Neurocomputing 333 (2019) 440–451

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A hybrid model of fuzzy min–max and brain storm optimization for feature selection and data classification

Farhad Pourpanah[a,*], Chee Peng Lim[b], Xizhao Wang[c], Choo Jun Tan[d], Manjeevan Seera[e], Yuhui Shi[f]

[a] College of Mathematics and Statistics, Shenzhen University, China
[b] Institute for Intelligent Systems Research and Innovation, Deakin University, Australia
[c] College of Computer Science and Software Engineering, Guang Dong Key Lab. of Intelligent Information Processing, Shenzhen University, China
[d] School of Science and Technology, Wawasan Open University, Malaysia
[e] Faculty of Engineering, Computing and Science, Swinburne University of Technology, Sarawak Campus, Malaysia
[f] Department of Computer Science and Engineering, Southern University of Science and Technology, China

ARTICLE INFO

Article history:
Received 15 October 2018
Revised 23 December 2018
Accepted 2 January 2019
Available online 12 January 2019

Communicated by Dr. Nianyin Zeng

Keywords:
Feature selection
Brain storm optimization
Fuzzy min–max
Data classification
Motor fault detection

ABSTRACT

Swarm intelligence (SI)-based optimization methods have been extensively used to tackle feature selection problems. A feature selection method extracts the most significant features and removes irrelevant ones from the data set, in order to reduce feature dimensionality and improve the classification accuracy. This paper combines the incremental learning Fuzzy Min–Max (FMM) neural network and Brain Storm Optimization (BSO) to undertake feature selection and classification problems. Firstly, FMM is used to create a number of hyperboxes incrementally. BSO, which is inspired by the human brainstorming process, is then employed to search for an optimal feature subset. Ten benchmark problems and a real-world case study are conducted to evaluate the effectiveness of the proposed FMM-BSO. In addition, the bootstrap method with the 95% confidence intervals is used to quantify the results statistically. The experimental results indicate that FMM-BSO is able to produce promising results as compared with those from the original FMM network and other state-of-the-art feature selection methods such as particle swarm optimization, genetic algorithm, and ant lion optimization.

1. Introduction

Feature selection is an important pre-processing step in data mining, especially for solving classification problems. The performance of classification algorithms is affected by redundant and noisy features, in addition to a long execution time to process all features [1]. Feature selection is a process of removing redundant and irrelevant features from a data set so that the classification algorithm can achieve better accuracy and/or reduce model complexity (by using fewer numbers of features). Nevertheless, it is a difficult task to select a relevant and useful feature subset, particularly with high-dimensional features due to a large search space [2].

A feature selection method employs a search technique to identify a feature subset and uses an algorithm to evaluate the selected feature subset. In general, feature selection methods can be categorized into three: filter, wrapper, and embedded methods [3]. Filter-based methods mainly use the characteristics of the training samples such as distance, similarity and dependency, to evaluate the selected feature subset [4]. Embedded-based methods integrate a search mechanism during the learning process, in order to increase the search speed [5]. Wrapper-based methods employ a classifier to operate as a feedback mechanism to evaluate the effectiveness of the various selected feature subsets. Wrapper-based methods are more effective, but they are more complex and require a longer execution time [6].

Over the years, many traditional wrapper-based feature selection methods, such as sequential forward selection (SFS) [7] and sequential backward selection (SBS) [8], have been used to produce promising results in tacking feature selection problems. However, they suffer from several limitations, including computational complexity [2] and nesting effects [9]. In addition, these methods sequentially add or remove features to improve the performance of the wrapped algorithm. When the features are added or removed, they are not updated in further steps. To overcome this problem, a floating strategy was used with SFS and SBS to devise sequential forward floating selection (SFFS) and sequential backward

* Corresponding author.
E-mail addresses: farhad@szu.edu.cn (F. Pourpanah), chee.lim@deakin.edu.au (C.P. Lim), xizhaowang@ieee.org (X. Wang), cjtan@wou.edu.my (C.J. Tan), msseera@swinburne.edu.my (M. Seera), shiyh@sustc.edu.cn (Y. Shi).

https://doi.org/10.1016/j.neucom.2019.01.011
0925-2312/

floating selection (SBFS) methods [10]. These methods evaluate all possible solutions, and then select the best feature subset. They are computationally expensive methods, especially when the feature dimension is high. To alleviate these problems, many population-based optimization algorithms, such as genetic algorithm (GA) [11], particle swarm optimization (PSO) [12], ant colony optimization (ACO) [13], and ant lion optimization (ALO) [14], have been utilized. These methods generate new solutions randomly and evaluate them based on their fitness values. New solutions are generated in subsequent iterations based on the individuals that yield better results in the current iteration. Therefore, these methods avoid generating solutions similar to inferior ones, leading to reduced computational time in obtaining the best feature subsets.

