Applied Intelligence
https://doi.org/10.1007/s10489-021-02302-9

Improved binary pigeon-inspired optimization and its application for feature selection

Jeng-Shyang Pan[1] · Ai-Qing Tian[1] · Shu-Chuan Chu[1,2] · Jun-Bao Li[3]

Accepted: 2 March 2021
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

The Pigeon-Inspired Optimization (PIO) algorithm is an intelligent algorithm inspired by the behavior of pigeons returned to the nest. The binary pigeon-inspired optimization (BPIO) algorithm is a binary version of the PIO algorithm, it can be used to optimize binary application problems. The transfer function plays a very important part in the BPIO algorithm. To improve the solution quality of the BPIO algorithm, this paper proposes four new transfer function, an improved speed update scheme, and a second-stage position update method. The original BPIO algorithm is easier to fall into the local optimal, so a new speed update equation is proposed. In the simulation experiment, the improved BPIO is compared with binary particle swarm optimization (BPSO) and binary grey wolf optimizer (BGWO). In addition, the benchmark test function, statistical analysis, Friedman's test and Wilcoxon rank-sum test are used to prove that the improved algorithm is quite effective, and it also verifies how to set the speed of dynamic movement. Finally, feature selection was successfully implemented in the UCI data set, and higher classification results were obtained with fewer feature numbers.

Keywords Pigeon-inspired optimization · Transfer function · Binary version · Wilcoxon rank sum test · Feature selection

1 Introduction

The swarm intelligence algorithm refers to a group inspired by individuals, with both the discrete behavior of individuals and the overall movement of the entire population [1, 2]. According to several simple behaviors of the individuals in the population, the entire population is affected, and the population behaves intelligently [3, 4]. For example, the behaviors of birds, fish, and gray wolves going home, foraging, and migration, the growth behavior of microorganisms and plants. In physics, the temperature or pressure of gas molecules under the conditions of temperature and humidity and the effect of transition from stable to unstable, the movement of gas molecules always presents an optimal path. Inspired by the behaviors of biological and non-biological groups in nature, researchers have proposed a random optimization algorithm called swarm intelligence algorithm, which is used to simulate the process of individual foraging, returning home, migration, and other behaviors. It uses a random number of simulated individuals in a specified search space to simulate individuals in nature, and uses the fitness function value of the individual at a certain position as a judgment of the individual's adaptability to the environment. According to each individual fitness can be used to simulate natural selection [5, 6].

Kenneth Sorensen proposed why the development of meta-heuristic algorithms was introduced, what are the main fallacies of most meta-heuristic algorithms, and why the field of meta-heuristic algorithms is easily affected [7]. Exhaustive search is quite impractical. No matter how clever the design, heuristic algorithms have been designed to solve

✉ Shu-Chuan Chu
scchu0803@gmail.com

Jeng-Shyang Pan
jspan@cc.kuas.edu.tw

Ai-Qing Tian
stones12138@163.com

Jun-Bao Li
lijunbao@hit.edu.cn

1 College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

2 College of Science and Engineering, Flinders University, 1284 South Road, Tonsley SA 5042, Australia

3 Harbin Institute of Technology, Harbin, China

Published online: 12 April 2021

people's daily life problems. Although the computing power of computers has increased exponentially, many problems are still difficult to solve, so faster and more accurate algorithms are needed to solve real problems. Even though the heuristic method is highly resisted by academia, under certain circumstances, the heuristic method can achieve relatively good results.

With the continuous development of swarm intelligence algorithms, researchers continue to try to apply swarm intelligence algorithms to various fields to solve various problems in this area. From the current perspective, the results obtained are satisfactory [8, 9]. According to different research results, the clustering intelligent algorithm performs well in discrete and continuous solution spaces, and is an object worthy of studying. In addition, it also has outstanding performance in combinatorial optimization problems. The development of intelligent computing so far has mainly formed three major parts, namely Fuzzy [10, 11], Neural Network [12–14] and Evolutionary Computation [15–17]. The swarm intelligence algorithm includes Genetic Algorithm (GA) [18–20], Particle Swarm Optimization (PSO) [21–23], Differential Evolution Algorithm (DE) [24–26],. Artificial Bee Colony Algorithm (ABC) [27, 28], Grey Wolf Optimizer (GWO) [29, 30], Pigeon Inspired Optimization (PIO) [31–33].

As a meta-heuristic algorithm, the swarm intelligence algorithm can effectively solve most parameter optimization problems. However, compared with other mature algorithms, the research is still in its infancy in this area, and there are still many problems that need further study. In terms of research, it lacks theoretical analysis and basic support of mathematics and has no universal significance. In the research process, the parameters in the algorithm are generally summarized based on the continuous research of the researchers and based on experience, and it does not explain well that the choice of parameters is scientific, systematic, and mathematical. There is no precise theoretical basis and application. The application program environment and specific problems are highly dependent. Therefore, for special problems, it is necessary to expand the application field of cluster intelligence and propose new algorithms. With the innovation and development of network and information technology, the amount of data produced by human beings every day is increasing rapidly at a very fast rate.

Since the beginning of the information age, all walks of life gradually began to spread information, which led to the explosion of data volumes. The information will result in a growing number of high-latitude data set. It makes it very difficult to classify [34]. However, all features are not suitable for classification. In addition to related features, the data set also includes irrelevant and redundant features. This even leads to low classification accuracy and becomes a dimensional disaster. In the classification process, a growing number of other dimensions will bring the noise to the entire data set, this noise can damage the ability to classify data sets. So, choose the original features from the more appropriate subset of features in order to get all the features of similar or even more good classification results are very important [35].

Feature selection is an important task in data mining and machine learning, which can reduce the dimension of data and improve the performance of learning algorithms [36–40]. Feature selection is to select a subset of features form all features. It is a representative process to reduce the number of dimensions of the feature space. It is one of the key issues of data mining and machine learning. For data mining and machine learning, a good learning sample is the key to training the classifier. Whether the sample contains irrelevant or redundant features directly affects the performance of the classifier. The purpose of feature selection is to find adequate minimum subset necessary to solve feature problem. Removed from the original by several features irrelevant and redundant features to reduce the dimensionality of data, accelerate the process of learning, learning model simplification and improvement of the performance of the algorithm. An effective feature selection method is the key to find an optimal feature subset [41, 42].

Realistic data sets are usually described by a set of features, which contain a lot of information, it also increases redundancy and noise. As the data dimension increases and the search space increases, it becomes particularly difficult to select the optimal feature subset. For example, for a data set with n features, the number of feature subsets is 2^n. As the complexity of the problem increases, the feature dimensions of data in many domains are gradually increasing, and feature selection becomes more challenging. Therefore, in most cases, exhaustive method to solve the selection of the optimal feature subset of the data set is basically impossible. There are many search techniques applied to feature selection, such as full search, greedy search, heuristic search and random search. Existing search technology has achieved great success in feature selection, however, most of the methods are easy to fall into local optimum, and the higher cost of computing. Therefore, a simple and efficient global search technique is needed to better solve the feature selection problem.

The swarm intelligence algorithm has powerful global and local search capabilities and potential, so it has been widely used in feature selection in recent years. The researchers obtained high accuracy and fewer features in the swarm intelligence algorithm. It changed the continuous algorithm into a binary version through the conversion function to achieve feature selection.

The PIO is a new type of intelligent intelligence algorithm that imitates the behavior of pigeon groups returned to the nest [43, 44]. The algorithm has achieved good results in

global search, Asaju La'aro Bolaji first proposed a new binary algorithm [45], and other algorithms provided various binary versions for feature selection [46, 47]. This paper proposes an improved binary PIO algorithm to improve the solution quality, the main contributions are as follows:

(1) Four new transfer function are proposed.
(2) Improved the speed update equation.
(3) The speed value is normalized and used.
(4) The performance of the improved binary PIO algorithm is proved through experiments.
(5) Based on the improved algorithm, the feature selection is successfully implemented.

Other arrangements for the remainder of this paper is as follows. The Section 2 in details introduces the implementation of PIO and BPIO. The Section 3 proposes four new transfer function from the perspective of improving the quality of the solution. Section 4 uses the test function to prove the performance of the newly proposed BPIO algorithm. The Section 5 introduces the feature selection, and uses the improved BPIO to select the feature of the test database and evaluate its performance. Section 6 summarizes the work and puts forward research suggestions for the next step.

2 Related work

Duan and Qiao proposed the Pigeon-Inspired Optimization algorithm based on the pigeons going home in 2014. In this part, this article only briefly introduces the basic theory of PIO and BPIO.

2.1 The original pigeon-inspired optimization

The behavior mechanism of bird clusters in nature, especially the intelligent behavior modeling and practical research of pigeon groups, have attracted the attention of scholars in many fields such as ornithologists, physicists, computer experts. After years of investigation and research, scientists have confirmed that the pigeon's upper beak does have a cell that can induce a magnetic field. It is this organ that guides the pigeon's flight. During the flight, the pigeons are greatly affected by the sun, the earth's magnetic field, and landmark buildings. As the distance from home decreases, the dependence of the pigeons on landmarks becomes greater. PIO is inspired by the above scenario [48].

$$\mathrm{vel}_i^{t+1} = e^{-R\times t} \times \mathrm{vel}_i^t + \mathrm{rand}() \times (\mathrm{pos}_{best}^t - \mathrm{pos}_i^t) \tag{1}$$

Where, i represents the ith pigeon in a certain iteration, vel_i is the flight speed of the pigeon and pos_i is the position of the pigeon in the search space, R is the map and compass factor. In the PIO algorithm, the value of R is usually taken 0.2, or 0.02, pos_{best} is the position of the pigeon with the best fitness value so far in the iteration, the pigeon with the best fitness, rand() is a random value between 0 and 1, and t is the current number of iterations.

