IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 26, 2018, accepted July 8, 2018, date of publication July 25, 2018, date of current version September 5, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2858441

A Self-Adaptive Fireworks Algorithm for Classification Problems

YU XUE[1,2], BINPING ZHAO[1,2], TINGHUAI MA[1], AND WEI PANG[3,4]

[1]School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China
[2]Jiangsu Engineering Research Center of Communication and Network Technology, Nanjing University of Posts and Telecommunications, Nanjing 210044, China
[3]School of Natural and Computing Sciences, University of Aberdeen, Aberdeen AB24 3UE, U.K.
[4]Shaanxi Key Laboratory of Complex System Control and Intelligent Information Processing, Xi'an University of Technology, Xi'an 710048, China

Corresponding author: Yu Xue (xueyu@nuist.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grants 61403206 and 61771258, in part by the Natural Science Foundation of Jiangsu Province under Grants BK20141005 and BK20160910, in part by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China under Grant 14KJB520025, in part by the Priority Academic Program Development of Jiangsu Higher Education Institutions, in part by the Open Research Fund of Jiangsu Engineering Research Center of Communication and Network Technology, NJUPT, under Grant JSGCZX17001, and in part by the Shaanxi Key Laboratory of Complex System Control and Intelligent Information Processing, Xi'an University of Technology, under Contract SKL2017CP01.

ABSTRACT Fireworks algorithm (FWA) is a novel swarm intelligence algorithm recently proposed for solving complex optimization problems. Because of its powerful global optimization ability to solve classification problems, we first present an optimization classification model in this paper. In this model, a linear equation set is constructed according to classification problems. This optimization classification model can be solved by most evolutionary computation techniques. In this paper, a self-adaptive FWA (SaFWA) is developed so that the optimization classification model can be solved efficiently. In SaFWA, four candidate solution generation strategies (CSGSs) are employed to increase the diversity of solutions. In addition, a self-adaptive search mechanism has also been introduced to use the four CSGSs simultaneously. To extensively assess the performance of SaFWA on solving classification problems, eight datasets have been used in the experiments. The experimental results show that it is feasible to solve classification problems through the optimization classification model and SaFWA. Furthermore, SaFWA performs better than FWA, FWA variants with only one CSGS, particle swarm optimization, and differential evolution on most of the training sets and test sets.

INDEX TERMS Classification, evolutionary classification algorithm, fireworks algorithm (FWA), self-adaptive, optimization.

I. INTRODUCTION

Classification is a typical task in machine learning. Its aim is to predict the labels of unseen instances using a model which is learned from a training sets [1], [2]. Classification methods, such as support vector machine (SVM) [3], [4], k-nearest neighbor (KNN) [5], [6], decision tree (DT) [7], [8], artificial neural network (ANN) [9], [10], and naive Bayesian classification (NBC) [11], [12] have been extensively studied and applied in the past several decades. However, many of the existing classification algorithms are deterministic, and some of them can easily trap into local optima. The global optimal solution cannot be found quickly. The existing classifiers will end up searching after the local optimal solution which will be mistaken as the global optimal solution. Evolutionary computation (EC) techniques have excellent global search ability, but in the past several decades, EC techniques were only used to improve the performance of classifiers by optimizing their parameters, structures, or inputs [13]–[16]. For example, Chen *et al.* [13] designed an integrated hybrid algorithm for training radial basis function neural network (RBFNN). The proposed integrated approach IPGO was composed of two approaches: particle swarm optimization (PSO) and genetic algorithm (GA), and it combines both their advantages to improve the learning performance of RBFNN. The IPGO algorithm with PSO-based and GA-based approaches had shown promising results in some benchmark problems. Besides, PSO was employed by Xue *et al.* [16] to optimize the inputs of classification methods. In their methods, three new initialization strategies and three new personal best and global best updating mechanisms were proposed to

develop novel feature selection approaches with the goals of maximizing the classification performance, minimizing the number of features, and reducing the computational time.

Xue *et al.* [17] proposed an optimization model for classification problems, and they used fireworks algorithm (FWA) to solve the optimization classification problem. The work of Xue *et al.* [17] has proved that EC techniques can be used to solve classification problems directly. However, the number of datasets employed in [17] is limited, and the employed EC algorithm is relatively simple. In addition, FWA is not very flexible, in particular, it does not utilize more information about other good solutions in the swarm. In other words, the individuals are not well-informed by the whole swarm. Thus, many researchers have designed different methods to improve the performance of FWA [18]–[21]. However, in the existing research work, the performance of FWA is improved usually by modifying its parameters or operators. For example, Yu *et al.* [18] proposed a FWA with differential evolution (DE) mutation operator (FWA-DM), which replaces the fireworks algorithm's Gaussian mutation operator by a differential mutation operator. In addition, Zheng *et al.* [19] improved FWA by introducing DE operators. The new algorithm increases the degree of information sharing among the fireworks and sparks, and diversifies the search process.

Self-adaptive mechanism is useful for developing EC techniques, and many effective self-adaptive EC techniques have been proposed in recent years. Many researchers have introduced self-adaptive mechanisms into EC techniques and they have achieved better results [22]–[29]. For example, Fan and Yan [22] proposed a DE algorithm with self-adaptive mutation strategy and control parameters (SSCPDE) to optimize the operating conditions. Simulation results show that the performance of SSCPDE is better than that of the other 6 state-of-the-art DE variants. Banitalebi *et al.* [23] proposed a new self-adaptive binary DE algorithm (SabDE). SabDE was compared to some existing state-of-the-art algorithms on a set of benchmark problems, and their results revealed that the proposed algorithm was superior to the existing algorithms on those problems. Additionally, Xue *et al.* [24] proposed an ensemble of evolution algorithm based on self-adaptive learning population search techniques (EEA-SLPS) to solve the numerical optimization problems, and the experimental results indicate that the universality and robustness performance of EEA-SLPS is better than other state-of-the-art algorithms. EC techniques with self-adaptive mechanism have usually achieved excellent performance. Inspired by the powerful ability of self-adaptive mechanism, we introduce this mechanism into FWA for optimization classification problems in this paper.

In this research, first, we present an optimization classification model for classification problems. Through this model, classification problems can be solved by EC techniques. Second, we propose a self-adaptive fireworks algorithm (SaFWA). Four efficient candidate solution generation strategies (CSGSs) from differential evolution are employed in SaFWA. The purpose of this research is to extensively investigate the performance of SaFWA when it is employed in classification problems. This is achieved as follows: we first convert classification problems into optimization problems, then we use SaFWA to solve the optimization problems. Some preliminary work has been presented in [17] and [30]. To further investigation, in the experiments, eight different data sets from UCI Machine Learning Repository are employed, and the proposed algorithm is compared with many other state-of-the-art algorithms.

The rest of the paper is organized as follows: Section II gives an outline of the FWA algorithm. Section III focuses on the self-adaptive mechanism and four CSGSs. Section IV presents the optimization classification model. Section V describes the experimental design. Section VI presents the experimental results and analysis. Section VII concludes the study and provides an insight into the future trends.