Among population based optimization algorithms, PSO-based feature selection methods have shown promising results due to their less complexity, simple structure, and fast convergence [15]. Brain storm optimization (BSO) [16] is a type of swarm intelligence (SI)-based optimization algorithm that imitates the human brainstorming process. Since its introduction, BSO has produced promising results in solving various optimization problems, e.g. approximating complex functions [17]. The focus of this research is to adopt BSO as a feature selection method to solve data classification problems.

On the other hand, the fuzzy min–max (FMM) neural network [18,19] is an incremental learning model that can be used to solve both classification and clustering problems. FMM combines the capability of fuzzy set theory with artificial neural network to form a unified framework. FMM is able to overcome the problem of catastrophic forgetting (which is also known as the stability-plasticity dilemma) [20], that means FMM is able to learn new samples without forgetting previously learned samples. The catastrophic forgetting phenomenon is the main challenge of many batch-based learning methods. During learning, FMM creates hyperboxs incrementally to store information in its network structure. Each hyperbox defines two points, i.e., minimum (min) and maximum (max), for each dimensional of an n-dimensional input space. The hyperbox size i.e., θ, is set between zero and one. Larger hyperboxs reduce the network complexity, but may compromise the performance. FMM uses the fuzzy set to determine the degree of membership function among its existing hyperboxs and the current input sample, in an attempt to identify which class/cluster the input sample belongs to. FMM has also been used with the GA to tackle rule extraction and classification problems [21,22].

In this paper, we present a hybrid model of FMM and BSO, i.e., FMM-BSO, to undertake feature selection and classification problems. Firstly, FMM is used as an incremental learning model to create a number of hyperboxes to encode knowledge from the data samples. Then, BSO is employed to remove redundant and irrelevant features and select an optimal feature subset. In order to identify the most significant features and remove irrelevant features, the concept of an "open" hyperbox in FMM is employed. FMM-BSO is evaluated using ten benchmark data problems and a real-world case study, i.e., motor fault detection. In addition, to quantify the results statistically, the bootstrap method [23] with its 95% confidence intervals is used. The main contributions of this research include:

- a hybrid FMM-BSO model to increase predictive accuracy and reduce the computational complexity by selecting a feature subset with few important features;
- a comprehensive evaluation of FMM-BSO for feature selection and data classification using benchmark and real-world problems, with the results analyzed and compared with those from other state-of the art methods.

The rest of the paper is organized as follows: Section 2 presents a review on population-based feature selection methods. Section 3 explains the structures of both BSO and FMM. Section 4 presents the details of the proposed FMM-BSO model. The experimental results and discussion are provided in Section 5. Finally, conclusions and suggestions for future study are presented in Section 6.

2. Related work

This section presents a review on population-based features selection methods. The GA is a useful first population-based method for feature selection [24]. Single and two-objective feature selection and rule extraction methods based on GA were proposed in [11]. FMM-GA [21], i.e., a hybrid model of GA and FMM, was proposed for feature selection and pattern classification. FMM-GA operated in two stages. Firstly, FMM was used to create a number of hyperboxs. Then, the GA selected the best feature subset from the created hyperboxes. Similar to [21], a two-stage hybrid model of Q-learning Fuzzy ARTMAP (QFAM) [25] and the GA was proposed for feature selection and rule extraction [26,27].

A feature selection method based on artificial bee colony (ABC) was proposed to tackle data classification problems [28]. The results showed that ABC was able to reduce classification error with fewer features, as compared with those from PSO, GA and ACO. A hybrid model of ABC and support vector machine (SVM) for solving medical classification problems with comparable results was reported in [29]. In [2], a multi-objective feature selection method using ABC optimization based on non-dominated sorting and genetically inspired search was proposed. Both binary and continuous versions of the proposed model were implemented, i.e., Bin-MOABC and Num-MOABC. Bin-MOABC outperformed other methods such as single-objective ABC and linear forward selection.