In order to make the expression more clear to understand, the algorithm uses the following rules: Finding a home for the entire population means finding the optimal solution of the algorithm. Each scheme represents a pigeon candidate solutions, during the execution of the algorithm does not consider the quality and size of the pigeons. The algorithm compared with the particle swarm optimization (PSO), thinking that the pigeon only has position and speed. In the early stage, the sun and geomagnetism were used to form a map in the pigeon mind. Later, when approaching the destination, the landmark was used for navigation. In each iteration flight, each individual in the pigeon group follows the speed update (1).

$$\mathrm{pos}_i^{t+1} = \mathrm{pos}_i^t + \mathrm{vel}_i^{t+1} \tag{2}$$

As can be seen from the above (1) and (2), PIO and PSO are similar. One disadvantage of PSO is that it lacks the ability to exploit at a precise point. In this respect, PIO lacks the ability to explore, because PIO has less influence of an individual's historical optimal parameters than PSO, resulting in a global search ability that is much greater than a local search ability. In order to prevent local optimization and increase the accuracy of local search capabilities, PIO built a second-stage search. The first phase relies on the sun and geomagnetism to form a map in the pigeon's mind to guide the pigeon flight. In the second phase, the map operator does not apply at this stage. When the pigeon is close to the destination, the pigeon will continue to guide the pigeon flight iteration according to the landmark.

$$\mathrm{pos}_c^t = \frac{\sum_{i=1}^{Np^t} \mathrm{pos}_i^t \times F(\mathrm{pos}_i^t)}{Np^t \times \sum_{i=1}^{Np^t} F(\mathrm{pos}_i^t)} \tag{3}$$

$$Np^{t+1} = \frac{Np^t}{2} \tag{4}$$

$$\mathrm{pos}_i^{t+1} = \mathrm{pos}_i^t + \mathrm{rand}() \times (\mathrm{pos}_c^t - \mathrm{pos}_i^t) \tag{5}$$

Where Np is the number of pigeons. In the second stage of the landmark operator, in order to ensure fast convergence and optimal value of the solution, a halving operation is performed after each iteration, as shown in (4) and Fig. 1. The solution represented by the pigeon, where the solution is better, or a virtual solution as a landmark to guide other pigeons to continue to iterate until the end of the algorithm iteration, (3) and (5) represent the selection of landmarks and the position update equation for each pigeon, where pos_c^t is the center point selected in the second stage that can guide the pigeon flight.

$$F(\mathrm{pos}_i^t) = \begin{cases} \frac{1}{\mathrm{fitness}(\mathrm{pos}_i^t)+\theta} & \text{for minimization problems} \\ \mathrm{fitness}(\mathrm{pos}_i^t) & \text{for maximization problems} \end{cases} \tag{6}$$

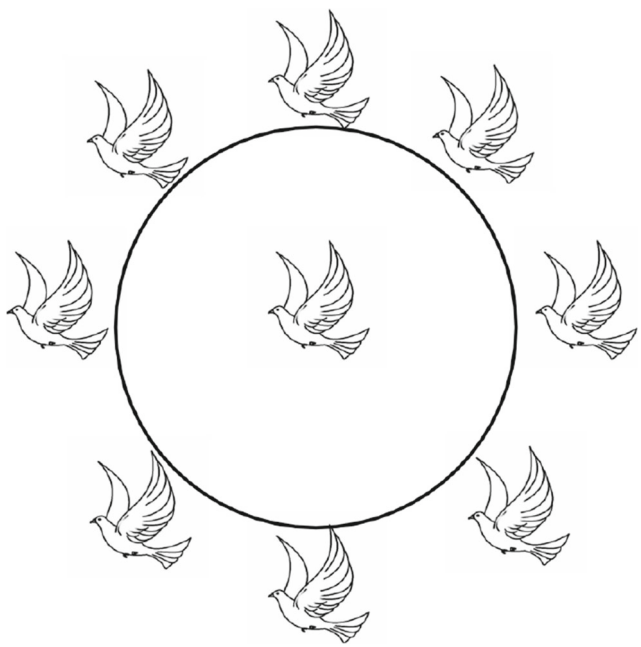

Fig. 1 Landmark operator model

Algorithm 1 PIO algorithm pseudo code display.

```
set Map and compass factors R to 0.02.
set Maximum iterations of map and operators Max_1 to 100.
set Maximum iterations of landmark operator Max_2 to 50.
for each i ∈ [1, Np] do
    Randomly initialize the position and velocity of the each pigeon.
end for
Calculate the fitness value of each pigeon
If the new fitness value is greater than the global optimal value, replace it
//Enter map and compass operator
while t ≤ Max_1 do
    Update the position and velocity of the each pigeon using (1) and (2).
    Update the best fitness value and global best position.
end while
//Enter landmark operator
while Max_1 < t ≤ Max_1 + Max_2 do
    Use (3) and (6) to find the pigeon at the center point
    Update the number of population by (4)
    Update the position by (5)
    Update the best fitness value and global best position.
end while
```

In addition, the definition of $F(pos_i^t)$ function is expressed as (6), and θ guarantees that the denominator is not zero, and $F(pos_i^t)$ is a value that is always greater than zero.

Similarly, when the iteration achieve the maximum iteration number, the landmark operator stops working. PIO pseudo code is shown in Algorithm 1. The parameters in the pseudo code have been introduced in the above article.

2.2 Binary pigeon-inspired optimization

In the original pigeon group algorithm, because the position value of the pigeons changes continuously, the pigeons can fly freely in the search space.

$$F(vel_i^t)) = |\tanh(vel_i(t))| \tag{7}$$

$$F(vel_i^t)) = \frac{2}{1 + e^{-|vel_i^t|}} - 1 \tag{8}$$

In the binary PIO, because the value of the pigeon position can only take 0 or 1, the pigeons cannot fly to any position at will, so for this feature, the basic theory in the original pigeon algorithm must be revised. In order to convert the continuity value into a binary value, binary pigeon-inspired optimization uses a conversion function and a probability function. As far as we know the use of (7) [46] and (8) [49] to control the speed of pigeon movement.

$$pos_i^{t+1} = \begin{cases} 1 - pos_i^t & \text{if rand} < F(vel_i^t)) \\ pos_i^t & \text{otherwise} \end{cases} \tag{9}$$

$$vel_i^{t+1} = vel_i^t + \text{rand}() \times \text{round}(pos_c^t - pos_i^t) \tag{10}$$

Where $F(vel_i^t)$ is the transfer function. rand() is a random value between 0 and 1, and round() is the rounding method.

Then use (9) to update the position of pigeon i, and force the continuous search space into binary space. It still uses the original speed update equation, the population halving equation, and the center point selection equation. Then, after converting the continuous PIO into a binary algorithm, it can be applied to the optimization problem of the discrete binary search space. In addition, considering that the landmark operator in the second stage does not have a finer equation for speed, the difference between BPIO and the original PIO algorithm are explained below. Equation (10) is the revised speed update equation for the second stage of BPIO. Algorithm 2 is the pseudo code of BPIO.

Algorithm 2 BPIO algorithm pseudo code display.

```
set Map and compass factors R to 0.02.
set Maximum iterations of map and compass operators Max_1 to 100.
set Maximum iterations of landmark operator Max_2 to 50.
for each i ∈ [1, Np] do
    Randomly initialize the position and velocity of the each pigeon.
end for
Calculate the fitness value of each pigeon
If the new fitness value is greater than the global optimal value, replace it
//Enter map and compass operator
while t ≤ Max_1 do
    Update the position and velocity of the each pigeon using (1) and (7), (9).
    Update the best fitness value and global best position
end while
//Enter landmark operator
while Max_1 < t ≤ Max_1 + Max_2 do
    Use (3) and (6) to find the pigeon at the center point
    Update the number of population by (4)
    Update the velocity by (10)
    Update the position by (7) and (9)
    Update the best fitness value and global best position
end while
```

3 Improved binary pigeon-inspired optimization

As can be seen from the previous section, one of the biggest differences between continuous PIO and binary PIO is that the position update equation becomes a transfer function and probability equation. The transfer function controls the speed value limited to 0 and 1. The mathematical properties of the transfer function and the function image also play a very important role in the performance of the algorithm. This paper proposes four new type of transfer function, in order to improve the overall PIO global and local search capabilities, the speed update equation has been modified.

3.1 New transfer function

The transfer function is very important in the running process of the BPIO algorithm, so the necessary attention must be given to the selection and setting of the transfer function. Since the transfer function value is a random number, the range of this function is limited to 0 and 1.

Table 1 The information about the improved transfer function

Function Name	Function expression
$V-1$	$\frac{\cos(\frac{\pi}{6}\times x+\pi)+1}{2}$
$V-2$	$\frac{\lvert\tanh(x)\rvert}{\tanh(4)}$
$V-3$	$\left\lvert\frac{x^{\frac{3}{2}}}{\sqrt{x+x^3}}\right\rvert\times\frac{\sqrt{37}}{6}$
$V-4$	$\frac{2}{\sqrt{\pi}}\times\int_0^{\frac{\pi}{2\times e}\times x} e^{t^2}$

When the pigeon moves away from the optimal solution of the function, the function value of the transfer function will gradually increase with the increase of speed, and the probability that it will change its position is relatively large. When the algorithm is working, the position of the pigeon is constantly changing between 0 and 1 in the early stage to search for more space. At the later stage of the algorithm work, it is necessary that the position of the pigeons is not so obvious, which can increase the algorithm local exploitation ability and also increase the ability to find the optimal solution. Therefore, this paper designs another transfer function, as shown in Table 1 and Fig. 2.

$V-4$ When the absolute value is large, the slope is relatively small and the absolute value is small, the function's slope is relatively large.

3.2 Mathematical analysis of velocity update equation

In this section, in order to promote discussion and understanding, this article assumes that pigeons have only one dimension. Therefore, use pos_i instead of pos_i^t. Then in the binary space, no matter what the value of the position is, it can only take 0 or 1, so in the process of speed update $(pos_{best}^t - pos_i^t)$ there are four representations.

