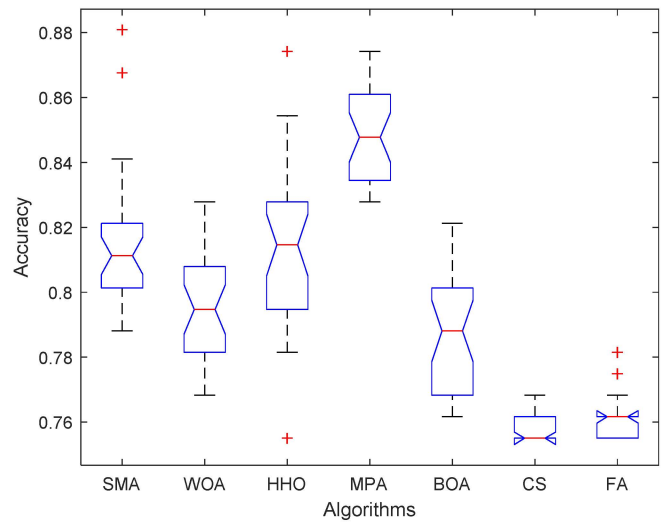
(i) Im



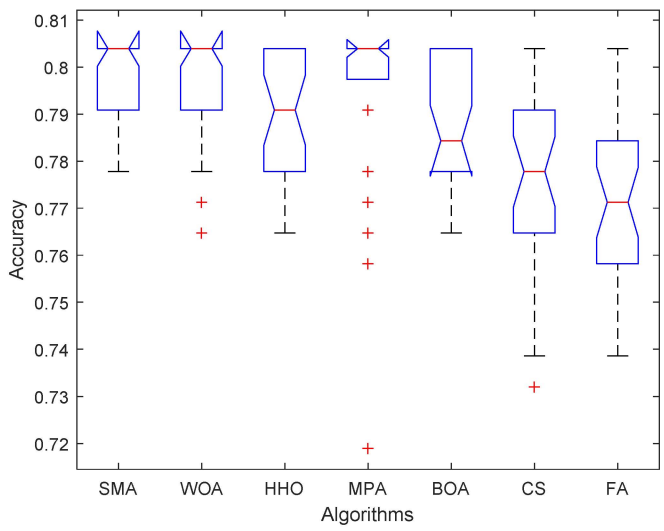
(j) Ionosphere



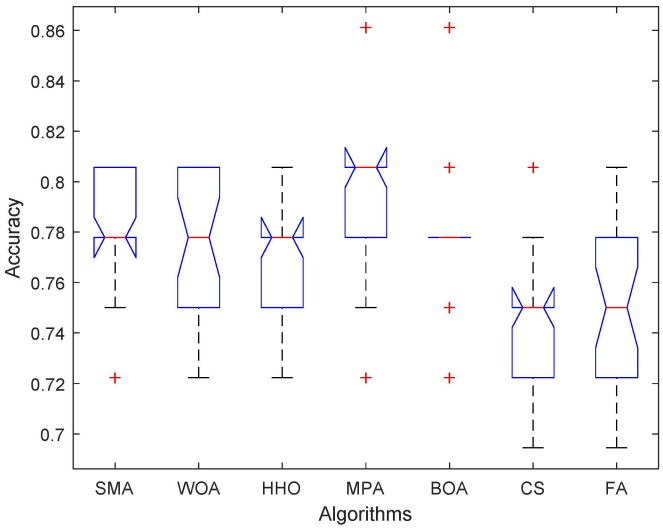
(k) Page



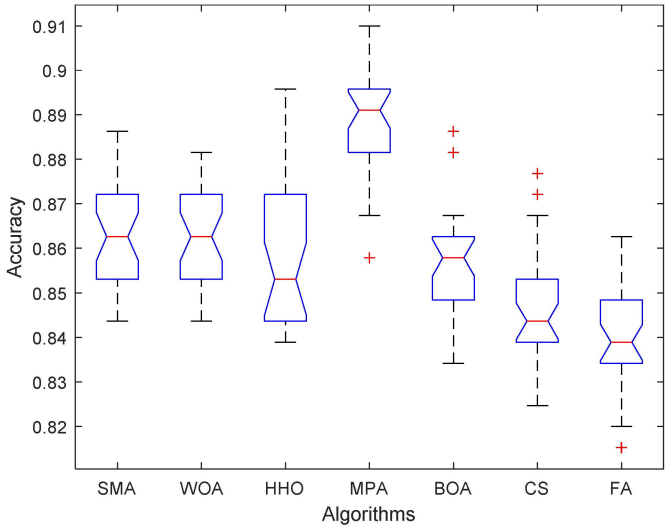
(l) Parkinson