In [30], a hybrid model of ACO and neural network for solving medical classification problems was developed. A novel hybrid feature selection method combining ACO and GA (ACO-GA) to solve high dimensional classification problems was presented in [31]. ACO-GA showed a superior performance as compared with those of ACO and GA. In [1], a hybrid feature selection algorithm combining ACO and ABC, known as AC-AB, was designed to tackle classification problems. AC-AB was able to overcome the stagnation problem of ants and reduce the global search time of bees. AC-AB produced better results as compared with those from other feature selection methods such as PSO and ACO.

In [32], a binary ant lion optimization (ALO) based feature selection method was proposed for classification. The experimental results indicated the capability of the proposed technique in solving classification problems as compared with those from PSO, GA and binary bat algorithm (BBA). In [33], BBA was combined with the forest classifier to select an optimal feature subset for solving classification problems. In [34], the firefly algorithm (FA) was employed as a discriminative feature selection method to solve classification and regression problems. In [35], a binary grey wolf optimization (GWO) method was used to tackle feature selection and data classification. All these bio-inspired evolutionary computation (EC)-based feature selection methods have shown promising results as reported in the corresponding papers.

A binary PSO was demonstrated as an effective feature selection method to tackle classification problems in [36]. Indeed, PSO and its variants have been successfully employed to tackle feature selection and data classification problems, due to their simple structure and fast convergence. As an example, a hybrid model of modified multi-swarm PSO (MSPO) with SVM for solving feature selection was proposed in [37]. MSPO-SVM was able to produce superior results as compared with those from the original PSO, GA and grid search based methods. In [15], HPSO-LS, namely a hybrid model of PSO and local search (LS) strategy, was introduced for feature selection. The LS used the correlation information of

features which helped PSO to select district features. Two Bacterial Foraging Optimization (BFO)-based feature selection methods, denoted as adaptive chemotaxis BFO (ACBFO) and improved swarming and elimination dispersal BFO (ISEDBFO) were proposed to solve classification problems in [38].

HBBEPSO [39], i.e., a hybrid model consisting of binary bat and enhanced PSO, was proposed for feature selection and classification. HBBEPSO used the capability of the bat algorithm to help search the feature space, and the capability of PSO to converge the best global solution in the feature space. In [40], two feature selection methods using slap swarm algorithm (SSA) were presented to tackle classification problems. In the first method, eight transfer functions were used to convert continuous version of SSA to binary, while in the second method, the crossover was used in addition to transfer functions to improve the technique. The proposed method outperformed state-of-the-art feature selection methods such as GA, binary GWO, BBA and binary PSO. In [41], a hybrid model of improved PSO and shuffled frog leaping was developed for feature selection. Three classification algorithms i.e., naïve Bayes (NB), *K*-nearest neighbor and SVM, were used as the classification algorithms to evaluate the effectiveness of the selected feature subsets.

A new switching delayed PSO (SDPSO) was developed to undertake parameter identification problem of the lateral flow immunoassay (LFIA) [42]. In addition, a hybrid model of Extreme Learning Machine (ELM) and SDPSO was proposed to solve the short-term load forecasting problem [43]. The SDPSO was used to optimize the input weights and biases of ELM. Similar to [43], the SDPSO model was employed to optimize the SVM parameters [44].

All the above mentioned population-based feature selection methods are inspired from swarm and natural evolution. This paper employs the advantageous of BSO, which is a new SI method inspired by the human brainstorming process, as a feature selection method for solving classification problems.

3. The brain storm optimization and fuzzy min–max models

In this section, the BSO is firstly explained. Then, the structure of the supervised FMM is described in detail.

3.1. Brain storm optimization

BSO [16] has three main steps: clustering individuals, disrupting cluster centers, and creating new solutions. Firstly, BSO generates n random solutions, and evaluates them based on a fitness function. Then, BSO clusters n solutions into m groups using the k-mean clustering method. After that, a new solution is generated to replace a randomly selected cluster center. This step is accomplished through disrupting a cluster center. Finally, an individual is randomly selected based on one or a combination of two cluster center(s), as follows:

$$X_{selected} = \begin{cases} X_i, & one\ cluster \\ rand \times X_{1i} + (1 - rand) \times X_{2i}, & two\ clusters \end{cases} \quad (1)$$

where *rand* is a random value between 0 and 1, X_{1i} and X_{2i} are the i-th dimension of the selected clusters. The selected idea is updated as follows:

$$X_{new} = X_{selected} + \xi * random(0, 1) \quad (2)$$

where *random(0,1)* is a Gaussian random value with 0 and 1 as the mean and variance, respectively; and ξ is the adjusting factor, which is defined as follows:

$$\xi = logsin\left(\frac{0.5 * m_i - c_i}{k}\right) \times rand \quad (3)$$

where *logsin()* is the logarithmic sigmoid function, k is a changing rate for the slope of the *logsin()* function, *rand()* is a random value between 0 and 1, m_i and c_i are the maximum and current number of iterations, respectively. Fig. 1 shows the flowchart of the BSO algorithm.