(1) If pos_i=1 and pos_{best}=1
$$pos_{best} - pos_i = 0$$
(2) If pos_i=1 and pos_{best}=0
$$pos_{best} - pos_i = -1$$
(3) If pos_i=0 and pos_{best}=1
$$pos_{best} - pos_i = 1$$
(4) If pos_i=0 and pos_{best}=0
$$pos_{best} - pos_i = 0$$

Comment 1. It can be seen from the above analysis that the calculation of $(pos_{best} - pos_i)$ has three kinds of results. In addition, it can be inferred from (1) that after $(t-1)^{th}$ iterations $vel_i^t = e^{-R\times\sum_{i=1}^{t-1} i} \times vel_i^1 + e^{-R*\sum_{i=2}^{t-1}} \times rand() \times (pos_{best}^1 - pos_i^1) + e^{-R*\sum_{i=2}^{t-1}} \times rand() \times (pos_{best}^2 - pos_i^2) + \cdots + e^{-R*(t-1)} \times rand() \times (pos_i^{t-2} - pos_i^{t-2}) + rand() \times (pos_{best}^{t-1} - pos_i^{t-1})$. Let $H = e^{-R\times\sum_{i=1}^{t-1} i} \times vel_i^1 + e^{-R*\sum_{i=2}^{t-1}} \times rand() \times (pos_{best}^1 - pos_i^1) + e^{-R*\sum_{i=2}^{t-1}} \times rand() \times (pos_{best}^2 - pos_i^2) +$

Fig. 2 Transfer function

$\cdots + e^{-R*(t-1)} \times \text{rand}() \times (\text{pos}_i^{t-2} - \text{pos}_i^{t-2}) + \text{rand}() \times (\text{pos}_{best}^{t-1} - \text{pos}_i^{t-1})$. Then the values of H should be part of $\{-(t-1), -(t-2), ..., 0, (t-2), (t-1)\}$.

Comment 2. In order to search for a better solution, the selection of the coefficient parameter of the speed update equation is also important. It can be seen from Fig. 3 and Table 2 that when $R = 0.02$ is greater than $R = 0.2$, the function declines slowly and the derivative is large [50]. In order to prevent the candidate solution from quickly falling into the local optimum and the end of convergence quickly, this paper chooses $R = 0.02$ as the parameter of the speed coefficient.

Comment 3. In the early stage of the binary PIO algorithm, in other words, at the beginning of BPIO, the speed range of the pigeons varies slightly. In the middle and late stages of the algorithm, the range of changes in the algorithm is relatively large. Considering this aspect, in order to obtain a better optimal solution, the swarm intelligence algorithm must expand the search space and find a better optimal solution from a broader solution. It can be know from Fig. 2

Fig. 3 coefficient function

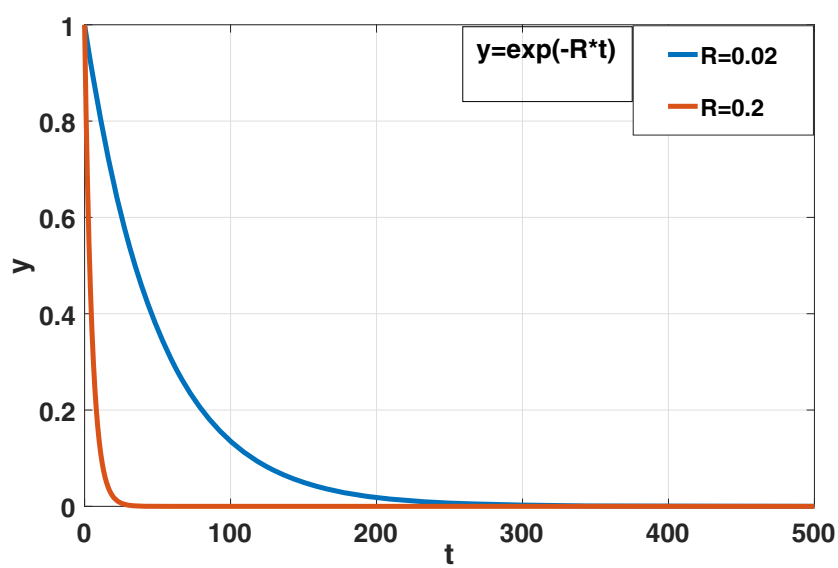

Table 2 R-value sensitivity analysis

R \ F	F1	F2	F3	F4	F5	F6
R=0.2	0.1000	0.1333	0.3333	**0.9667**	374.8333	7.7000
R=0.02	**0**	**0.0333**	**0**	1	**249.9000**	**7.5000**
R \ F	F7	F8	F9	F10	F11	F12
R=0.2	0.6871	−22.6075	0.0666	0.0239	0.0033	1.6807
R=0.02	**0.0101**	**−24.4026**	**0**	8.88×10^{-16}	**0**	**1.6689**
R \ F	F13	F14	F15	F16	F17	F18
R=0.2	0.2866	**12.6705**	**0.1484**	**0**	**27.7029**	**600**
R=0.02	**0.0933**	**12.6705**	**0.1484**	**0**	**27.7029**	**600**
R \ F	F19	F20	F21	F22	F23	
R=0.2	**−0.3348**	−0.1553	−4.3576	−4.6732	−4.2925	
R=0.02	**−0.3348**	**−0.1657**	**−5.0551**	**−5.0876**	**−5.1284**	

that the speed absolute value of pigeons are smaller, the steeper the tangent slope of the transfer function, the more obvious the change of the function value. With the increase of the absolute value of the switching position is gradually reduced, it is more desirable intelligent algorithm.

3.3 Improved updating equation for velocity equation

Inspired by PSO, many researchers also will increase the velocity of inertia weight updating equation to strengthen the capacity up search for a solution. But in PIO, the first phase focuses on exploration, and the second phase focuses on exploitation. This will cause a problem. As previously analyzed, as the number of iterations reaches the later stage, the absolute speed value of the pigeon will increase rapidly. For this reason, the algorithm may surpass the theoretical global optimal solution of the first stage and choose the suboptimal solution. Develop sub-optimal solutions in the second stage. However, the advantages of the first stage are also obvious. The setting of each step length is very important and must conform to the change of transfer function. Such a setting can prevent the algorithm from falling into a local optimal solution, and can increase the exploitation and exploration capabilities of the algorithm.

$$\begin{aligned} \mathrm{vel}_i^{t+1} &= \mathrm{e}^{-\mathrm{R}} \times \mathrm{t} \times \mathrm{vel}_\mathrm{i}^\mathrm{t} + \mathrm{rand}() \times (\mathrm{pos}_\mathrm{pbest}^\mathrm{t} - \mathrm{pos}_\mathrm{i}^\mathrm{t}) \\ &\quad + \mathrm{rand}() \times (\mathrm{pos}_\mathrm{gbest}^\mathrm{t} - \mathrm{pos}_\mathrm{i}^\mathrm{t}) \end{aligned} \tag{11}$$

Table 3 Unimodal test function

No	Function expression	Space	Dimension	TM
1	$F_1(y) = \sum_{j=1}^{No} y_j^2$	[−100, 100]	30	0
2	$F_2(y) = \sum_{j=1}^{No} \lvert y_j \rvert + \prod_{j=1}^{No} \lvert y_j \rvert$	[−10, 10]	30	0
3	$F_3(y) = \sum_{j=1}^{No} \left(\sum_{k=1}^{j} y_k\right)^2$	[−100, 100]	30	0
4	$F_4(y) = max_j \lvert y_j \rvert, j \in [1, m]$	[−100, 100]	30	0
5	$F_5(y) = \sum_{j=1}^{No-1} \left[100\left(y_{j+1} - y_j^2\right)^2 + \left(y_j - 1\right)^2\right]$	[−30, 30]	30	0
6	$F_6(y) = \sum_{j=1}^{No} \left(\left[y_j + 0.5\right]\right)^2$	[−100, 100]	30	0
7	$F_7(y) = \sum_{j=1}^{No} j \times y_j^2 + rand[0, 1)$	[−1.28, 1.28]	30	0

$$\begin{aligned} \mathrm{vel}_i^{t+1} &= \mathrm{vel}_\mathrm{i}^\mathrm{t} + \mathrm{rand}() \times \mathrm{round}(\mathrm{pos}_\mathrm{c}^\mathrm{t} - \mathrm{pos}_\mathrm{i}^\mathrm{t}) \\ &\quad + \mathrm{rand}() \times (\mathrm{pos}_\mathrm{gbest}^\mathrm{t} - \mathrm{pos}_\mathrm{i}^\mathrm{t}) \end{aligned} \tag{12}$$

In summary, in order to increase the local search capability of the first stage of BPIO, this paper refers to the speed update equation of PSO. On the other hand, as the number of iterations gradually increases, the absolute value of the speed increases. If the speed is kept at a large value, the optimal solution may be skipped, so there will be a limit to the speed. As shown in (11) and (12).

In BPIO, as the speed is limited to $[-6, 6]$, the nature of the function image in Fig. 2 is observed. It is better to limit the speed to $[-1, 1]$ to have a steeper property.

4 Experimental results and analysis

In this section, it mainly introduce the simulation experiment process of the algorithm, and analyze and explain the experimental results. This article uses 23 test benchmark functions to test the proposed improved algorithm exploration and exploitation capabilities [29].

Tables 3, 4 and 5 introduces the basic information of these 23 test functions. Where **TM** represents the minimum value of each test function that should be achieved in theory. **Dimension** represents the dimension of the test function. **Space** means the search space of all pigeons.