II. RELATIVE WORK

FWA is a new swarm intelligent algorithm which was proposed in 2010 by Tan and Zhu [31] and Guendouz *et al.* [32] for complex optimization problems. It searches for an optimal point in the search space by iterating the explosion operation and the selection operation.

The process of FWA is presented as follows: At first, n fireworks are initialized randomly, and their qualities (fitness) are evaluated to determine the explosion amplitudes and the numbers of sparks. Subsequently, the fireworks explode and generate sparks within their local space. To ensure diversity and balance of the global and local search, the explosion amplitudes and the population sizes of the newly generated explosion sparks differ among the fireworks. A firework with better fitness can generate a larger population of explosion sparks within a smaller range (explosion amplitude). On the contrary, a firework with worse fitness value can only generate a smaller population within a larger range (explosion amplitude). This technique allows the balance between exploration and exploitation capabilities during the search process. After the explosion, another type of sparks is generated by Gaussian mutation operator. The idea behind this is to further ensure diversity of the swarm. At last, individuals from the current generation of fireworks and sparks are selected to enter into the next generation.

For this explosion operator, the explosion amplitudes and the numbers of explosion sparks are two important factors, and they are respectively defined as follows.

$$s_i = m \cdot \frac{y_{\max} - f(x_i) + \xi}{\sum_{i=1}^{n} (y_{\max} - f(x_i)) + \xi} \tag{1}$$

$$A_i = \hat{A} \cdot \frac{f(x_i) - y_{\min} + \xi}{\sum_{i=1}^{n} (f(x_i) - y_{\min}) + \xi} \tag{2}$$

where m and $\hat{A}$ is a parameter controlling the total number of sparks and the maximum explosion amplitude generated by

the n fireworks, $f(x_i)$ is the function value of x_i, $y_{\max}$ and $y_{\min}$ are respectively the maximum and minimum fitness values among the n fireworks, and ξ, which denotes the smallest constant in the computer, is utilized to avoid zero-division-error.

The selection probability of a location x_i is defined as follows.

$$p(x_i) = \frac{L(x_i)}{\sum_{j \in K} L(x_j)} \tag{3}$$

where $L(x_i)$ denotes the general distance between a location x_i and other locations.

The framework of the FWA is described in Algorithm 1.

Algorithm 1 The Framework of FWA

Input: input parameters including maximum explosion amplitude m, total number of sparks $\hat{A}$
Output: the best location and its fitness value.
Randomly initialize n locations to set off fireworks and evaluate their fitness;
while *stop criteria == false* **do**
  foreach *firework* x_i **do**
    Calculate the number of sparks that the firework yields: $\hat{s}_i$ as Eq.(1);
    Obtain locations of $\hat{s}_i$ sparks of the firework x_i according to Eq.(2);
  end
  for *k=1:n* **do**
    Generate a specific spark for the firework x_k with Gaussian explosion;
  end
  Evaluate each location and store the best location and its fitness value;
  Select the best location and keep it for next explosion generation;
  Randomly select n-1 locations from the two types of sparks and the current fireworks according to the probability given in Eq.(3);
end

III. SELF-ADAPTIVE FIREWORKS ALGORITHM

A. SELF-ADAPTIVE MECHANISM

Obviously, a fireworks algorithm with only single CSGS cannot meet demands of solving a wide range of practical problems because different problems have their specific characteristics. Meanwhile, many existing EC methods may still suffer from the problems of getting trapped into local optima. Moreover, it is time-consuming to select a suitable EC method manually for a classification problem. Thus, a self-adaptive mechanism and several CSGSs are employed in SaFWA. Different CSGSs are suitable for different problems, and they can also increase the diversity of solutions. The self-adaptive mechanism can dynamically choose the best CSGS for the corresponding problem during the search process of optimization classification problems, which can improve the universality and robustness of the algorithm.

The process of self-adaptive mechanism is described as follows: Four CSGSs are used in a strategy pool, and each CSGS has its selection probability value (P). *straNum* represents the number of CSGSs. It is noted that different number of CSGSs may be used depending on problem characteristics. At the initialization phase, the selection probability value (P) for each CSGS is the same, and it is set to be the reciprocal of *straNum* (1/4). Each individual is assigned a CSGS, which is chosen from the strategy pool randomly. The strategy which is selected from the strategy pool is denoted as *curStra*. The new individual, which is produced using *curStra*, will be evaluated. Then, the new individual and the previous one are compared against each other. If the fitness of the new individual is better than that of the previous one, the strategy flag success matrix (*straFlagS*) of *curStra* will be updated. Otherwise, the strategy flag failure matrix (*straFlagF*) will be updated. At the beginning of each iteration, the *straFlagS* and *straFlagF* of all the strategies are set to 0. The information stored in *straFlagS* and *straFlagF* will be counted in total flag success matrix (*totalFlagS*) and total flag failure matrix (*totalFlagF*), respectively. Moreover, both *straFlagS* and *straFlagF* are initialized to 0 for the next generation. When the number of iteration reaches the learning period (*LP*), new P of each CSGS will be produced based on Equations (4) and (5).

$$P_q{}' = \begin{cases} \sum_{n=1}^{LP} totalFlagS_n / (\sum_{n=1}^{LP} totalFlagS_n \\ + \sum_{n=1}^{LP} totalFlagF_n), \ \sum_{n=1}^{LP} totalFlagS_n \neq 0 \\ (\sum_{n=1}^{LP} totalFlagS_n + \varepsilon) / \sum_{n=1}^{LP} totalFlagF_n, \\ \sum_{n=1}^{LP} totalFlagS_n = 0 \end{cases} \tag{4}$$

$$P = P_q{}' / \sum_{q=1}^{straNum} P_q{}' \tag{5}$$

where $q \in \{1, 2, ..., straNum\}$, ε represents a small constant closed to 0, $\sum_{n=1}^{LP} totalFlagS_n$ and $\sum_{n=1}^{LP} totalFlagF_n$ represents the number of successful and failed operation during the evolutionary process, P is the new selection probabilities of all strategies.

B. CANDIDATE SOLUTION GENERATION STRATEGIES (CSGSS)

The CSGSs greatly affect the efficiency of SaFWA. 4 CSGSs [33] are used in the strategy pool. These strategies are the most popular DE strategy variants [34]–[36], and all of them have achieve great performance evidenced by previous applications.

1) DE/*rand*/1 (CSGS1): The new mutation vector is generated by using a random individual and a difference vector that mutates two random individuals.

$$V_i^{G+1} = X_{r1}^G + F(X_{r2}^G - X_{r3}^G) \tag{6}$$

2) DE/*rand*/2 (CSGS2): The new mutation vector is generated by using a random individual and a difference vector that mutates four random individuals.

$$V_i^{G+1} = X_{r1}^G + F(X_{r2}^G - X_{r3}^G) + F(X_{r4}^G - X_{r5}^G) \tag{7}$$

3) DE/*best*/2 (CSGS3): The new mutation vector is generated by using the best individual of the current generation and a difference vector that mutates four random individuals.

$$V_i^{G+1} = X_{best}^G + F(X_{r1}^G - X_{r2}^G) + F(X_{r3}^G - X_{r4}^G) \tag{8}$$

4) DE/*current-to-best*/2 (CSGS4): The new mutation vector is generated by using the current individual and a difference vector that mutates the best individual of the current generation, the current individual, and four random individuals.

$$V_i^{G+1} = X_i^G + F(X_{best}^G - X_i^G) + F(X_{r1}^G - X_{r2}^G) + F(X_{r3}^G - X_{r4}^G) \tag{9}$$

In the above four equations (Eqs 6-9), G is the number of current generation, $G+1$ is the next generation, X_{r_1}, X_{r_2}, X_{r_3}, X_{r_4} are different individuals, parameter $F \in [-1, 1]$. X_i^G is the current individual, and V_i^{G+1} is the newly produced individual. X_{best} is the individual with the best fitness value.