3.2. Fuzzy min–max

As shown in Fig. 2, FMM consists of three layers, namely the input layer (F_A), hyperbox layer (F_b) and output layer (F_C). The number of nodes in F_A and F_C are the same as the dimension of the input samples and number of output classes, respectively. Each node in the hyperbox layer (F_B) indicates a hyperbox fuzzy set. Each hyperbox is indicated by a set of minimum and maximum points; therefore the feature space is in an n-dimensional unit cube (I^n). The F_A and F_B are connected through the minimum and maximum points of the hyperboxes. The hyperbox membership function is used as the transfer function, F_B. Each hyperbox fuzzy set can be defined as follows:

$$B_j = \left\{A_h, V_j, W_j, f\left(A_h, V_j, W_j\right)\right\} \forall X \epsilon I^n \quad (4)$$

where $A_h = (a_{h1}, a_{h2}, \ldots, a_{hn})$ is the input samples, $V_j = (v_{j1}, v_{j2}, \ldots, v_{jn})$ and $W_j = (w_{j1}, w_{j2}, \ldots, w_{jn})$ are the minimum and maximum points of B_j, respectively; and $f(A_h, V_j, W_j)$ is the membership function. Fig. 3 shows an example of a three dimensional hyperbox with its minimum (V_j) and maximum (W_j) points. The hyperboxes belong to the same class are allowed to overlap with each other, while FMM eliminates overlap hyperboxes belonging to those from different classes (shown in Fig. 4).

Each node in F_C represents a class. The F_B and F_C nodes are connected using binary values, which are stored in a matrix U, as follows:

$$u_{jk} = \begin{cases} 1 & if\ b_j\ is\ a\ hyperbox\ for\ class\ c_k \\ 0 & otherwise \end{cases} \quad (5)$$

where b_j and c_k are the jth and kth F^b and F^c node, respectively.

In order to find the closest hyperbox to the hth input sample (A_h), FMM uses a membership function, i.e., $B_j(A_h)$, where $0 \le B_j(A_h) \le 1$. The membership function can be written as follows:

$$B_j(A_h) = \frac{1}{2n}\sum_{i=1}^{n}\left[\begin{matrix} max\left(0, 1 - max\left(0, \gamma min\left(1, a_{hi} - w_{ji}\right)\right)\right) \\ + max\left(0, 1 - max\left(0, \gamma min\left(1, v_{ji} - a_{hi}\right)\right)\right) \end{matrix}\right] \quad (6)$$

where $A_h = (a_{h1}, a_{h2}, \ldots, a_{hn}) \in I^n$ represents the hth input sample, and γ is the sensitivity parameter that formulates how fast the membership function decreases when the distance between A_h and B_j increases.

The FMM training procedure consists of three steps, including hyperbox expansion, hyperbox overlap test, and hyperbox contraction test. The details of these steps are as follows:

Expansion: During the learning process, FMM performs the hyperbox expansion process to include the learning sample in the respective hyperbox. To expand hyperbox B_j for absorbing the learning sample, A_h, the following condition must be satisfied:

$$n\theta \ge \sum_{i=1}^{n}\left(max\left(w_{ji}, a_{hi}\right) - min\left(v_{ji}, a_{hi}\right)\right) \quad (7)$$

where $0 \le \theta \le 1$ indicates the maximum hyperbox size.

If the condition in Eq. (7) is satisfied, the maximum and minimum points of the hyperbox are updated as follows:

$$v_{ji}^{new} = min\left(v_{ji}^{old}, a_{hi}\right) \forall_i, i = 1, 2, \ldots, n \quad (8)$$

$$w_{ji}^{new} = max\left(w_{ji}^{old}, a_{hi}\right) \forall_i, i = 1, 2, \ldots, n \quad (9)$$

However, if the expansion criterion, i.e., Eq. (7), fails for all existing hyperboxes, a new hyperbox is added to encode the learning