The above 23 test functions can be divided into three types of functions. Among them, $1-7$ are single-peak

Table 4 Multimodal test function

No	Function expression	Space	Dimension	TM
8	$F_8(y) = \sum_{j=1}^{No} -y_j \times \sin\left(\sqrt{\lvert y_j \rvert}\right)$	$[-500, 500]$	30	-12569
9	$F_9(y) = \sum_{j=1}^{No}\left[y_j^2 - 10 \times \cos\left(2\pi y_j\right) + 10\right]$	$[-5.12, 5.12]$	30	0
10	$F_{10}(y) = -20 \times \exp\left(-0.2\sqrt{\frac{1}{No}\sum_{j=1}^{No} y_j^2}\right) - \exp\left(\frac{1}{No}\sum_{j=1}^{No}\cos\left(2\pi y_j\right) + 20 + 2.718\right)$	$[-32, 32]$	30	0
11	$F_{11}(y) = \frac{1}{4000} \times \sum_{j=1}^{No} y_j^2 - \prod_{j=1}^{No}\cos\left(\frac{y_j}{\sqrt{j}}\right) + 1$	$[-600, 600]$	30	0
12	$F_{12}(y) = \frac{\pi}{No} \times \left\{10 \times \sin\left(\pi y_1\right) + \sum_{j=1}^{No-1}\left(y_j - 1\right)^2\left[1 + 10 \times \sin^2(\pi y_{j+1})\right] + \left(y_{No} - 1\right)^2\right\} + \sum_{j=1}^{No} u(y_j, 10, 100, 4)$, $y_j = 1 + \frac{y_j+1}{4}$ $\times$ $u(z_l, a, k, m) = \begin{cases} k(y_j - a), y > a \\ 0, -a < y_j < a \\ k(-y_j - a), y > a \end{cases}$	$[-50, 50]$	30	0
13	$F_{13}(y) = 0.1 \times \left\{\sin^2\left(3\pi y_1\right) + \sum_{j=1}^{No}\left(y_j - 1\right)^2\left[1 + \sin^2\left(3\pi y_j + 1\right)\right] + \left(y_{No} - 1\right)^2\left[1 + \sin^2\left(2\pi y_{No}\right)\right]\right\} + \sum_{j=1}^{No} u\left(y_j, 10, 100, 4\right)$	$[-50, 50]$	30	0

Table 5 Fixed dimension function

No	Function expression	Space	Dimension	TM
14	$F_{14}(y) = \left(\frac{1}{500} \times \sum_{j=1}^{25} \frac{1}{j+\sum_{k=1}^{2}(z_k - a_{kj})^6}\right)^{-1}$	$[-65, 65]$	2	1
15	$F_{15}(y) = \sum_{j=1}^{11}\left[a_j - \frac{y_1\left(b_j^2+b_j y^2\right)}{b_j^2+b_j y^3+y^4}\right]^2$	$[-5, 5]$	4	0.00030
16	$F_{16}(y) = 4y_j^2 - 2.1y_j^4 + \frac{1}{3}y_j^6 + y_j y_2 - 4y_2^2 + 4y_2^4$	$[-5, 5]$	2	−1.0316
17	$F_{17}(y) = \left(y_2 - \frac{5.1}{4\pi^2}y_j^2 + \frac{5}{\pi}y_j - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos y_j + 10$	$[-5, 5]$	2	0.398
18	$F_{18}(y) = \left[1 + (y_1 + y_2 + j)^2 \times \left(19 - 14y_1 + 3y_1^2 - 14y_2 + 6y_1 y_2 + 3y_2^2\right)\right] \times \left[30 + (2y_1 - 3y_2)^2 \times \left(18 - 32y_1 + 12y_1^2 + 48y_2 - 36y_1 y_2 + 27y_2^2\right)\right]$	$[-2, 2]$	2	3
19	$F_{19}(y) = -\sum_{j=1}^{4} c_j \times \exp\left(-\sum_{k=1}^{3} a_{jk}\left(y_k - p_{jk}\right)^2\right)$	$[1, 3]$	3	−3.86
20	$F_{20}(y) = -\sum_{j=1}^{4} c_j \times \exp\left(-\sum_{k=1}^{6} a_{jk}\left(y_k - p_{jk}\right)^2\right)$	$[0, 1]$	6	−3.32
21	$F_{21}(y) = -\sum_{j=1}^{5}\left[\left(y - a_j\right)\left(y - a_j\right)^T + c_j\right]^{-1}$	$[0, 10]$	4	−10.1532
22	$F_{22}(y) = -\sum_{j=1}^{7}\left[\left(y - a_j\right)\left(y - a_j\right)^T + c_j\right]^{-1}$	$[0, 10]$	4	−10.4028
23	$F_{23}(y) = -\sum_{j=1}^{10}\left[\left(y - a_j\right)\left(y - a_j\right)^T + c_j\right]^{-1}$	$[0, 10]$	4	−10.5363

function, $8 - 13$ are multi-peak function, and $14 - 23$ are other types of functions.

In order to verify the test results, this paper introduces the following algorithms in BPIO in detail, as shown in Table 6. The algorithms in the table all run 30 times, each iteration 100 times in the first phase, and 50 times in the second phase. The number of population size is set to 10.

4.1 Simulation experiment results

For the convenience of readers, the written expression of experimental data is uniformly expressed to four decimal places. Table 7 shows the mean ($Mean$) and standard deviation (Std) of the comparison function in the test function. The blue value means that it performs best in the

Table 6 The detail of the compared algorithms

Algorithm	Transfer function
BPIO	Original Transfer Function (7) from [46]
BPIO1	Improved Transfer Function $V - 1$
BPIO2	Improved Transfer Function $V - 2$
BPIO3	Improved Transfer Function $V - 3$
BPIO4	Improved Transfer Function $V - 4$
BPIO5	Original Transfer Function (8) from [49]
BGWO	Transfer Function from [51]
BPSO-S	Transfer Function from [52]

Table 7 Compare the results of mathematical statistics of several algorithms.

Functions	BPIO		BPIO1		BPIO2		BPIO3		BPIO4		BPIO5		BGWO		BPSO-S	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	0.0333	0.1826	0	0	0	0	0	0	0	0	0	0	4.6667	1.5025	3.9000	0.6618
F2	0.1667	0.3790	0	0	0	0	0	0	0	0	0	0	4.0333	1.5196	4.2000	0.6102
F3	0.3667	0.3790	0	0	0	0	0	0	0	0	0	0	140.0333	94.2855	147.2667	43.1516
F4	1	0	0	0	0.6333	0.4901	0.7000	0.4660	0.2333	0.4301	0	0	1	0	1	0
F5	385.1667	162.7281	29	0	0	0	0	0	0.9667	5.2946	29	0	0	0	418.5	76.4143
F6	7.7667	0.6914	7.5	0	7.5	0	7.5	0	7.5	0	7.5	0	15.6333	2.8735	16.1	1.6733
F7	0.7389	1.3653	5.14×10^{-4}	4.81×10^{-4}	5.06×10^{-4}	4.29×10^{-4}	5.59×10^{-4}	4.01×10^{-4}	5.46×10^{-4}	4.36×10^{-4}	6.73×10^{-4}	7.49×10^{-4}	42.8334	22.6518	44.9295	10.5653
F8	−23.0002	0.8366	−23.6172	3.1402	−25.24413	1.08×10^{-14}	−25.24413	1.08×10^{-14}	−25.24413	1.08×10^{-14}	−21.9623	1.0447	−24.2441	1.2234	−21.8782	0.5846
F9	0.0667	0.2537	0	0	0	0	0	0	0	0	0	0	3.8667	1.5477	4.0333	0.8899
F10	0.0478	0.1819	8.88×10^{-16}	0	8.88×10^{-16}	0	8.88×10^{-16}	0	8.88×10^{-16}	0	8.88×10^{-16}	0	1.3201	0.3747	1.4072	0.1354
F11	0.0024	0.0064	0	0	0	0	0	0	0	0	0	0	0.1862	0.0838	0.1203	0.0272
F12	1.6857	0.0518	1.6689	1.13×10^{-15}	1.6689	1.13×10^{-15}	1.6689	1.13×10^{-15}	1.6689	1.13×10^{-15}	1.6689	1.13×10^{-15}	2.3542	0.2627	2.2913	0.1216
F13	0.2866	0.1332	0.3633	0.4559	1.35×10^{-32}	5.57×10^{-48}	1.35×10^{-32}	5.57×10^{-48}	1.35×10^{-32}	5.57×10^{-48}	0.39	0.1348	1.69×10^{-32}	0.0032	0.4133	0.0628
F14	12.6705	3.61×10^{-15}	12.6705	3.61×10^{-15}	12.6705	3.61×10^{-15}	12.6705	3.61×10^{-15}	12.6705	3.61×10^{-15}	12.6705	3.61×10^{-15}	12.6705	3.61×10^{-15}	12.6705	3.61×10^{-15}
F15	0.1484	0	0.1484	0	0.1484	0	0.1484	0	0.1484	0	0.1484	0	0.1484	0	0.1484	0
F16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F17	27.7029	1.81×10^{-14}	27.7029	1.81×10^{-14}	27.7029	1.81×10^{-14}	27.7029	1.81×10^{-14}	27.7029	1.81×10^{-14}	27.7029	1.81×10^{-14}	27.7029	1.81×10^{-14}	27.7029	1.81×10^{-14}
F18	600	0	600	0	600	0	600	0	600	0	600	0	600	0	600	0
F19	−0.3348	1.69×10^{-16}	−0.3348	1.69×10^{-16}	−0.3348	1.69×10^{-16}	−0.3348	1.69×10^{-16}	−0.3348	1.69×10^{-16}	−0.3348	1.69×10^{-16}	−0.3348	1.69×10^{-16}	−0.3348	1.69×10^{-16}
F20	−0.1553	0.0272	−0.1657	2.82×10^{-17}	−0.1657	2.82×10^{-17}	−0.1657	2.82×10^{-17}	−0.1612	0.0474	−0.1333	0.0354	−0.1033	0.3742	−0.0021	0.1654
F21	−4.3576	1.5513	−5.0552	4.52×10^{-15}	−5.0552	4.52×10^{-15}	−5.0551	4.52×10^{-15}	−4.0814	1.7953	−4.9166	0.7619	−3.8765	0.8861	−1.2371	2.3591
F22	−4.67	1.2774	−5.0877	0	−5.0877	0	−5.0877	0	−5.0877	0	−4.5308	1.4444	−4.1732	2.0168	−3.4230	2.9634
F23	−4.2925	1.7001	−5.1285	2.71×10^{-15}	−5.1285	2.71×10^{-15}	−5.1285	2.71×10^{-15}	−4.8493	1.0608	−4.7105	1.2764	−4.6301	1.4760	−3.9614	1.6973