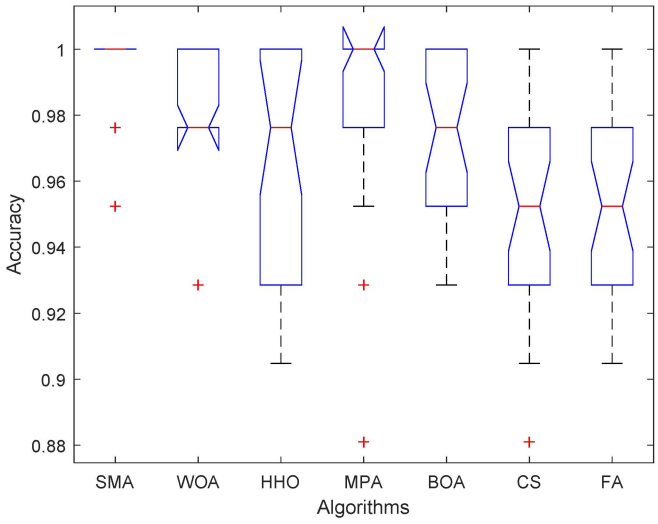
(m) Pima



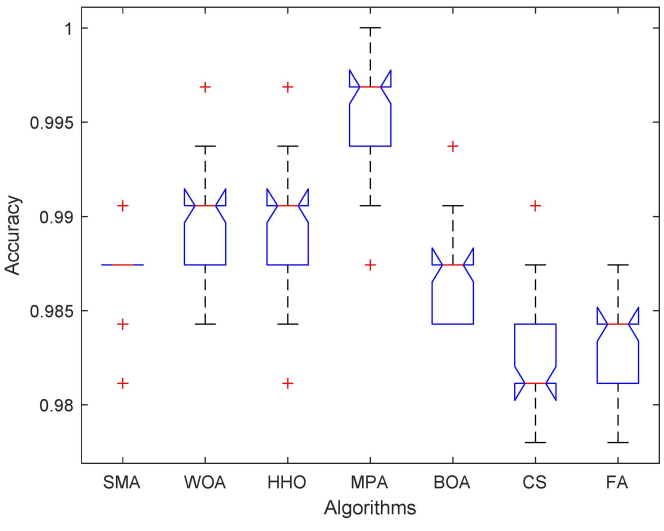
(n) Planning



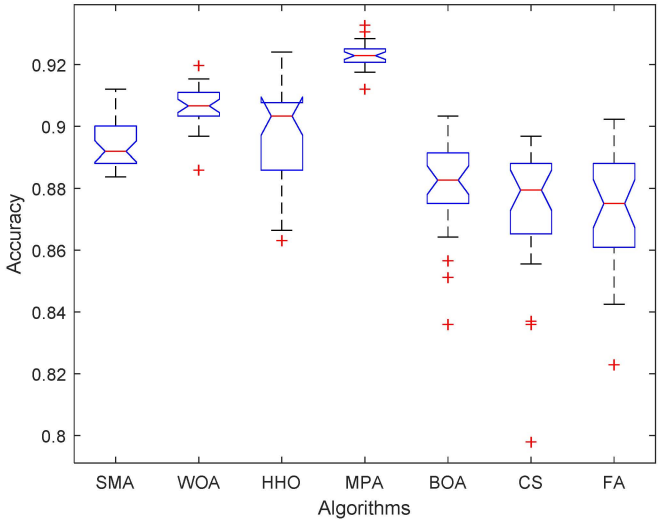
(o) QSAR



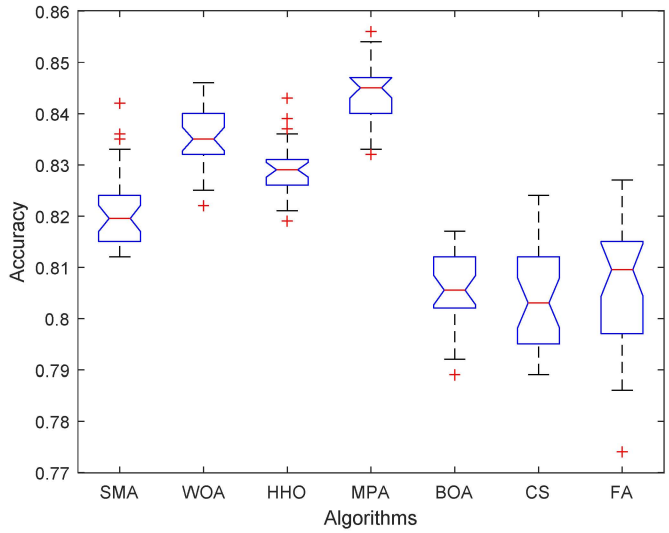
(p) Seeds