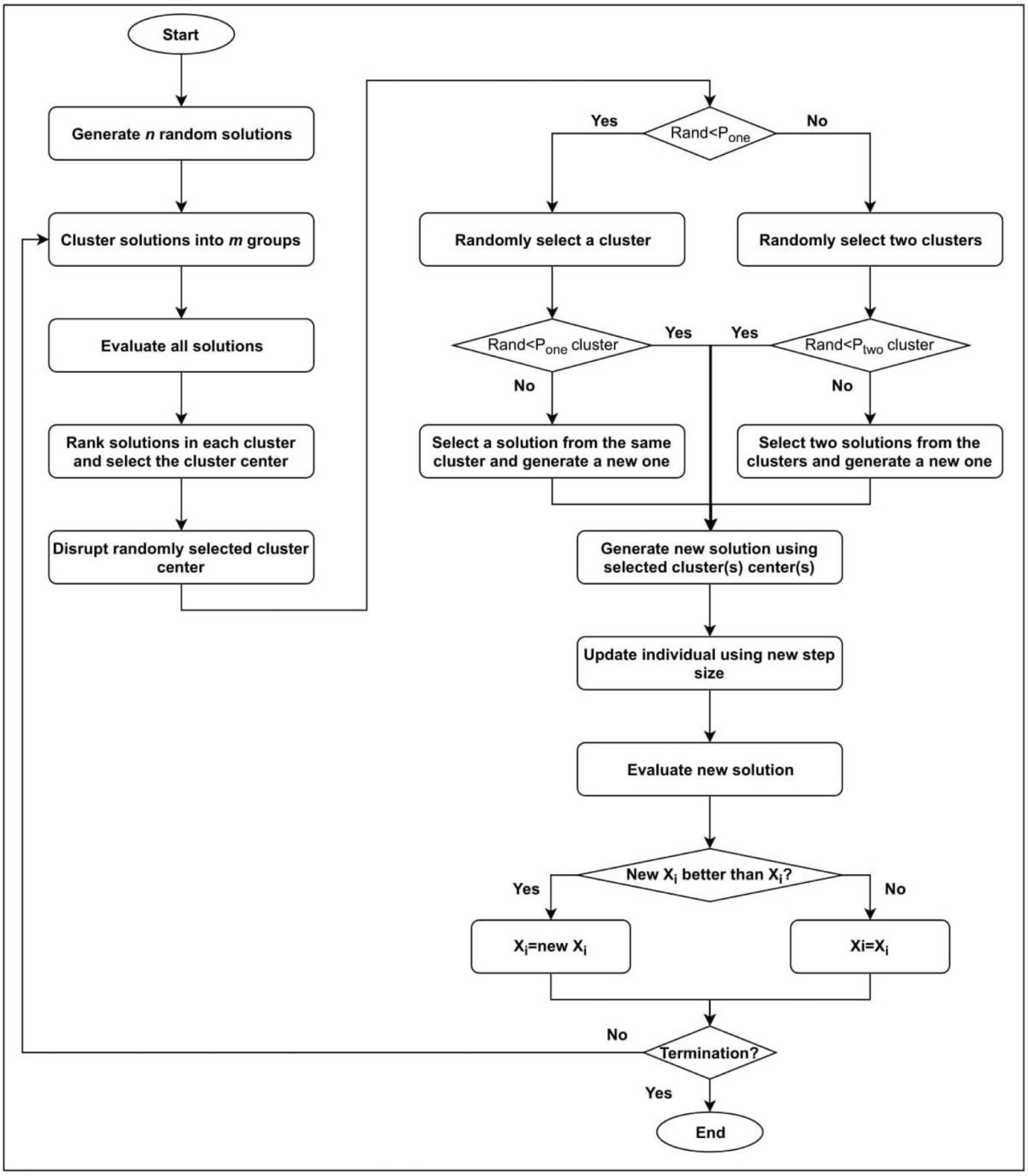

Fig. 1. Flowchart of BSO.

sample. This incremental learning process allows the network to add new hyperboxes without retraining.