Table 8 Wilcoxon's rank-sum test for the optimal results on the benchmark functions

Function	BPIO	BPIO1	BPIO2	BPIO3	BPIO4	BPIO5	BGWO	BPSO-S
F1	−	0	0	0	0	0	−	−
F2	=	0	0	0	0	0	−	−
F3	=	0	0	0	0	0	−	−
F4	−	0	−	−	−	0	−	−
F5	−	−	0	0	=	−	0	−
F6	−	7.5	7.5	7.5	7.5	7.5	−	−
F7	−	=	5.06×10^{-4}	=	=	=	−	−
F8	−	−	−25.2441	−25.2441	−25.2441	−	−	−
F9	=	0	0	0	0	0	−	−
F10	−	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	−	−
F11	=	0	0	0	0	0	=	=
F12	=	1.6689	1.6689	1.6689	1.6689	1.6689	=	=
F13	−	−	1.35×10^{-32}	1.35×10^{-32}	1.35×10^{-32}	−	=	−
F14	12.6705	12.6705	12.6705	12.6705	12.6705	12.6705	12.6705	12.6705
F15	0.1484	0.1484	0.1484	0.1484	0.1484	0.1484	0.1484	0.1484
F16	0	0	0	0	0	0	0	0
F17	27.7029	27.7029	27.7029	27.7029	27.7029	27.7029	27.7029	27.7029
F18	600	600	600	600	600	600	600	600
F19	−0.3348	−0.3348	−0.3348	−0.3348	−0.3348	−0.3348	−0.3348	−0.3348
F20	−	−0.1657	−0.1657	−0.1657	=	−	−	−
F21	−	−5.0552	−5.0552	−5.0552	−	=	−	−
F22	−	−5.0877	−5.0877	−5.0877	−5.0877	=	−	−
F23	−	−5.1285	−5.1285	−5.1285	−	−	−	−

test function. In addition, in order to judge whether the result data of the experiment has statistical significance. Tables 8 and 9 show the statistical data of the experimental results and the results of the rank-sum test. The blue numbers in Table 8 indicate the values with better values in the test function. "−" Indicates that this algorithm is not as good as those that achieve the best results, and "=" refers to the equivalent value of this algorithm and other algorithms. The "+" symbol represents that the algorithm of comparison is excellent compared to BPIO, and "−" represents that the comparison algorithms worse than BPIO. Then "=" indicates that comparison algorithm and BPIO performance have the same performance.

4.2 Analysis of results

The unimodal function has only one global optimal solution and no other local optimal solutions. Therefore, such functions are very important for studying the convergence performance of the meta-heuristic algorithm. In addition to the original BPIO algorithm, the performance of other comparison algorithms is quite excellent and basically can achieve the theoretical optimal value of each function. In F1- F5, the improved algorithm can basically achieve the theoretical optimal value. For F6, all algorithms are far different from the theoretical optimal value, but BPIO1, BPIO2, and BPIO4 have the smallest distances from the optimal value, which means they are closer to the optimal value. Especially for F5, the worst BPIO got 418.5, BPIO2 and BPIO3 can reach the best optimal value. In addition, in the function F7, BPIO1, BPIO2, BPIO3, BPIO4 and BPIO5 performs better than BPIO algorithms, because limiting the value range of the transfer function to between 0 and 1 can increase the conversion capability of the algorithm. In the comparison of the improved algorithm with BGWO and BPSO-S, the latter performance is basically not perfect, except that BGWO can achieve the theoretical optimal value in F7.

The multimodal functions have more local optimal values than single-modal functions. Such functions can detect the exploration ability of the algorithm and whether the algorithm can avoid local optimal values. Among these functions, the improved algorithm of F9 and F11 can obtain the optimal value, but for other functions, the optimal value is not obtained. For F10 and F12, although the improved BPIO did not achieve the optimal value, the value compared to the original algorithm has become much better. Finally, for F8, the algorithm has not achieved an ideal solution. This

Table 9 Friedman's test for BPIO on the benchmark functions

Friedman's test	Sum of squares	Degree of freedom	Mean squares	P-value
F1	4534	239	19.1084	0.2574
F2	4348.5	239	19.8673	0.9648
F3	5440	239	22.9935	0.2803
F4	3975	239	16.8842	0.3219
F5	4906	239	20.9175	0.3561
F6	4314	239	17.8147	0.1335
F7	17980	239	81.5468	0.9357
F8	7665.5	239	32.2977	0.2555
F9	4111	239	18.0092	0.6423
F10	4003.5	239	17.2167	0.4409
F11	5592	239	24.633	0.7019
F12	4436	239	27.94	0.5212
F13	6915.5	239	28.9255	0.2042
F14	0	239	0	1
F15	0	239	0	1
F16	0	239	0	1
F17	0	239	0	1
F18	0	239	0	1
F19	0	239	0	1
F20	1233	239	5.31466	0.4651
F21	0	239	0	1
F22	0	239	0	1
F23	0	239	0	1

is a further in-depth mining and analysis of the multimodal function. In the comparison of the improved BPIO with BGWO and BPSO-S, the former can basically achieve the theoretical optimal value, while the latter cannot achieve the former effect in comparison.

Finally, in other test functions, there are only a few local minimums and features with fewer dimensions. It can be clearly seen from Table 7 that the mean and variance of the six algorithms are almost the same. The exploration and exploitation of algorithms at low latitudes require further research. However, although the values of the F14- F23 functions are the same, compared with the already calculated excellent solutions, the next step requires designing some solutions to the characteristics of low-latitude functions to accelerate exploitation and exploration.

Discussion regarding the above, it can improve the ability to optimize the binary PIO by modifying the updating equation of velocity and the transfer function of the swarm. Although the two speed update equation can speed up finding the optimal value, it is very easy to fall into the local optimal. It can be seen from Table 7 that it have 12 does not obtain good function values at low latitudes and multi-peak functions, and does not have very good local exploration capabilities. Optimal results in the second stage of the exploitation process is not effective, it may be due to the iterative update of the algorithm to the second stage, the exploitation stage does not meet the needs of exploitation.

The data in Table 8 can show that the overall performance of the improved algorithm is better than the original algorithm. BPIO1 have nineteen functions better than BPIO, and the others are the same. Compared with the original BPIO, BPIO2, it has sixteen excellent, and seven functions are the same results. In other words, the conversion function proposed in this paper has a better ability than the original algorithm to help algorithm exploitation and exploration. This shows that the proposed new update method improves the ability of BPIO to solve functions in this paper.

In this paper, Friedman's test was used to detect significant differences between the comparison algorithms [53–55]. Table 9 clearly shows that the p-values are all greater than 5%, which means that there is no significant difference in levels between the eight comparison algorithm samples, and therefore the hypothesis is rejected.

5 Apply to feature selection

For a particular learning algorithm, which feature is effective is unknown. Therefore, it is necessary to select relevant features that are beneficial to the learning algorithm from all the features. And in practical applications, the problem of dimensional disasters often occurs, especially in text processing. For example, you can represent a document as a word vector, but often use all words as a dictionary, so

for a document that may only contain 100 or 200 words, you may need tens of thousands of dimensions (that is features). In the process of feature selection, the model constructed by some features selected by the algorithm can solve such problems to a certain extent. Therefore, feature selection is similar to dimensionality reduction to a certain extent. In addition, from the above results, it can be easily found that if only some features are selected to construct the model, the running time of the learning algorithm can be greatly shortened, and the interpretability of the model can also be increased.

5.1 Feature selection

In the process of classification, too high dimensions will cause the problem of dimensionality disaster. In the process of dimensionality reduction, high dimensions may retain some unrelated dimensions, and low dimensions may also remove some of the more important features. Therefore, feature selection has become a more important issue. In the judgment process of the classifier, feature selection will have a very important impact on accuracy and complexity. The classification accuracy of new instances after feature selection is an important evaluation criterion for feature subsets. Feature selection includes three methods: filter, wrap and embedded. (1) Filter: Select features first, then train the learner, so the feature selection process has nothing to do with the learner. It is equivalent to filtering the features first. Then use the feature subset to train the classifier. (2) Wrap: directly use the last classifier as the evaluation function of feature selection, and select the optimal feature subset for a specific classifier. (3) Embedded: The process of feature selection is combined with the process of classifier learning, and feature selection is performed during the learning process. The most common uses L1 regularization for feature selection.

5.2 Simulation results applied to feature selection

The source of these data sets is UCI machine learning [56]. Each category has different instances and the size of the database. The specific information can be seen in Table 10.