According to the strategy selection probability P, one strategy can be chosen through the roulette wheel algorithm during evolutionary process. At first, the selection probability value (P) for each CSGS is the same, and it is set to the reciprocal of *straNum*. After that, the strategy which achieves better performance will be selected with higher possibility by the self-adaptive mechanism. This can greatly improve the ability of SaFWA when solving different kinds of problems.

The datails of SaFWA is described in Algorithm 2.

Algorithm 2 SaFWA

Input: Set parameter values including max number of fitness evaluations (*MaxFES*), current number of fitness evaluations (*fitCount* = 0), *straNum*, $p_q = 1/straNum$ for each $q \in straNum$, $LP = 10$, *straFlagS*, *straFlagF*, *totalFlagS*, *totalFlagF*, *curIter* = 0, *flagIter* = 0.

Output: the best location

```
Randomly initialize n locations to set off fireworks and evaluate their fitness;
while fitCount < MaxFES do
    foreach firework x_i do
        Calculate the number of sparks that the firework yields: ŝ_i by Eq.(1);
        Obtain locations of ŝ_i sparks of the firework x_i by Eq.(2);
    end
    foreach i < n + s_1 + ... + s_n do
        Select one CSGS for current solution x_i from the strategy pool by roulette wheel selection method based on p_1,p_2,...,p_straNum. curStra is selected. Generate a new solution x_i^new by the selected CSGS, and calculate its fitness value;
        if f(x_i^new) is better than f(x_i) then
            straFlagS_{i,curStra} = 1;
            Replace x_i with x_i^new ;
        else
            straFlagF_{i,curStra} = 1;
        end
        fitCount=fitCount+1;
    end
    curIter = curIter+1;
    Update the totalFlagS and totalFlagF with the sum of all the rows in straFlagS and straFlagF, and reset straFlagS and straFlagF to be all-zero matrix;
    if (curIter − flagIter) == LP then
        flagIter = curIter;
        Update {p_1, p_2, ..., p_curaNum} based on totalFlagS and totalFlagF according to Eqs. (4) and (5);
        Reset totalFlagS and totalFlagF;
    end
    Evaluate each location and store the best location and its fitness value;
    Select the best location and n − 1 locations by the roulette wheel algorithm and keep them for the next generation according to the probability given in Eq.(3);
end
```

IV. OPTIMIZATION CLASSIFICATION MODEL AND OBJECTIVE FUNCTION

For a data set D, 70% of data in D is selected as the training set T. The examples of training data can be written as

$$\begin{bmatrix} x_{11}, x_{12}, ..., x_{1d}, y_1 \\ x_{21}, x_{22}, ..., x_{2d}, y_2 \\ ... \\ x_{m1}, x_{m2}, ..., x_{md}, y_m \end{bmatrix} \tag{10}$$

where (x_i, y_i) is the i^{th} example, x_i=$x_{i1}, x_{i2}, ..., x_{id}$ is the i^{th} sample, $y_i \in \{1, 2, \cdots, l\}$ $(i = 1, 2, \cdots, m)$ is the label of the i^{th} sample.

First, we introduce a weight vector $W = (w_1, w_2, ..., w_d)$, and set up Eq. (11):

$$\begin{cases} w_1x_{11} + w_2x_{12} + ... + w_dx_{1d} = y_1 \\ w_1x_{21} + w_2x_{22} + ... + w_dx_{2d} = y_2 \\ ... \\ w_1x_{m1} + w_2x_{m2} + ... + w_dx_{md} = y_m \end{cases} \tag{11}$$

From Eq. (11), we can see that if a solution for Equation (11) can be found, the label of x_i could be predicted according to this solution. Obviously, this kind of problems can be solved efficiently by other EC techniques [30], [37].

Let

$$A = \begin{bmatrix} x_{11}, x_{12}, ..., x_{1D} \\ x_{21}, x_{22}, ..., x_{2D} \\ ... \\ x_{m1}, x_{m2}, ..., x_{mD} \end{bmatrix} \quad and \; Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \tag{12}$$

Thus, Equation (11) can be abbreviated as $A \cdot W^T = Y$. We then present the following theorem:

Theorem 1: If the equation set has no solution, the necessary and sufficient condition is $R(A) < R(A, Y)$. If the equation set has a unique solution, the necessary and sufficient condition is $R(A) = R(A, Y) = D$. If the equation set has many solutions, the necessary and sufficient condition is $R(A) = R(A, Y) < D$.

$R(A)$ is the rank of matrix A. In most cases, the number of example is far greater than the number of dimensions of an example. So, Equation (11) will be an inconsistent equation set with high probability. And then, in most cases, no exact solution for the inconsistent equation set will be calculated. However, this is an optimization classification problem, and it is not necessary to find an exact solution for this set of equations. In fact, for a classification problem, it is sufficient to find approximate solutions for the following equations:

$$A \cdot W^T \approx Y \cong \begin{cases} w_1x_{11} + w_2x_{12} + ... + w_Dx_{1D} \approx y_1 \\ w_1x_{21} + w_2x_{22} + ... + w_Dx_{2D} \approx y_2 \\ ... \\ w_1x_{m1} + w_2x_{m2} + ... + w_Dx_{mD} \approx y_m \end{cases} \tag{13}$$

In this process, we can predict the label of x_i belonging to y_i if $y_i - \delta \leq w_1x_{i1} + w_2x_{i2} + ... + w_dx_{id} < y_i + \delta$, ($y_i \in \{1, 2, \cdots, l\}$), where δ is a small threshold, it will be described that how to generate this value.

$$\begin{cases} y_1 - \delta \leq w_1x_{11} + w_2x_{12} + ... + w_dx_{1d} < y_1 + \delta \\ y_2 - \delta \leq w_1x_{21} + w_2x_{22} + ... + w_dx_{2d} < y_2 + \delta \\ ... \\ y_m - \delta \leq w_1x_{m1} + w_2x_{m2} + ... + w_dx_{md} < y_m + \delta \end{cases} \tag{14}$$

The above is a continuous numerical optimization problem, and obviously, EC techniques can be employed to solve this kind of problems. The objective function can be defined as follow:

$$\min(f(W) = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{d} (w_j \cdot x_{ij} - y_i)^2}) \tag{15}$$

In this research, we estimate the lower and upper boundaries of w_i, $i = 1, 2, ..., d$ by the following equations:

$$\pm\sigma \frac{\sum_{i=1}^{N} y_i}{\sum_{i=1}^{N} \sum_{j=1}^{d} x_{ij}} \tag{16}$$

where σ is a control parameter which can adjust the range of the boundaries. Finally, we acknowledge that a more representitative non-linear model may be used instead of Equation (11) to further improve the learning performance, but this is beyond the scope of this research and we will explore this in our future work.