Overlapping test: This test checks whether there is any overlap among hyperboxes that belong to different classes. For each dimension of the learning sample, if at least one of the following cases is met, there exists an overlap between two hyperboxes. The four test cases for the *i*th dimension are as follows:

Case 1:

$$v_{ji} < v_{ki} < w_{ji} < w_{ki}, \delta^{new} = min\left(w_{ji} - v_{ki}, \delta^{old}\right) \quad (10)$$

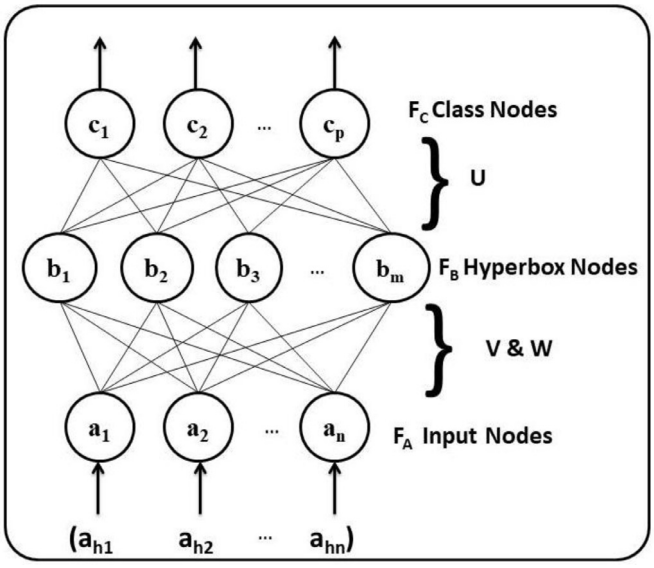

Fig. 2. The structure of FMM.

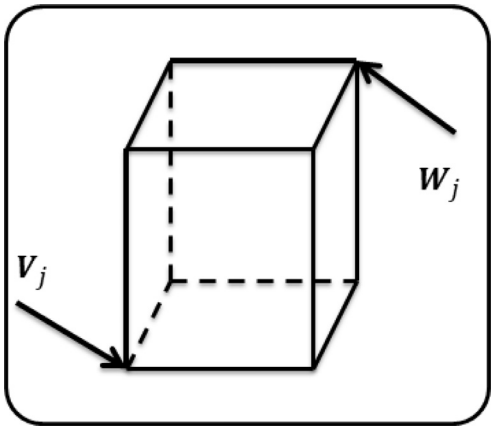

Fig. 3. An example of three dimensional hyperbox.

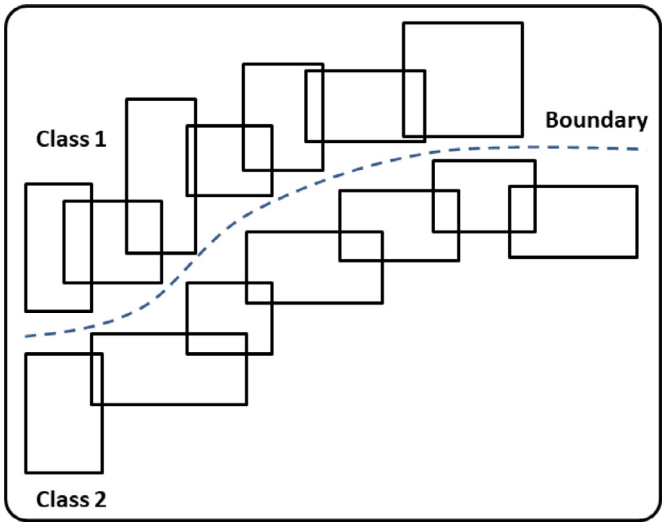

Fig. 4. An example of hyperboxes placed along the boundary of two classes.

Case 2:

$$v_{ki} < v_{ji} < w_{ki} < w_{ji}, \delta^{new} = min\left(w_{ki} - v_{ji}, \delta^{old}\right) \tag{11}$$

Case 3:

$$v_{ji} < v_{ki} < w_{ki} < w_{ji}, \delta^{new} = min\left(min\left(w_{ki} - v_{ji}, w_{ji} - v_{ki}\right), \delta^{old}\right) \tag{12}$$

Case 4:

$$v_{ki} < v_{ji} < w_{ji} < w_{ki}, \delta^{new} = min\left(min\left(w_{ji} - v_{ki}, w_{ki} - v_{ji}\right), \delta^{old}\right) \tag{13}$$

where j is the index of hyperbox B_j that is expanded in the previous step, and k is the index of the hyperbox B_k that belongs to another class currently being evaluated for possible overlap. If $\delta^{old} - \delta^{new} > 0$, $\delta^{old} = \delta^{new}$ and $\Delta = i$, the overlap test continues for the next dimension. When no overlap is found, Δ is set to a value, e.g., less than 0, to show that the contraction process is not required.