5.2.1 KNN and K-fold validation

The K-Nearest Neighbor(KNN) classification algorithm is one of the simplest methods in data mining classification technology that the meaning of k nearest neighbors means that each sample can be represented by the k nearest neighbors. KNN is a classification algorithm, which inputs instance-based learning. If most of the k most similar samples in the feature space (ie, the nearest neighbor in the feature space) of a sample belong to a certain category, the sample also belongs to this category, where k is usually

Table 10 The details information of the simulation data sets

Datasets	Instances	Number of classes (k)	Number of features (d)	Size of classes
Wine	178	3	13	59,71,48
WDBC	569	2	30	357212
Glass	214	6	9	29,76,70,17,13,9
WBC	683	2	9	444239
Vowel	871	6	3	72,89,172, 151,207,180
Vehicle	846	3	18	199217218212
Thyroid	215	3	5	150,35,30
Sonar	208	2	60	97111
Seeds	210	3	7	70,70,70
Robotnavigation	5456	4	25	82620972205329
Jain	373	2	2	276,97
Ionosphere	351	2	34	126225
Heartstatlog	270	2	13	150120
Haberman	306	2	3	225,81
Ecoli	336	8	8	143,77,2,2, 259,20,5,52
Diabetes	768	2	8	268500
CMC	1473	3	9	629333511
Cancer	683	2	9	444239
Bupa	345	2	6	145200
Blood	748	2	4	570178
Balancescale	625	3	4	49288288

Table 11 The accuracy and number of the compared algorithms

Functions	BPIO		BPIO1		BPIO2		BPIO3		BPIO4		BPIO5		BGWO		BPSO-S	
	Acc	Num	Acc	Num	Acc	Num	Acc	Num	Acc	Num	Acc	Num	Acc	Num	Acc	Num
Wine	0.9816	5.0666	0.9914	5.0333	0.9881	5.7	0.9881	4.9	0.9876	5.5667	0.9731	4.4333	0.9791	6.5667	0.9812	5.4
WDBC	0.9744	9.5333	0.9843	4.6667	0.9799	6.3667	0.9807	7.4	0.9807	6.0667	0.9778	3.6	0.9766	12.5	0.9741	10.7333
WBC	0.3761	3.0333	0.4097	1.2667	0.3973	2.9	0.3891	2.5333	0.39	2.2	0.3817	1.3667	0.4143	3.8333	0.4003	2.20667
Vowel	0.9626	3.4333	0.9781	3.0667	0.98	3.8	0.9791	3.7	0.9772	3.443	0.9628	3.9967	0.9711	3.3	0.9709	2.0667
Vehicle	0.8687	3	0.8731	3	0.879	3	0.8783	3	0.8883	3	0.8662	3	0.8734	3	0.8735	3
Thyroid	0.7387	9.1333	0.7114	8.1667	0.7259	7.7666	0.7244	7.8667	0.7225	7.5	0.7114	6.7667	0.7079	10.5667	0.7171	7.7
Sonar	0.9682	1.9666	0.9744	1.5667	0.9731	1.9	0.9729	1.8666	0.9733	1.9	0.9522	1.8667	0.9502	3.1667	0.9682	2.8667
Seeds	0.8502	25.4	0.886	22	0.8733	25.9	0.8697	22.4	0.8669	24.8	0.8538	24.9	0.8452	29.6	0.8753	27.9667
Robotnavigation	0.9337	2.05	0.9461	2	0.9429	2	0.9439	2	0.9439	2	0.9446	11.1	0.9052	3.4333	0.8853	2
Jain	0.9347	4.1667	0.9477	3.4333	0.9329	3.8667	0.9224	4.0333	0.9248	3.8333	0.9319	3.5	0.8952	3.4333	0.9165	2
Ionosphere	0.9589	1.9	0.9897	1.3667	0.9871	2.0333	0.9899	2.0333	0.9891	2.0333	0.9608	1.5	0.9141	7.1	0.9149	5.9667
Heartstatlog	0.9194	3.9333	1	1	1	3	1	3.1333	1	1.8667	0.9312	1	0.9656	1.1	0.9716	1
Haberman	0.8248	5.3333	0.8543	4.7333	0.8549	4.5666	0.8526	4.5667	0.8453	4.3333	0.8370	4.3	0.8265	6.8667	0.8157	4.5
Glass	0.7320	0.0666	0.7628	0.2666	0.7642	0.2	0.7653	0.2667	0.7508	0.1444	0.7333	0.1333	0.7436	0.1326	0.7349	0.6831
Ecoli	0.8562	5.8667	0.9015	5.4333	0.8778	6.6667	0.8746	6.4666	0.8751	6.1333	0.8609	5.7333	0.8268	4.8	0.8435	4.3
Diabetes	0.7506	3.7666	0.7912	3.6333	0.7871	3.5	0.7880	3.6666	0.7876	3.4666	0.7609	3.5333	0.7339	4.2	0.7536	3.7667
CMC	0.5288	3.0666	0.5422	2.9	0.5390	3.0666	0.5396	3.4	0.5330	3.0333	0.5332	3	0.5290	4.333	0.5296	3.5333
Cancer	0.9678	2.7667	0.9725	2.5666	0.9732	2.9	0.9728	2.8667	0.9722	2.8333	0.9602	2.6667	0.9507	3.3667	0.9672	2.4333
Bupa	0.6737	3.7333	0.7249	3.0333	0.7245	3.7666	0.7239	3.8666	0.6990	3.6667	0.6884	3.5	0.6696	3.3666	0.6336	3
Blood	0.7671	1	0.7671	1	0.7671	1	0.7671	1	0.7690	0.1333	0.7680	0.0666	0.7673	1	0.7670	1
Balancescale	0.8321	4	0.8454	3.9667	0.8442	4	0.8439	4	0.8433	4	0.8351	4	0.8159	4	0.8037	4

an integer not greater than 20. In the KNN algorithm, the selected neighbors are all objects that have been correctly classified. This method determines the category to which the samples to be classified belong based on the category of the nearest sample or samples in the decision of classification.

$$\mathrm{dis}(x, y) = \left(\sum_{k=1}^{n} |x_k - y_k|^t\right)^{\frac{1}{t}} \tag{13}$$

In KNN, the distance between objects is calculated as a non-similarity index between each object to avoid the matching problem between objects. Here, the distance generally uses Euclidean distance or Manhattan distance as shown in the (13):

Where, x_k, y_k are two instances in the collection, t is a variable constant. When $t = 1$, dis represents the Manhattan distance, and when $t = 2$, dis refers to the Euclidean distance.

Cross-Validation is sometimes called circular estimation, it is a practical method for statistically cutting data samples into smaller subsets. Therefore, the analysis can be performed on a subset first, and the other subsets are used for subsequent confirmation and verification of the analysis. The initial subset is called the training set. The other subsets are called validation sets or test sets. As an evolution of Holdout Methon, K-fold Cross Validation (K-CV) divides the original data into k groups (usually divided equally), and each subset of data is used as a validation set, and the remaining k-1 group of subset data is used as a training set k models, and use the average number of classification accuracy of the final verification sets of these k models as the performance index of the classifier under this K-CV. k is generally greater than or equal to 2, and it is generally taken from 3 in actual operation. Only when the amount of data in the original data set is small will try to take 2. The K-CV experiment needs to build a total of k models and calculate the average recognition rate of k test sets. In practice, k must be large enough to allow enough training samples in each round. Generally speaking, k = 10 (as an empirical parameter) is quite sufficient.

5.2.2 Evaluation standard fitness value function

$$\text{Fitness Function} = k_1 \times \text{Error(KNN)} + k_2 \times \frac{|SE|}{|AL|} \tag{14}$$

In general classification prediction, the accuracy or error rate of the classification will be selected as the most important evaluation standard. In the process of feature selection, what should be considered is the accuracy of classification and the number of subsets after feature selection. (14) is evaluation standard used in this article.

$$\text{Error(KNN)} = \frac{\text{Wrong}_{\text{num}}}{\text{Correct}_{\text{num}} + \text{Wrong}_{\text{num}}} \tag{15}$$

Table 12 The fitness of the compared algorithms

Datasets Function	BPIO	BPIO1	BPIO2	BPIO3	BPIO4	BPIO5	BGWO	BPSO-S
Wine	0.0376	0.0368	0.0294	0.0301	0.0294	0.0331	0.0383	0.3759
WDBC	0.0321	0.0271	0.0285	0.0297	0.0297	0.0306	0.0335	0.0329
WBC	0.5996	0.5774	0.5886	0.5944	0.5917	0.5949	0.5881	0.5926
Vowel	0.0407	0.0399	0.0371	0.0369	0.0373	0.0423	0.0457	0.0409
Vehicle	0.1800	0.1644	0.1647	0.1651	0.1751	0.1751	0.1945	0.1892
Thyroid	0.2736	0.2634	0.2655	0.2651	0.2693	0.2943	0.2854	0.2901
Sonar	0.0693	0.0458	0.0445	0.0444	0.0443	0.0550	0.0599	0.0588
Seeds	0.1553	0.1310	0.1413	0.1401	0.1359	0.1516	0.1411	0.1522
Robotnavigation	0.0772	0.0687	0.0678	0.0676	0.0696	0.0792	0.0832	0.0856
Jain	0.0754	0.0679	0.0687	0.0676	0.0709	0.0710	0.0963	0.0932
Ionosphere	0.0391	0.0360	0.0351	0.0350	0.0357	0.0388	0.4230	0.0394
Heartstatlog	0.0056	0.0014	0.0042	0.0045	0.0027	0.0991	0.1967	0.1327
Haberman	0.1751	0.1622	0.1654	0.1649	0.1661	0.1714	0.1727	0.1743
Glass	0.2849	0.2559	0.2558	0.2559	0.2557	0.2756	0.2634	0.2746
Ecoli	0.1754	0.1334	0.1367	0.1349	0.1371	0.1731	0.1881	0.1793
Diabetes	0.2512	0.2218	0.2344	0.2241	0.2234	0.2534	0.2409	0.2386
CMC	0.4637	0.4519	0.4545	0.4568	0.4542	0.4603	0.4715	0.4664
Cancer	0.0648	0.0415	0.0425	0.0428	0.0421	0.0451	0.0464	0.0446
Bupa	0.3352	0.2914	0.3021	0.3035	0.3165	0.3252	0.3334	0.3012
Blood	0.2194	0.2194	0.2194	0.2194	0.2375	0.2375	0.2294	0.2431
Balancescale	0.2091	0.1980	0.1983	0.1983	0.1988	0.2066	0.2268	0.2188