V. EXPERIMENTAL DESIGN

A. DATASETS

Eight different datasets were used in the experiments. The datasets were chosen from the UCI Machine Learning Repository [38]. The detailed information of all the datasets is presented in Table 1. The datasets have various numbers of features, classes and samples. For each dataset, examples were randomly divided into two sections: 70% of them were used as the training sets and the rest were used as the test sets.

TABLE 1. Description of data.

Datasets	Examples	Features	Labels
Biodegradation	1055	41	2
Climate model	540	20	2
Fertility	100	9	2
German	1000	24	2
Ionosphere	351	33	2
Iris	150	4	3
Spect heart	267	44	2
WBCD	569	30	2

B. PARAMETER SETTINGS

Eight different algorithms were chosen for experiments. They were standard FWA, FWA with CSGS1 (FWA-CSGS1), FWA with CSGS2 (FWA-CSGS2), FWA with CSGS3 (FWA-CSGS3), FWA with CSGS4 (FWA-CSGS4), SaFWA, DE and PSO [39], [40]. All the algorithms were employed to find W to minimize $\sqrt{\sum_{i=1}^{m} (\sum_{j=1}^{d} w_j \cdot x_{ij} - y_i)^2}$ on each dataset. For SaFWA, standard FWA and FWA variants with single CSGS, each algorithm ran 26 times on each dataset, and the maximum number of fitness evaluations (*NFE*) was set up to 100,000. The other parameter settings of FWA were: $n = 10$, $m = 90$, $a = 0.04$, $b = 0.8$, $\hat{A} = 2$, $\hat{m} = 8$. $LP = 10$, and $straNum = 4$.

After W was found, we calculated the classification accuracy for each example as follow: (x_i, y_i), if $-0.5 \leq (W^T \cdot x_i - y_i) < +0.5$, then we deemed the class label of

TABLE 2. Classification accuracies of eight algorithms on training sets.

Datasets	DE	PSO	FWA-CSGS1	FWA-CSGS2	FWA-CSGS3	FWA-CSGS4	standard FWA	SaFWA
	Mean±Std	Mean±Std	Mean±Std	Mean±Std	Mean±Std	Mean±Std	Mean±Std	Mean±Std
Biodegradation	0.624 ±0.252	0.483 ±1.100	0.8043 ±0.04879	0.8042± 0.04239	0.8036 ±0.04465	0.8043 ±0.0376	0.7566 ±0.07067	**0.8053** ±0.04899
Climate model	0.910 ±0.024	0.773 ±0.767	0.9131 ±0.03212	0.9136± 0.04148	0.9148 ±0.03906	0.9144 ±0.04489	0.9148 ±0.04082	**0.9155** ±0.03759
Fertility	0.871 ±0.000	0.668 ±1.394	0.8808 ±0.116	0.8885 ±0.08811	0.8835 ±0.1024	0.889 ±0.1148	0.8896 ±0.1153	**0.8918**± 0.1146
German	0.682± 0.144	0.465 ±0.902	0.7303 ±0.06051	0.732 ±0.05628	0.7314 ±0.0559	0.729 ±0.05193	0.7134 ±0.0602	**0.7321** ±0.05323
Ionosphere	0.362 ±0.415	0.507 ±1.233	0.7539 ±0.117	0.7577 ±0.1068	0.7486 ±0.1123	**0.7609** ±0.1079	0.7603 ±0.1231	0.7405 ±0.1396
Iris	0.962 ±0.000	0.700± 1.858	0.9634 ±0.05823	0.9634 ±0.05977	0.9615 ±0.05125	0.963± 0.05273	0.9601 ±0.0522	**0.9652** ±0.05206
Spect heart	**0.802**± 0.082	0.739 ±0.459	0.7993 ±0.08083	0.7933 ±0.08178	0.7958 ±0.07411	0.7935 ±0.09357	0.798±0.09485	0.7927 ±0.07383
WBCD	0.797± 0.020	0.714± 0.711	0.8061 ±0.05781	0.8029 ±0.05991	0.8055 ±0.05797	**0.807** ±0.06093	0.8049 ±0.07203	0.8032 ±0.07308

"Mean" and "Std" denote the average and and standard deviation of the corresponding classification accuracy obtained over 26 trials.

TABLE 3. Classification accuracies of eight algorithms on test sets.

Datasets	DE	PSO	FWA-CSGS1	FWA-CSGS2	FWA-CSGS3	FWA-CSGS4	standard FWA	SaFWA
	Mean±Std	Mean±Std	Mean±Std	Mean±Std	Mean±Std	Mean±Std	Mean±Std	Mean±Std
Biodegradation	0.629 ±0.259	0.480± 1.096	0.781 ±0.1005	0.793 ±0.1105	0.7898 ±0.1275	0.787 ±0.1115	0.7429 ±0.129	**0.7938** ±0.07156
Climate model	0.9122 ±0.042	0.804± 0.818	**0.9188** ±0.07494	0.9176 ±0.09679	0.9148 ±0.09115	0.9157 ±0.1047	0.9148 ±0.09524	0.9129 ±0.08831
Fertility	**0.900**± 0.000	0.664 ±1.550	0.8808 ±0.2835	0.8654 ±0.2332	0.8667 ±0.2582	0.8654 ±0.2886	0.8692 ±0.2786	0.8551 ±0.258
German	0.682 ±0.182	0.467± 0.920	0.7262 ±0.1062	0.7167 ±0.1193	0.7208 ±0.114	**0.7291** ±0.09587	0.6996 ±0.1099	0.7256 ±0.08892
Ionosphere	0.359 ±0.449	0.476 ±1.198	0.7161± 0.19	0.6952 ±0.1956	0.715 ±0.2245	0.7029 ±0.2364	0.7092 ±0.2267	**0.7278** ±0.2092
Iris	**0.978** ±0.000	0.704± 1.858	0.9547± 0.111	0.9547 ±0.149	0.959 ±0.1028	0.9598 ±0.1296	0.9607± 0.1269	0.9444 ±0.1054
Spect heart	0.760 ±0.163	0.720 ±0.380	0.7817 ±0.1889	0.7957 ±0.1912	0.7899 ±0.1732	0.7952 ±0.2187	0.7846 ±0.2217	**0.7971** ±0.1726
WBCD	0.794 ±0.048	0.782 ±0.705	0.8059± 0.1181	0.807± 0.1401	0.8041± 0.1238	0.8054 ±0.1655	0.8072± 0.1659	**0.8113**± 0.137

"Mean" and "Std" denote the average and and standard deviation of the corresponding classification accuracy obtained over 26 trials.

(x_i, y_i) was correctly predicted. So the classification accuracy for the whole data set can be calculated by counting the number of the examples which had the correct results.