Contraction: If there exists an overlap between hyperboxes from different classes, the contraction process eliminates the overlap as follows: If $\Delta > 0$, the Δth dimension of the two overlap hyperboxes are required to be adjusted. In order to adjust the hyperboxes properly, four cases are examined, as follows:

Case 1:

$$v_{j\Delta} < v_{k\Delta} < w_{j\Delta} < w_{k\Delta}, w_{j\Delta}^{new} = v_{k\Delta}^{new} = \frac{w_{j\Delta}^{old} + v_{k\Delta}^{old}}{2} \tag{14}$$

Case 2:

$$v_{k\Delta} < v_{j\Delta} < w_{k\Delta} < w_{j\Delta}, w_{k\Delta}^{new} = v_{j\Delta}^{new} = \frac{w_{k\Delta}^{old} + v_{j\Delta}^{old}}{2} \tag{15}$$

Case 3a:

$$v_{j\Delta} < v_{k\Delta} < w_{k\Delta} < w_{j\Delta} \wedge \left(w_{k\Delta} - v_{j\Delta}\right) < \left(w_{j\Delta} < v_{k\Delta}\right), v_{j\Delta}^{new} = w_{k\Delta}^{old} \tag{16}$$

Case 3b:

$$v_{j\Delta} < v_{k\Delta} < w_{k\Delta}\left\langle w_{j\Delta} \wedge \left(w_{k\Delta} - v_{j\Delta}\right)\right\rangle\left(w_{j\Delta} < v_{k\Delta}\right), w_{j\Delta}^{new} = v_{k\Delta}^{old} \tag{17}$$

Case 4a:

$$v_{k\Delta} < v_{j\Delta} < w_{j\Delta} < w_{k\Delta} \wedge \left(w_{k\Delta} - v_{j\Delta}\right) < \left(w_{j\Delta} < v_{k\Delta}\right), w_{k\Delta}^{new} = w_{j\Delta}^{old} \tag{18}$$

Case 4b:

$$v_{k\Delta} < v_{j\Delta} < w_{j\Delta}\left\langle w_{k\Delta} \wedge \left(w_{k\Delta} - v_{j\Delta}\right)\right\rangle\left(w_{j\Delta} < v_{k\Delta}\right), v_{k\Delta}^{new} = w_{j\Delta}^{old} \tag{19}$$

4. The proposed FMM-BSO model

The proposed FMM-BSO model consists of two stages: (i) learning stage, (ii) feature selection stage. FMM is used in the first stage to learn the training samples. BSO is adopted in the second stage to select the best feature subset. The goal is to achieve a high classification rate and reduce the model complexity by selecting fewer numbers of features. The details are described in the following subsections.

4.1. Open hyperboxes

Once the FMM learning stage is completed (as explained in Section 3.2), all created hyperboxes are used to generate "*open*" hyperboxes [21], in order to enable FMM to include the "*don't care*"

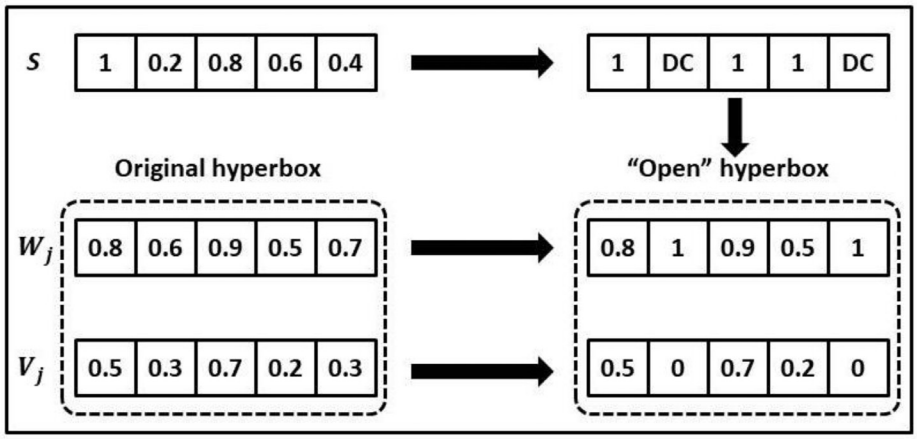

Fig. 5. An example of the generated solution, original hyperbox and "*Open*" hyperbox. where S is the generated solution, W_j and V_j are the maximum and minimum points of the original hyperbox, respectively. The "*Open*" hyperbox based on generated solution is shown in the right.

Algorithm 1 The procedure for measuring the fitness value of a single solution.