Table 13 Wilcoxon's rank-sum test for the optimal results on feature selection

Datasets Function	BPIO1	BPIO2	BPIO3	BPIO4	BPIO5	BGWO	BPSO-S
Wine	0.0079	0.7540	0.1270	0.0317	0.0476	0.2143	0.2619
WDBC	0.0317	0.2063	0.1508	0.1667	0.0397	0.3175	0.1587
WBC	0.5794	0.8333	0.5238	0.5873	0.0079	0.8016	0.3810
Vowel	0.3095	1	1	0.8095	0.9524	0.5476	0.4127
Vehicle	0.0079	0.6508	1	0.5397	0.0159	0.2778	0.6905
Thyroid	0.0159	0.4206	0.2222	0.2222	0.0317	0.6905	0.5476
Sonar	0.1667	0.5238	0.5238	0.5238	0.2381	1	1
Seeds	0.0079	0.0952	0.0238	0.0556	0.0159	0.5873	0.0159
Robotnavigation	0.0952	0.2857	0.8889	0.3968	0.4127	0.6349	0.9524
Jain	0.5476	0.4206	0.2222	0.6905	0.3095	0.0556	0.0079
Ionosphere	0	1	1	1	1	1	1
Heartstatlog	0.3175	0.5238	0.9603	0.3333	0.3968	0.9603	0.0476
Haberman	0	1	1	1	1	0.4444	0.4444
Glass	0.0952	0.1746	0.1508	0.0397	0.1508	0.4206	0.4206
Ecoli	0.0476	0.3889	0.4127	0.0873	0.0159	0.0476	0.3889
Diabetes	0.9524	0.6825	0.5159	0.8651	0.0079	0.0952	0.8095
CMC	0.0556	0.3095	0.2222	0.1508	0.0952	0.1508	0.5476
Cancer	0.0159	0.2222	0.0317	0.0079	0.0079	0.7302	0.0317
Bupa	0.0238	0.3571	0.4127	0.3175	0.1429	0.8810	0.0238
Blood	0.0079	0.3095	0.2460	0.5476	0.0079	0.0317	0.3095
Balancescale	0.5000	0.3175	0.6825	0.8889	0.6508	0.7857	0.3175

Where k_1 usually takes 0.99, and k_2 takes 0.01 as know in [57]. SE is the subset feature after feature selection. AL is the characteristic number of the data set. Error$_{(}$KNN) represents the classification error after K-fold cross validation test, which is expressed in (15). The correct and wrong classifications after each classification are respectively indicated by Correct$_{\text{num}}$ and Wrong$_{\text{num}}$.

Table 14 Friedman's test for BPIO on feature selection

Friedman's test	Sum of squares	Degree of freedom	Mean squares	P-value
Wine	0.2992	72.5	39	0.2490
WDBC	0.5257	79.5	39	0.4535
WBC	0.6827	79.5	39	0.5478
Vowel	0.9911	78.5	39	0.2951
Vehicle	0.9876	78	39	0.2960
Thyroid	0.1468	80	39	0.6500
Sonar	0.3783	57	39	0.3643
Seeds	0.9953	80	39	0.2964
Robotnavigation	0.1644	79	39	0.6263
Jain	0.3926	80	39	0.5053
Ionosphere	1	0	39	1
Heartstatlog	0.4448	65	39	0.3942
Haberman	0.9384	52.5	39	0.2156
Glass	0.4779	80	39	0.4732
Ecoli	0.3195	80	39	0.5375
Diabetes	0.1918	80	39	0.6126
CMC	0.8614	80	39	0.3553
Cancer	0.2772	80	39	0.5589
Bupa	0.3482	79.5	39	0.5210
Blood	0.3796	80	39	0.5107
Balancescale	0.8950	78.5	39	0.3379

5.2.3 Result analysis

In the simulation process of BPIO algorithm, BPIO and BPIO1, BPIO2, BPIO3, BPIO4, BPIO5, BGWO, BPSO-S are compared. They ran 20 times on each selected data set, 200 times per iteration, including 150 times in the first phase and 50 times in the second phase. The number of population in each population is 10. The value of K in KNN is 10.

Table 11 is the statistical results obtained using the fitness (14). Acc is the accuracy of classification, and Num is the number of feature subsets selected. For better analysis, Table 12 shows the value of the fitness according to (14).

It can be seen from Table 12 that the algorithms BPIO1 and BPIO3 perform better, and they perform well in 14 and 5 data sets. But in the data sets $Blood$, it can see that the four algorithms perform the same. The same reason may be that these two data sets have only one categories. From Table 11, It can be seen that the performance of the improved four algorithms is higher than the original BPIO accuracy rate. In $Thyroid$, the improved algorithm have higher accuracy rate than original BPIO. This is also the only data set where the improved algorithm does not perform well. In $Jain$, $Ionosphere$, $Heartstatlog$ and $Ecoli$, the performance of the improved algorithm is very excellent. Compared with the original algorithm, the accuracy of feature selection is basically improved by about 5%.

It can be seen from Tables 13 and 14, that the improved algorithm and comparison algorithm in this paper are statistical results of Wilcoxon's rank-sum test and Friedman's test. Table 13 shows that in feature selection Datasets, other algorithms are compared with BPIO. BPIO1 has 10, BPIO2 has 0, BPIO3 has 2, BPIO4 has 3, BPIO5 has 10, BGWO has 2 and BPSO-S There are 5 overall differences that are not significant. It can be seen that the data in the Table 14 are all greater than 5%, so it can be seen through the Friedman's test that there is no significant difference in all the compared algorithms.

6 Conclusions

This paper presents several new and improved binary PIO algorithm. More importantly, the conversion function is the key to turning PIO into a binary algorithm. This paper introduces four new transfer function to improve the solution performance of the algorithm. Subsequently, the update equation was modified according to the characteristics of the update equation of the algorithm, combined with the image of the conversion function. Finally, tested on 23 well-known benchmark functions, including BPIO, BPIO1, BPIO2, BPIO3, BPIO4, BPIO5, BGWO, and BPSO-S. Simulation experiments show that the improvement of BPIO proposed in this paper has achieved good results. The relationship between the speed update equation and the newly proposed transfer function is discussed, and the modification of the update equation is also important. In addition, this paper applies the improved algorithm to feature selection. The simulation results of feature selection show that the improved algorithm is more competitive in terms of classification accuracy and the number of selected features. For further research, the algorithm will be applied to a wider range of application scenarios in the future, such as the $0-1$ knapsack problem and task scheduling problem. It can better prove that the algorithm improvement is excellent and competitive. Finally, the performance of the algorithm in the low dimension is not very good, which is related to the separate results of the exploration and exploitation of the algorithm, and the focus can be placed here in future research.

Acknowledgements This paper was supported by National Natural Science Foundation of China with grant number NSF 61872085, Natural Science Foundation of Fujian Province with grant number 2018J01638, and project 2018Y3001 of Fujian Provincial Department of Science and Technology.

Declarations

Competing interests The authors declare that they have no conflict of interest.

References

1. Smith AE (2000) Swarm intelligence: from natural to artificial systems [book reviews]. IEEE Trans Evol Comput 4(2):192–193. https://doi.org/10.1109/TEVC.2000.850661
2. Slowik A, Kwasnicka H (2017) Nature inspired methods and their industry applications-swarm intelligence algorithms. IEEE Trans Ind Inf 14(3):1004–1015. https://doi.org/10.1109/TII.2017.2786782
3. Xue B, Zhang M, Browne WN, Yao X (2016) A survey on evolutionary computation approaches to feature selection. IEEE Trans Evol Comput 20(4):606–626. https://doi.org/10.1109/TEVC.2015.2504420
4. Mavrovouniotis M, Li C, Yang S (2017) A survey of swarm intelligence for dynamic optimization: Algorithms and applications. Swarm Evol Comput 33:1–17. https://doi.org/10.1016/j.swevo.2016.12.005
5. Hinchey MG, Sterritt R, Rouff C (2007) Swarms and swarm intelligence. Computer 40(4):111–113. https://doi.org/10.1109/MC.2007.144
6. Abraham A, Guo H, Liu H (2006) Swarm intelligence: foundations, perspectives and applications. In: Swarm intelligent systems. Springer, pp 3–25
7. Sörensen K (2015) Metaheuristics' the metaphor exposed. Int Trans Oper Res 22(1):3–18. https://doi.org/10.1111/itor.12001
8. Chu S-C, Huang H-C, Roddick JF, Pan J-S (2011) Overview of algorithms for swarm intelligence. In: International Conference on Computational Collective Intelligence. Springer, pp 28–41
9. Krause J, Ruxton GD, Krause S (2010) Swarm intelligence in animals and humans. Trends Ecol Evol 25(1):28–34. https://doi.org/10.1016/j.tree.2009.06.016