VI. EXPERIMENTAL RESULTS AND DISCUSSIONS

The experimental results of all the algorithms on the training sets and test sets over 26 independent trials are listed in Tables 2 and 3 in terms of mean values (Mean) and standard deviations (Std). We compared SaFWA with PSO, DE, standard FWA, FWA-CSGS1, FWA-CSGS2, FWA-CSGS3 and FWA-CSGS4. To investigate the robustness of the eight algorithms on the training sets or test sets, their box plots on the 8 datasets are shown in Figures 1 and 2. Moreover, Figure 3 illustrates the convergence characteristics of SaFWA, standard FWA, PSO, and DE on both the 8 training datasets in terms of the fitness values as opposed to the corresponding objective functions. The best results in terms of mean values are typed in bold.

A. COMPUTATIONAL RESULTS AND ANALYSIS ON TRAINING SETS

By observing Table 2, we can see that the following results: SaFWA has better performance than DE on 7 datasets, and worse performance than DE on 1 dataset. SaFWA has better performance than PSO on all the datasets. SaFWA has better performance than standard FWA on 5 datasets, similar performance on 2 datasets, and worse performance than standard FWA on 1 dataset. SaFWA has better performance than FWA-CSGS1 on 5 datasets, similar performance on 1 dataset, and worse performance than FWA-CSGS1 on 2 datasets. SaFWA has better performance than FWA-CSGS2 on 6 datasets, similar performance on 1 dataset, and worse performance than FWA-CSGS2 on 1 dataset. SaFWA has better performance than FWA-CSGS3 on 5 datasets, similar performance on 2 datasets, and worse performance than FWA-CSGS3 on 1 dataset. SaFWA has better performance than FWA-CSGS4 on 5 datasets, similar performance on 1 dataset, and worse performance than FWA-CSGS4 on

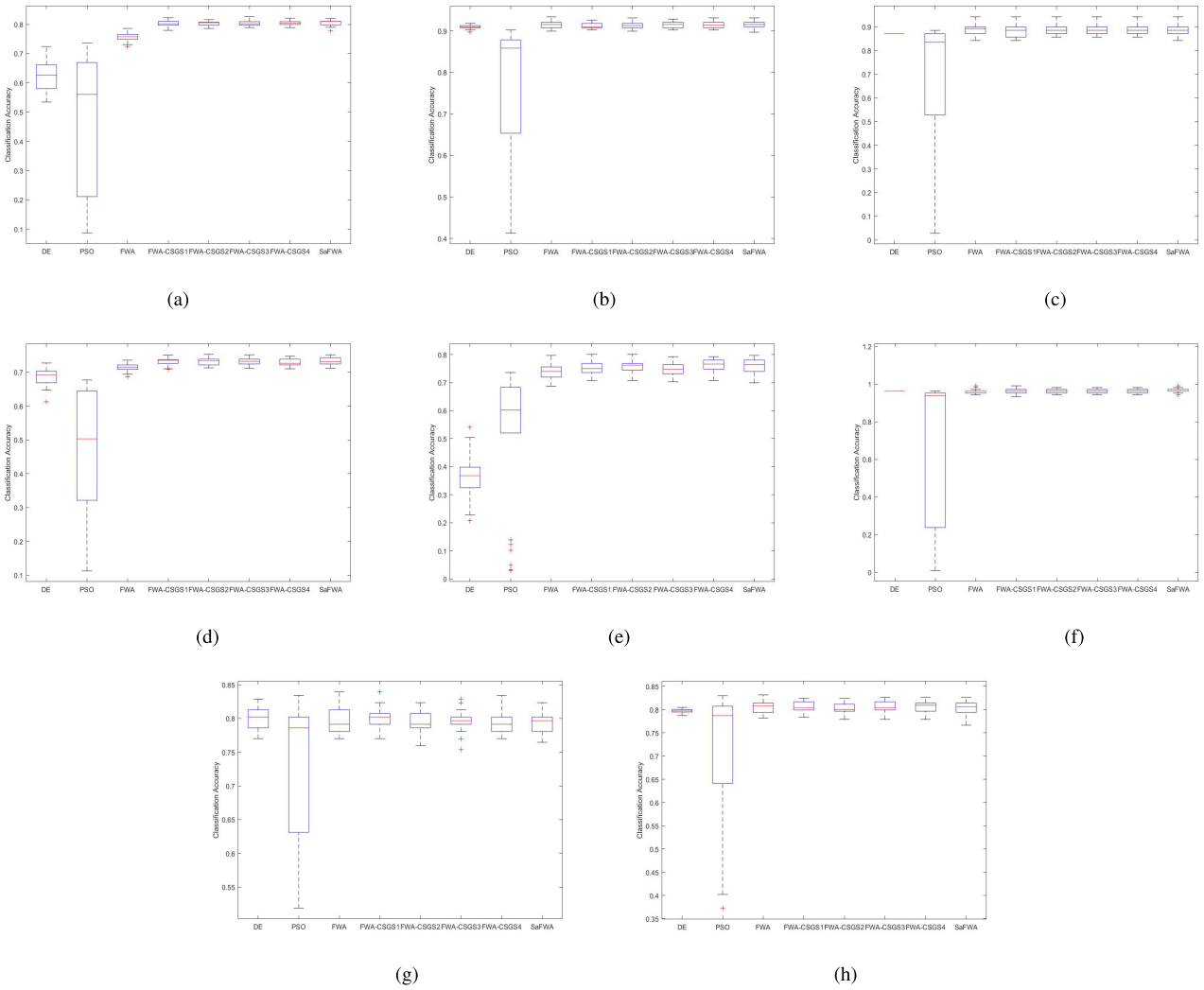


FIGURE 1. Box plots of accuracy on the training sets. (a) Biodegradation. (b) Climate model. (c) Fertility. (d) German. (e) Ionosphere. (f) Iris. (g) Spect heart. (h) WBCD.

2 datasets. This means that SaFWA performs better than other algorithms on most training sets. To examine stability of the eight algorithms, the box plots of the eight algorithms on the 8 training datasets are shown in Figure 1, from which it is observed that the robustness of SaFWA is better than that of the other algorithms on 4 datasets, similar to that of the other algorithms on 3 datasets, and worse than that of the other algorithms on 1 dataset. This means that SaFWA is more robust than the other algorithms. Thus, SaFWA obtains higher classification accuracy and it has much better robustness in most cases. Overall, SaFWA performs much better than the other algorithms in terms of classification accuracy and robustness on the training datasets.