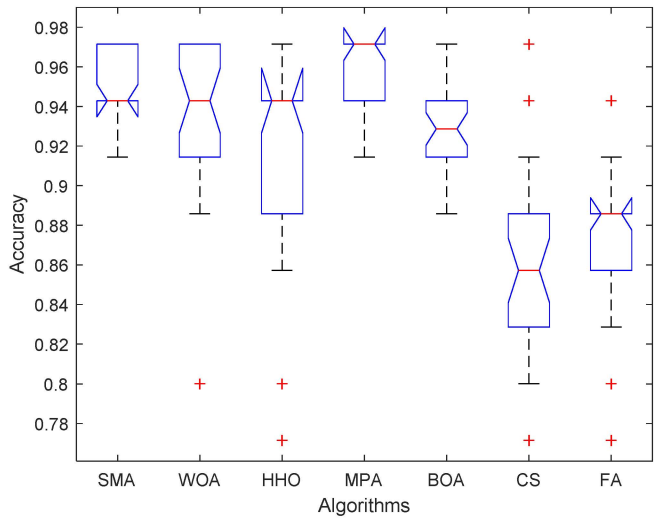
(q) Semeion



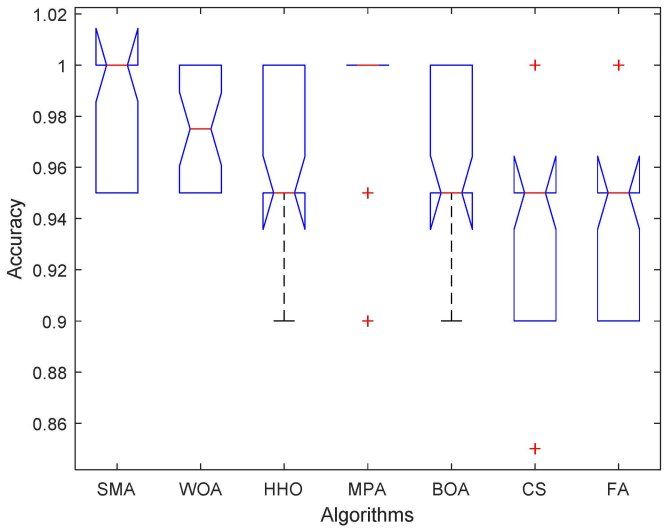
(r) Spambase



(s) Waveform



(t) Wine



(u) Zoo

Fig. 4 Boxplots of adaptive values of seven natural heuristic algorithms.