10. Chandra DK, Ravi V (2009) Feature selection and fuzzy rule-based classifier applied to bankruptcy prediction in banks. Int J Inf Decis Sci 1(4):343–365. https://doi.org/10.1504/IJIDS.2009.027756
11. Chen S-M, Chang Y-C, Pan J-S (2013) Fuzzy rules interpolation for sparse fuzzy rule-based systems based on interval type-2 gaussian fuzzy sets and genetic algorithms. IEEE Trans Fuzzy Syst 21(3):412–425. https://doi.org/10.1109/TFUZZ.2012.2226942
12. McCulloch WS, Pitts W (1943) A logical calculus of the ideas immanent in nervous activity. Bullet Math Biophys 5(4):115–133. https://doi.org/10.1007/BF02478259
13. Chaotao Chen JR (2019) Gated recurrent neural network with sentimental relations for sentiment classification. Inf Sci 502:268–278. https://doi.org/10.1016/j.ins.2019.06.050
14. Xu Y, Han J, Wang E, Ming J, Xiong H, Yang Y (2019) Slanderous user detection with modified recurrent neural networks in recommender system. Inf Sci 505:265–281. https://doi.org/10.1016/j.ins.2019.07.081
15. Bäck T, Fogel DB, Michalewicz Z (1997) The handbook of evolutionary computation. Release 97(1):B1
16. Canayaz M, Karci A (2016) Cricket behaviour-based evolutionary computation technique in solving engineering optimization problems. Appl Intell 44(2):362–376. https://doi.org/10.1007/s10489-015-0706-6
17. Robles-Berumen H, Zafra A, Fardoun HM, Ventura S (2019) Leac: An efficient library for clustering with evolutionary algorithms. Knowl-Based Syst 179:117–119. https://doi.org/10.1016/j.knosys.2019.05.008
18. Holland JH (1973) Genetic algorithms and the optimal allocation of trials. SIAM J Comput 2(2):88–105. https://doi.org/10.1137/0202009
19. Sayed S, Nassef M, Badr A, Farag I (2019) A nested genetic algorithm for feature selection in high-dimensional cancer microarray datasets. Expert Syst Appl 121:233–243. https://doi.org/10.1016/j.eswa.2018.12.022
20. Song Y, Wang F, Chen X (2019) An improved genetic algorithm for numerical function optimization. Appl Intell 49(5):1880–1902. https://doi.org/10.1007/s10489-018-1370-4
21. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks, vol 4. IEEE, pp 1942–1948
22. Chen K, Zhou F-Y, Yuan X-F (2019) Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection. Expert Syst Appl 128:140–156. https://doi.org/10.1016/j.eswa.2019.03.039
23. Chen K, Zhou F, Liu A (2018) Chaotic dynamic weight particle swarm optimization for numerical function optimization. Knowl-Based Syst 139:23–40. https://doi.org/10.1016/j.knosys.2017.10.011
24. Storn R, Price K (1997) Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359. https://doi.org/10.1023/A:1008202821328
25. Hancer E, Xue B, Zhang M (2018) Differential evolution for filter feature selection based on information theory and feature ranking. Knowl-Based Syst 140:103–119. https://doi.org/10.1016/j.knosys.2017.10.028
26. Wang S, Li Y, Yang H (2017) Self-adaptive differential evolution algorithm with improved mutation mode. Appl Intell 47(3):644–658. https://doi.org/10.1007/s10489-017-0914-3
27. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. J Glob Optim 39(3):459–471. https://doi.org/10.1007/s10898-007-9149-x
28. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical Report, Technical report-tr06, Erciyes university, engineering faculty, computer
29. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
30. Pan J-S, Hu P, Chu S-C (2019) Novel parallel heterogeneous meta-heuristic and its communication strategies for the prediction of wind power. Processes 7(11):845. https://doi.org/10.3390/pr7110845
31. Duan H, Qiao P (2014) Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. International journal of intelligent computing and cybernetics. https://doi.org/10.1108/IJICC-02-2014-0005
32. Tian A-Q, Chu S-C, Pan J-S, Liang Y (2020) A novel pigeon-inspired optimization based mppt technique for pv systems. Processes 8(3):356. https://doi.org/10.3390/pr8030356
33. Nguyen T-T, Dao T-K, Sung T-W, Ngo T-G, Pan J-S (2020) Pigeon inspired optimization for node location in wireless sensor network. Int Conf Eng Res Appl 104:589–598. https://doi.org/10.1007/978-3-030-37497-6_67
34. Elawady RM, Barakat S, Elrashidy NM (2014) Different feature selection for sentiment classification. Int J Inf Sci Intell Syst 3(1):137–150
35. Larry A, Rendell KK (1992) The feature selection problem: Traditional methods and a new algorithm. AAAI, vol 2, pp 129–134
36. Singh D, Singh B (2019) Hybridization of feature selection and feature weighting for high dimensional data. Appl Intell 49(4):1580–1596. https://doi.org/10.1007/s10489-018-1348-2
37. Onan A, Korukoğlu S (2017) A feature selection model based on genetic rank aggregation for text sentiment classification. J Inf Sci 43(1):25–38. https://doi.org/10.1177/0165551515613226
38. Yang D-H, Yu G (2013) A method of feature selection and sentiment similarity for chinese micro-blogs. J Inf Sci 39(4):429–441. https://doi.org/10.1177/0165551513480308
39. Gao W, Hu L, Zhang P (2020) Feature redundancy term variation for mutual information-based feature selection. Appl Intell 50(4):1272–1288. https://doi.org/10.1007/s10489-019-01597-z
40. Wang X, Yang J, Teng X, Xia W, Jensen R (2007) Feature selection based on rough sets and particle swarm optimization. Pattern Recogn Lett 28(4):459–471. https://doi.org/10.1016/j.patrec.2006.09.003
41. Dornaika F (2020) Multi-layer manifold learning with feature selection. Appl Intell 50:1859–1871. https://doi.org/10.1007/s10489-019-01563-9
42. Maza S, Touahria M (2019) Feature selection for intrusion detection using new multi-objective estimation of distribution algorithms. Appl Intell 49(12):4237–4257. https://doi.org/10.1007/s10489-019-01503-7
43. Cui Z, Zhang J, Wang Y, Cao Y, Cai X, Zhang W, Chen J (2019) A pigeon-inspired optimization algorithm for many-objective optimization problems. Sci China Inf Sci 62:070212:1–070212:3
44. Duan H, Wang X (2015) Echo state networks with orthogonal pigeon-inspired optimization for image restoration. IEEE Trans Neural Netw Learn Syst 27(11):2413–2425. https://doi.org/10.1109/TNNLS.2015.2479117
45. Bolaji AL, Babatunde BS, Shola PB (2018) Adaptation of binary pigeon-inspired algorithm for solving multidimensional knapsack problem. In: Soft Computing: Theories and Applications, vol 583. Springer, pp 743–751
46. Yang Z, Liu K, Fan J, Guo Y, Niu Q, Zhang J (2019) A novel binary/real-valued pigeon-inspired optimization for economic/environment unit commitment with renew-

ables and plug-in vehicles. Sci China Inf Sci 62(7):70213. https://doi.org/10.1007/s11432-018-9730-4
47. Bolaji AL, Okwonu FZ, Shola PB, Balogun BS, Adubisi OD (2020) A modified binary pigeon-inspired algorithm for solving the multi-dimensional knapsack problem. J Intell Syst 30(1):90–103. https://doi.org/10.1515/jisys-2018-0450
48. Tian A-Q, Chu S-C, Pan J-S, Cui H, Zheng W-M (2020) A compact pigeon-inspired optimization for maximum short-term generation mode in cascade hydroelectric power station. Sustainability 12(3):767. https://doi.org/10.3390/su12030767
49. Zheng H, Wei C (2020) Binary pigeon-inspired optimization for quadrotor swarm formation control. Adv Swarm Intell:71–82. https://doi.org/10.1007/978-3-030-53956-6_7
50. Dou R, Duan H (2016) Pigeon inspired optimization approach to model prediction control for unmanned air vehicles. Aircraft Eng Aerosp Technol Int J 88(1):108–116. https://doi.org/10.1108/AEAT-05-2014-0073
51. Hu P, Pan J-S, Chu S-C (2020) Improved binary grey wolf optimizer and its application for feature selection. Knowl-Based Syst:105746. https://doi.org/10.1016/j.knosys.2020.105746
52. Beheshti Z (2020) A time-varying mirrored s-shaped transfer function for binary particle swarm optimization. Inf Sci 512:1503–1542. https://doi.org/10.1016/j.ins.2019.10.029
53. Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Stat Assoc 32(200):675–701. https://doi.org/10.1080/01621459.1937.10503522
54. Friedman M (1940) A comparison of alternative tests of significance for the problem of m rankings. Ann Math Stat 11(1):86–92. https://doi.org/https://www.jstor.org/stable/2235971
55. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18. https://doi.org/10.1016/j.swevo.2011.02.002
56. Dua D, Graff C (2017) UCI machine learning repository University of California, Irvine, School of Information and Computer Sciences. http://archive.ics.uci.edu/ml
57. Emary E, Zawbaa HM, Hassanien AE (2016) Binary grey wolf optimization approaches for feature selection. Neurocomputing 172:371–381. https://doi.org/10.1016/j.neucom.2015.06.083

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Jeng-Shyang Pan received the B.S. degree in electronic engineering from the National Taiwan University of Science and Technology in 1986, the M.S. degree in communication engineering from National Chiao Tung University, Taiwan, in 1988, and the Ph.D. degree in electrical engineering from the University of Edinburgh, U.K., in 1996. He is currently the Director of the Fujian Provincial Key Lab of Big Data Mining and Applications, and an Assistant President with the Fujian University of Technology. He is also the Professor with the Harbin Institute of Technology. He is the IET Fellow, U.K., and has been the Vice Chair of the IEEE Tainan Section. He was offered Thousand Talent Program in China in 2010.

Ai-Qing Tian received his B.S. degree from Taishan College of Science and Technology, Taian ,China, in 2019. He is currently pursuing the master degree with the Shandong University of Science and Technology, Qingdao, China. His recent research interests are swarm intelligence and artificial neural networks.

Shu-Chuan Chu received the Ph.D. degree in 2004 from the School of Computer Science, Engineering and Mathematics, Flinders University of South Australia. She joined Flinders University in December 2009 after 9 years at the Cheng Shiu University, Taiwan. She is the Research Fellow in the College of Science and Engineering of Flinders University, Australia from December 2009. Currently, She is the Research Fellow with PhD advisor in the College of Computer Science and Engineering of Shandong University of Science and Technology from September 2019. Her research interests are mainly in Swarm Intelligence, Intelligent Computing and Data Mining.

Jun-Bao Li received the Ph.D. degree in pattern recognition from Harbin Institute of Technology, China, in 2008. He is the full professor at School of Electronics and Information Engineering, Harbin Institute of Technology. His research interests include Deep learning, kernel learning and applications (Kernel Optimization; Kernel Sparse Learning; Kernel Manifold Learning; Supervised Kernel Self-adaptive Learning and Applications of Kernel Learning) and Medical image processing and classification (Segmentation, enhance and feature extraction of medical image; Mammogram based breast cancer diagnosis). He has authored more than 80 papers including 60 SCI journals in pattern recognition areas.