B. COMPUTATIONAL RESULTS AND ANALYSIS ON TEST SETS

We further compare SaFWA with DE, PSO, standard FWA, FWA-CSGS1, FWA-CSGS2, FWA-CSGS3 and FWA-CSGS4 on the corresponding 8 test sets. The results are shown in Table 3, it can be observed that SaFWA has better performance than DE on 6 datasets and worse performance than DE on 2 datasets. SaFWA has better performance than PSO on all the datasets. SaFWA has better performance than standard FWA on 5 datasets, similar performance on 1 dataset, and worse performance than standard FWA on 2 datasets. SaFWA has better performance than FWA-CSGS1 on 4 datasets, similar performance on 1 dataset, and worse performance than FWA-CSGS1 on 3 datasets. SaFWA has better performance than FWA-CSGS2 on 5 datasets, similar performance on 1 dataset, and worse performance than FWA-CSGS2 on 2 datasets. SaFWA has better performance than FWA-CSGS3 on 5 datasets, similar performance on 1 dataset, and worse performance than FWA-CSGS3 on 2 datasets. SaFWA has better performance than FWA-CSGS4 on 4 datasets, similar performance on 2 datasets, and worse performance than FWA-CSGS4 on 2 datasets. In order to observe the robustness of the eight algorithms, the box

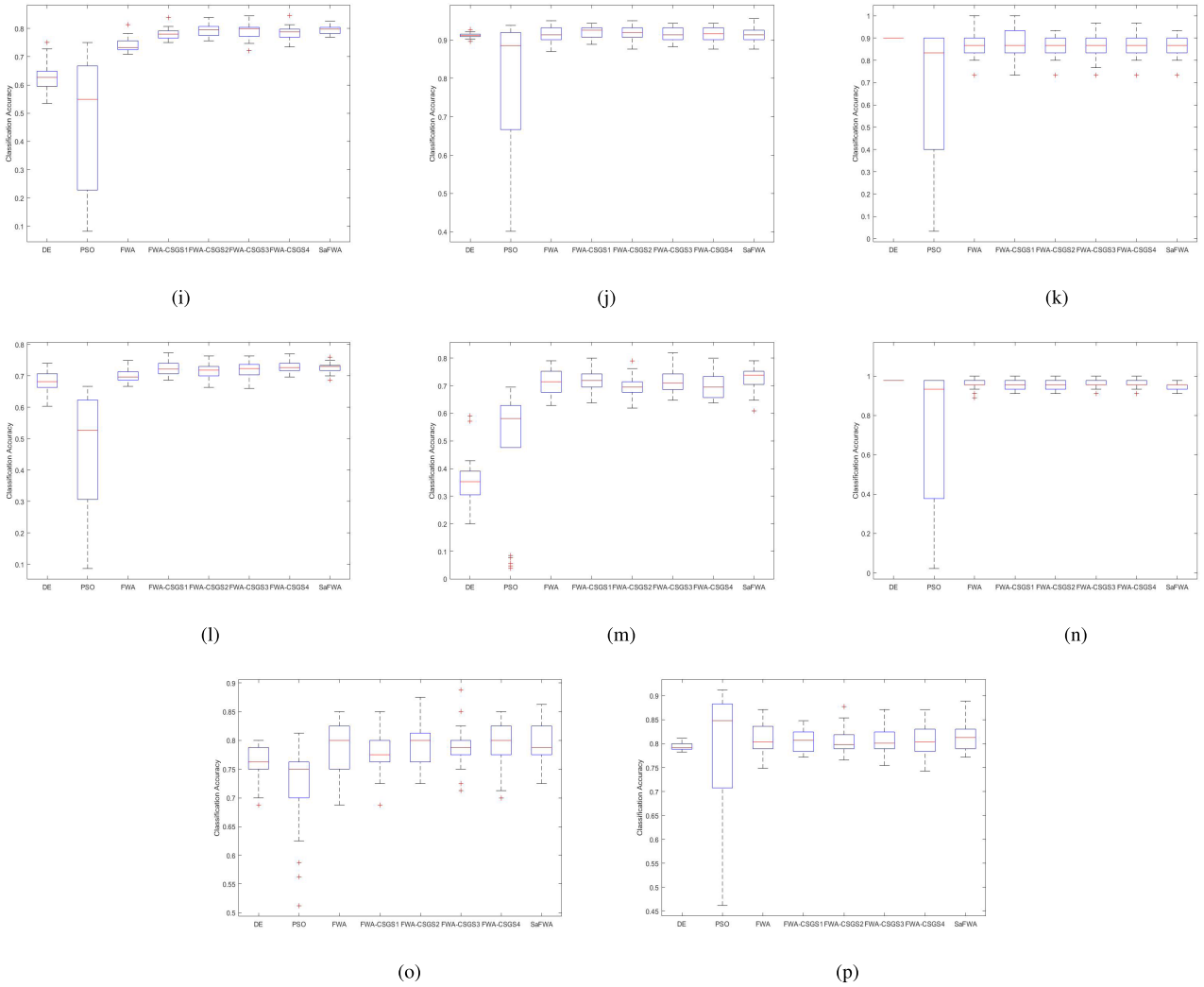


FIGURE 2. Box plots of accuracy on test sets. (a) Biodegradation. (b) Climate model. (c) Fertility. (d) German. (e) Ionosphere. (f) Iris. (g) Spect heart. (h) WBCD.

plots are shown in Figure 2, from which we can observe that the robustness of SaFWA is better than that of the other algorithms on 4 datasets, similar to that of the other algorithms on 3 datasets, and worse than that of the other algorithms on 1 dataset. Overall, the performance of SaFWA is better than that of the other algorithms on the test sets in terms of classification accuracy and stability. It is due to four effective SCGSs can increase the diversity of solutions and avoid trapping in local optima. Besides, the self-adaptive mechanism of SaFWA can dynamically choose the best CSGS for the corresponding problem during the search process of optimization classification problems, which can achieve better performance than other fixed algorithms. On the whole, it is possible and reliable to solve classification problems by EC techniques using the classification optimization model, and all the eight algorithms have high classification accuracy. Besides, the results obtained by SaFWA are better than those obtained by other algorithms.

C. CONVERGENCE PERFORMANCE OF STANDARD FWA AND SAFWA

Figure 3 illustrates the convergence characteristics of DE, PSO, standard FWA and SaFWA on the 8 datasets in terms of fitness value. In order to make the images clearer, we convert fitness to $e^{fitness}$ named as relevantFitness. So the horizontal axis and vertical axis represent the corresponding evolution generations and relevant fitness, respectively. By comparing the convergence curves of these four algorithms, it can be observed that at the beginning stage of evolution, SaFWA converges faster than other three algorithms on most datasets. Besides, at the later stages, although the convergence performance of the four algorithms decreases significantly, the value of objective function obtained by SaFWA is lower than those of the other three algorithms on most datasets. Thus, SaFWA has a better diversity property instead of stagnating into local optima. In addition to that, the convergence speed of SaFWA is faster, and the objective