V. Conclusion

Based on the wrapper feature selection method, this paper compares seven natural heuristic algorithms, including SMA, WOA, HHO, MPA, BOA, CS and FA. The convergence curves and the boxplots of accuracy of seven natural heuristic algorithms on 21 data sets are given. The results show that MPA has the highest average fitness value in most data sets (16 data sets), followed by FA algorithm (6 data sets), and SMA has the best performance. SMA can find the minimum eigenvalues in all 20 data sets and has an advantage in computing time. Combined with the proposed natural heuristic algorithms, the results were evaluated according to the mean and standard deviation of fitness, the number of selected features and the running time, and the optimal value was represented with deepen. In this paper, it is found that MPA has the best average fitness value, while SMA has the advantage in eigenvalue and operation time. Both algorithms have their own advantages. The mean and standard deviation data of fitness, the number of selected features, the running time data, the convergence curve and the boxplot of accuracy obtained by the 7 natural heuristic algorithms through 21 data sets are of great reference value for subsequent research.

References

[1] A. Sankalap, and A. Priyanka, "Binary Butterfly Optimization Approaches for Feature Selection," *Expert Systems with Applications*, vol. 116, pp. 147-160, 2019.

[2] Q. Hu, A. Shuang, and D. Yu, "Soft Fuzzy Rough Sets for Robust Feature Evaluation and Selection," *Information Sciences*, vol. 180, no. 22, pp. 4384-4400, 2010.

[3] M. Tubishat, M. Abushariah, N. Idris, and I. Aljarah, "Improved Whale Optimization Algorithm for Feature Selection in Arabic Sentiment Analysis," *Applied Intelligence*, vol. 49, no. 3, pp. 1688-1707, 2019.

[4] W. Siedlecki, and J. Sklansky, "On Automatic Feature Selection," *Handbook Of Pattern Recognition and Computer Vision*, pp. 63-87, 1993.

[5] J. Doak, "An Evaluation of Feature Selection Methods and Their Application to Computer Security," *UC Davis Dept. of Computer Science Tech. Reports*, CSE-92-18,1992.

[6] M. Belkin, and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation," *Neural Computation*, vol. 15, no. 6, pp. 1373-1396, 2003.

[7] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions," *Proceedings of the Twentieth International Conference Machine Learning (ICML 2003)*, pp. 912-919, 2003.

[8] L. E. Raileanu, and K. Stoffel, "Theoretical Comparison between the Gini Index and Information Gain Criteria," *Annals of Mathematics and Artificial Intelligence*, vol. 41, no. 1, pp. 77-93, 2004.
[9] M. I. Jordan, and T. M. Mitchell, "Machine Learning: Trends, Perspectives, and Prospects," *Science*, vol. 349, pp. 255-260, 2015.
[10] X. Yan, G. Jones, and J. Li, "A Study on Mutual Information-based Feature Selection under Text Categorization," *Journal of Computational Information Systems*, vol. 3, no. 3, pp. 1007-1012, 2007.
[11] X. He, D. Cai, and P. Niyogi, "Laplacian Score for Feature Selection," *Advances in Neural Information Processing Systems*, pp. 507-514, 2005.
[12] Q. Gu, Z. Li and J. Han, "Generalized Fisher Score for Feature Selection," *Proceedings of the 18th International Conference on Neural Information Processing Systems*, pp. 507-514, 2012.
[13] X. Zhang, G. Wu, and Z. Dong, "Embedded Feature-selection Support Vector Machine for Driving Pattern Recognition," *Journal of the Franklin Institute*, vol. 352, no. 2, pp. 669-685, 2015.
[14] B. Aaha, C. Sm and D. Hf, "Harris Hawks Optimization: Algorithm and Applications," *Future Generation Computer Systems*, vol. 97, pp. 849-872, 2019.
[15] A. Faramarzi, M. Heidarinejad and S. Mirjalili, "Marine Predators Algorithm: A Nature-inspired Metaheuristic," *Expert Systems with Applications*, vol. 152, pp. 113377, 2020.
[16] X. S. Yang, "Firefly Algorithm, Levy Flights and Global Optimization," *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications*, pp. 209-218, 2010.
[17] I. Fister, X. S. Yang and J. Brest, "Modified Firefly Algorithm Using Quaternion Representation," *Expert Systems with Applications*, vol. 40, no. 18, pp. 7220-7230, 2013.
[18] E. Emary, H. M. Zawbaa, and A. E. Hassanien, "Binary Ant Lion Approaches for Feature Selection," *Neurocomputing*, vol. 213, pp. 54-65, 2016.