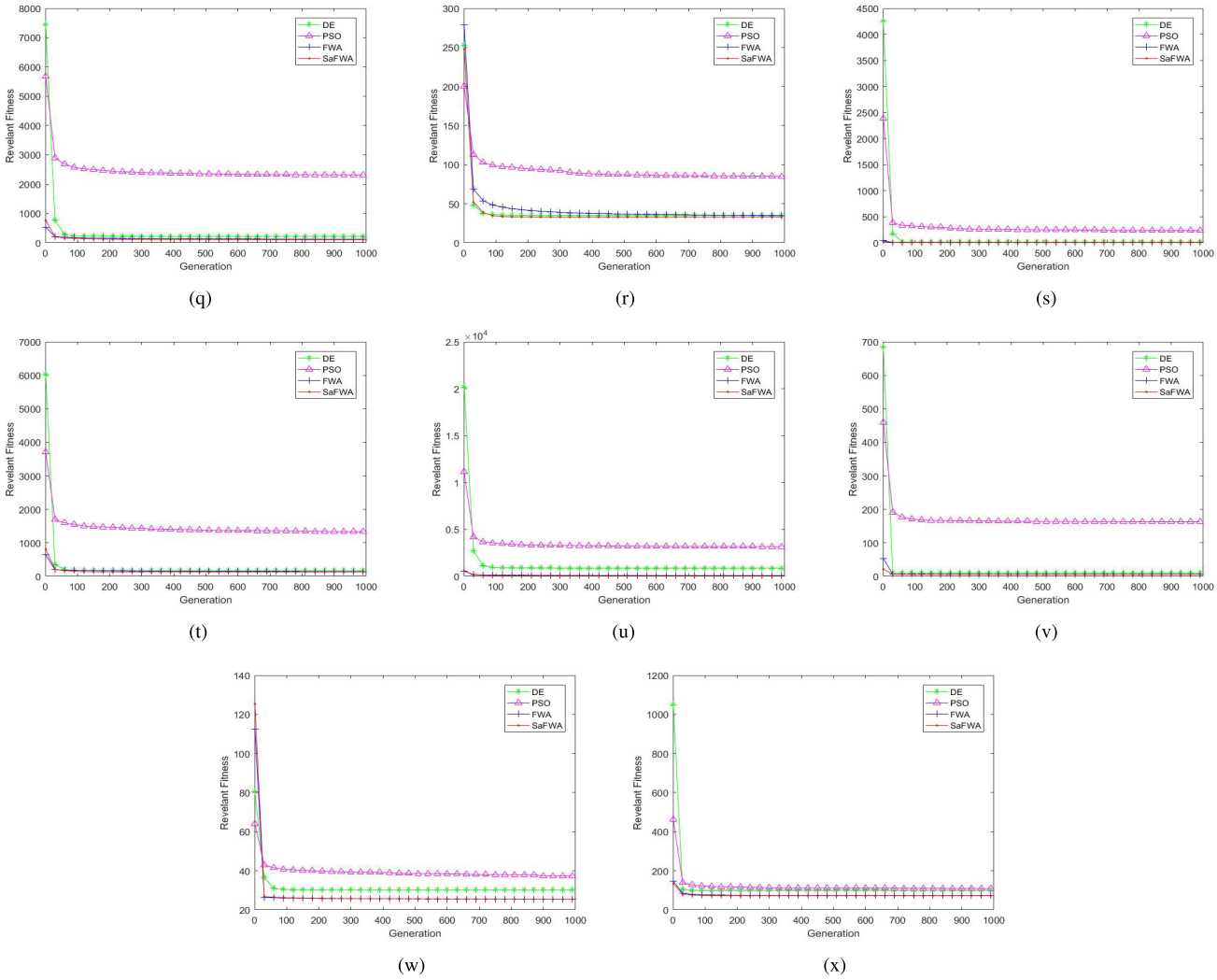


FIGURE 3. Curves of convergence on the eight datasets. (a) Biodegradation. (b) Climate model. (c) Fertility. (d) German. (e) Ionosphere. (f) Iris. (g) Spect heart. (h) WBCD.

fitness of SaFWA is usually lower than that of DE, PSO and standard FWA.

VII. CONCLUSIONS AND FUTURE WORK

There are many excellent methods proposed to solve classification problems and they all have good performance for classification. In this research, we propose an optimization classification model, and it can be used to solve classification problems by most EC techniques easily and straightforwardly. Besides, SaFWA, which employed a self-adaptive mechanism and four CSGSs, has been developed. Eight different datasets have been employed in the experiments. The results show that it is possible to solve classification problems by EC techniques with this new classification optimization model, and the performance of EC technique on classification problems indicates that it is a promising technique to straightforwardly solve classification problems by EC techniques. Moreover, the performance of SaFWA is better than that of DE, PSO, standard FWA and FWA variants with single CSGS.

Our next work is to further improve the optimization classification model and SaFWA. More data sets which are of high dimensions or more classes will be tested in the experiments. Finally, we will also take the structure of optimal classification model into account and consider using non-linear models for more complicated classification tasks.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for the valuable suggestions on improving this paper.

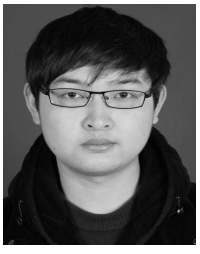
REFERENCES

[1] E. Viegas, A. O. Santin, A. França, R. Jasinski, V. A. Pedroni, and L. S. Oliveira, "Towards an energy-efficient anomaly-based intrusion detection engine for embedded systems," *IEEE Trans. Comput.*, vol. 66, no. 1, pp. 163–177, Jan. 2017.

[2] X. Li, S. Han, L. Zhao, C. Gong, and X. Liu, "New dandelion algorithm optimizes extreme learning machine for biomedical classification problems," *Comput. Intell. Neurosci.*, vol. 2017, Sep. 2017, Art. no. 4523754.

[3] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[4] Y. Xu, Z. Yang, and X. Pan, "A novel twin support-vector machine with pinball loss," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 359–370, Feb. 2017.
[5] J. E. Goin, "Classification bias of the *k*-nearest neighbor algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 3, pp. 379–381, May 1984.
[6] N. García-Pedrajas, J. A. R. del Castillo, and G. Cerruela-García, "A proposal for local *k* values for *k*-nearest neighbor rule," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 2, pp. 470–475, Feb. 2017.
[7] R. C. Barros, D. D. Ruiz, and M. P. Basgalupp, "Evolutionary model trees for handling continuous classes in machine learning," *Inf. Sci.*, vol. 181, no. 5, pp. 954–971, Mar. 2011.
[8] O. T. Yıldız, "VC-dimension of univariate decision trees," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 2, pp. 378–387, Feb. 2015.
[9] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, no. 8, pp. 2554–2558, 1982.
[10] L.-W. Kim, "DeepX: Deep learning accelerator for restricted boltzmann machine artificial neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1441–1453, Mar. 2018.
[11] I. Zelič, I. Kononenko, N. Lavrač, and V. Vuga, "Induction of decision trees and Bayesian classification applied to diagnosis of sport injuries," *J. Med. Syst.*, vol. 21, no. 6, pp. 429–444, Dec. 1997.
[12] S. Liu, G. Mingas, and C.-S. Bouganis, "An unbiased MCMC FPGA-based accelerator in the land of custom precision arithmetic," *IEEE Trans. Comput.*, vol. 66, no. 5, pp. 745–758, May 2017.
[13] Z.-Y. Chen, R. J. Kuo, and T.-L. Hu, "An integrated hybrid algorithm based on nature inspired evolutionary for radial basis function neural network learning," *Int. J. Artif. Intell. Tools*, vol. 25, no. 2, p. 25, Apr. 2016.
[14] T. H. Oong and N. A. M. Isa, "Adaptive evolutionary artificial neural networks for pattern classification," *IEEE Trans. Neural Netw.*, vol. 22, no. 11, pp. 1823–1836, Nov. 2011.
[15] S. N. Qasem, S. M. Shamsuddin, S. Z. M. Hashim, M. Darus, and E. Al-Shammari, "Memetic multiobjective particle swarm optimization-based radial basis function network for classification problems," *Inf. Sci.*, vol. 239, pp. 165–190, Aug. 2013.
[16] B. Xue, M. Zhang, and W. N. Browne, "Particle swarm optimization for feature selection in classification: A multi-objective approach," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1656–1671, Dec. 2013.
[17] Y. Xue, B. Zhao, T. Ma, and A. X. Liu, "An evolutionary classification method based on fireworks algorithm," *Int. J. Bio-Inspired Comput.*, vol. 11, no. 3, pp. 149–158, May 2018.
[18] C. Yu, L. Kelley, S. Zheng, and Y. Tan, "Fireworks algorithm with differential mutation for solving the cec 2014 competition problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Sep. 2014, pp. 3238–3245.
[19] Y.-J. Zheng, X.-L. Xu, H.-F. Ling, and S.-Y. Chen, "A hybrid fireworks optimization method with differential evolution operators," *Neurocomputing*, vol. 148, pp. 75–82, Jan. 2015.
[20] X. Li, S. Han, L. Zhao, and C. Gong, "Adaptive fireworks algorithm based on two-master sub-population and new selection strategy," in *Proc. Int. Conf. Neural Inf. Process.*, Nov. 2017, pp. 70–79.
[21] X.-G. Li, S.-F. Han, L. Zhao, C.-Q. Gong, and X.-J. Liu, "Adaptive mutation dynamic search fireworks algorithm," *Algorithms*, vol. 10, no. 2, p. 48, Apr. 2017.
[22] Q. Fan and X. Yan, "Differential evolution algorithm with self-adaptive strategy and control parameters for p-xylene oxidation process optimization," *Soft Comput.*, vol. 19, no. 5, pp. 1363–1391, May 2015.
[23] A. Banitalebi, M. I. A. Aziz, and Z. A. Aziz, "A self-adaptive binary differential evolution algorithm for large scale binary optimization problems," *Inf. Sci.*, vols. 367–368, pp. 487–511, Nov. 2016.
[24] Y. Xue, S. Zhong, Y. Zhuang, and B. Xu, "An ensemble algorithm with self-adaptive learning techniques for high-dimensional numerical optimization," *Appl. Math. Comput.*, vol. 231, pp. 329–346, Mar. 2014.
[25] Y. Xue, J. M. Jiang, B. P. Zhao, and T. H. Ma, "A self-adaptive artificial bee colony algorithm based on global best for global optimization," *Soft Comput.*, vol. 22, no. 9, pp. 2935–2952, May 2018.
[26] Y. Xue, J. Jiang, T. Ma, J. Liu, and W. Pang, "A self-adaptive artificial bee colony algorithm with symmetry initialization," *J. Internet Technol.*, 2016.
[27] Y. Xue, Y. Zhuang, T. Ni, S. Ni, and X. Wen, "Self-adaptive learning based discrete differential evolution algorithm for solving CJWTA problem," *J. Syst. Eng. Electron.*, vol. 25, no. 1, pp. 59–68, Feb. 2014.
[28] Y. Xue, S. M. Zhong, T. H. Ma, and J. Cao, "A hybrid evolutionary algorithm for numerical optimization problem," *Intell. Automat. Soft Comput.*, vol. 21, no. 4, pp. 473–490, Oct. 2014.
[29] Y. Xue, J. Jiang, B. Xue, and M. Zhang, "A classification method based on self-adaptive artificial bee colony," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Nov. 2017, pp. 1–8.
[30] Y. Xue, B. Zhao, and T. Ma, "Classification based on fireworks algorithm," in *Bio-inspired Computing—Theories and Applications*. 2016, pp. 35–40.
[31] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *Proc. Int. Conf. Swarm Intell.* Berlin, Germany: Springer, 2010, pp. 355–364.
[32] M. Guendouz, A. Amine, and R. M. Hamou, "A discrete modified fireworks algorithm for community detection in complex networks," *Appl. Intell.*, vol. 46, no. 2, pp. 373–385, Mar. 2017.
[33] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximity-based mutation operators," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 99–119, Feb. 2011.
[34] S. Wang, Y. Li, and H. Yang, "Self-adaptive differential evolution algorithm with improved mutation mode," *Appl. Intell.*, vol. 47, no. 3, pp. 644–658, Oct. 2017.
[35] X. Li, S. Ma, and J. Hu, "Multi-search differential evolution algorithm," *Appl. Intell.*, vol. 47, no. 1, pp. 231–256, Jul. 2017.
[36] J. Jiang, Y. Xue, T. Ma, and Z. Chen, "Improved artificial bee colony algorithm with differential evolution for the numerical optimisation problems," *Int. J. Comput. Sci. Eng.*, vol. 16, no. 1, pp. 73–84, Jan. 2018.
[37] Y. Xue, Y. Zhuang, X. Meng, and Y. Y. Zhang, "Self-adaptive learning based ensemble algorithm for solving matrix eigenvaluee," *J. Comput. Res. Develop.*, vol. 50, no. 7, pp. 1435–1443, 2013.
[38] A. Frank and A. Asuncion. (2010). *UCI Machine Learning Repository*. Accessed: Apr. 25, 2018. [Online]. Available: http://archive.ics.uci.edu/ml
[39] M. A. Mohiuddin, S. A. Khan, and A. P. Engelbrecht, "Fuzzy particle swarm optimization algorithms for the open shortest path first weight setting problem," *Appl. Intell.*, vol. 45, no. 3, pp. 598–621, Oct. 2016.
[40] N. Saxena and K. K. Mishra, "Improved multi-objective particle swarm optimization algorithm for optimizing watermark strength in color image watermarking," *Appl. Intell.*, vol. 47, no. 2, pp. 362–381, Sep. 2017.

YU XUE received the Ph.D. degree from the School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China, in 2013. During 2016–2017, he was a Visiting Scholar with the Victoria University of Wellington, Wellington, New Zealand. He is currently an Associate Professor with the School of Computer and Software, Nanjing University of Information Science and Technology. He is also a Visiting Scholar with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, USA. He has authored over twenty journal and conference papers. His research interests include evolutionary computation, feature selection, adaptive search, and evolutionary machine learning. He is a member of ACM and CCF.



BINPING ZHAO received the bachelor's degree from the School of Computer and Software, Nanjing University of Information Science and Technology, China, where he is currently pursuing the master's degree. His research interests include biomimetic algorithm, self-adaptive search, evolutionary multi-objective optimization, and machine learning.

TINGHUAI MA received the bachelor's and master's degrees from HUST, China, in 1997 and 2000, respectively, and the Ph.D. degree from the Chinese Academy of Science in 2003. In 2004, he joined AJOU University as a Post-Doctoral Associate. He is currently a Professor of computer science with the Nanjing University of Information Science and Technology, China. His research interests are data mining, cloud computing, ubiquitous computing, and privacy preserving.

WEI PANG received the Ph.D. degree in computing science from the University of Aberdeen in 2009. He is currently a Senior Lecturer with the University of Aberdeen. He has authored over 70 papers, including 30+ journal papers. His research interests include bio-inspired computing, data mining, machine learning, and qualitative reasoning.