Input: Parameters of trained FMM and BSO, validation samples and a solution S
Output: Fitness value
1. Create the "*open*" hyperboxes by setting minimum and maximum point with 0 and 1, respectively.
2. Initialize "*don't care*" antecedents (i.e., Eqs. (20) and (21)).
3. For each validation sample **do**
3.1. Calculate the membership value (i.e., Eq. (6))
3.2. Select the hyperbox with the highest score as the winning hyperbox
3.3. Update performance indicator
4. End for
5. Compute the fitness value (classification error)

Algorithm 2 The proposed FMM-BSO model.

Input: Parameters of trained FMM and BSO, data samples (training, validation and test samples)
Output: Performance indicators
1. Train FMM using training samples.
2. Generate n random solutions.
3. Calculate the fitness function of all solutions.
4. **While** termination condition is not satisfied **do**
4.1. Cluster solutions into m groups using k-mean clustering.
4.2. Disrupt cluster center.
4.3. Update individual solutions.
4.4. Determine fitness value.
5. Use test samples to evaluate the performance of the selected feature subset.

antecedent [11,45]. A "*don't care*" dimension fully covers the specific "*don't care*" feature of the input space. To satisfy the "*don't care*" feature, the minimum and maximum points of the corresponding dimension can be set to 0 and 1, respectively. A total of $(2^d - 2)$ number of possible "*open*" hyperboxes (except the one hyperbox with all "*don't care*" antecedents) can be generated from a D-dimensional input sample. Fig. 5 shows an example of the generated solution, original hyperbox with its corresponding maximum and minimum points, and the "*Open*" hyperbox based on the solution. In this example, $\delta = 0.6$ (as explained in the next Subsection).

4.2. Adaptation of BSO for feature selection

In BSO, a solution, S, is formulated as follows:

$$S = \left\{D_1^1, D_2^1, \ldots, D_d^1, D_1^2, \ldots, D_d^2, \ldots, D_1^p, \ldots, D_D^P\right\} \tag{20}$$

where D is the dimension of each hyperbox, P is the number of hyperboxes created by FMM, and D_d^p is defined as follows:

$$D_d^p = \begin{cases} don'tcarefeature, & if\ D_d^p < \delta \\ otherfeatures, & if\ D_d^p > \delta \end{cases} \tag{21}$$

where $0 < \delta < 1$ is a user-defined threshold (see Fig. 5). The classification error is used as the fitness function to evaluate the performance of each feature subset. The step-by-step measurement of the fitness function pertaining to a single solution is given in Algorithm 1.

4.3. Summary of the proposed FMM-BSO model

Fig. 6 shows the flowchart of the proposed FMM-BSO model. Firstly, the parameters of FMM and BSO are initialized, and the data set is split into three subsets, i.e., learning, validation, and test samples. Before generating n random solutions (i.e., Eq. (20)), FMM is trained using the learning samples. After that, the *"open"* hyperboxs are created for each solution using Eq. (21), and the fitness value is measured using Algorithm 1. Next, k-mean clustering is used to group the solutions into m clusters. After disrupting the cluster center, the individual solution is updated using Eqs. (1)–(3). Then, the *"open"* hyperboxes for each updated solution is created, and its fitness value is measured. Replacement takes place if the new solution is performed better than the existing one. These steps are continued until the termination condition is satisfied. Finally, the test samples are used to evaluate the effectiveness of the selected feature subset. The procedure of FMM-BSO is shown in Algorithm 2.

4.4. Complexity analysis of FMM-BSO

The big-O notation [46] is used to analyze the computational complexity of FMM-BSO. The analysis can be split into two parts, namely, FMM and BSO, since both methods operate sequentially. In FMM, let D be the number of training samples, L be the dimension of the input sample in the F_A layer, M be the total number of hyperboxes in the F_B layer, and K $(K<M)$ be the number hyperboxes belonging to other classes (with respect to the current input sample). The procedure of FMM is shown in Algorithm 3. Given a new input sample A_h, FMM measures the membership values, i.e., $B_j(A_h)$, of those hyperboxes in F_B which are belonging to same class of the input sample. It selects the hyperbox with the highest membership value. If the selected hyperbox satisfies the condition in Eq. (7), it expands the hyperbox, and checks for any overlap between the selected hyperbox and those hyperboxes from the other classes. If there exists an overlap, the contraction operation occurs. In the case, if the condition in Eq. (7) is not satisfied, FMM adds a new hyperbox to encode the current input (A_h). In the worst-

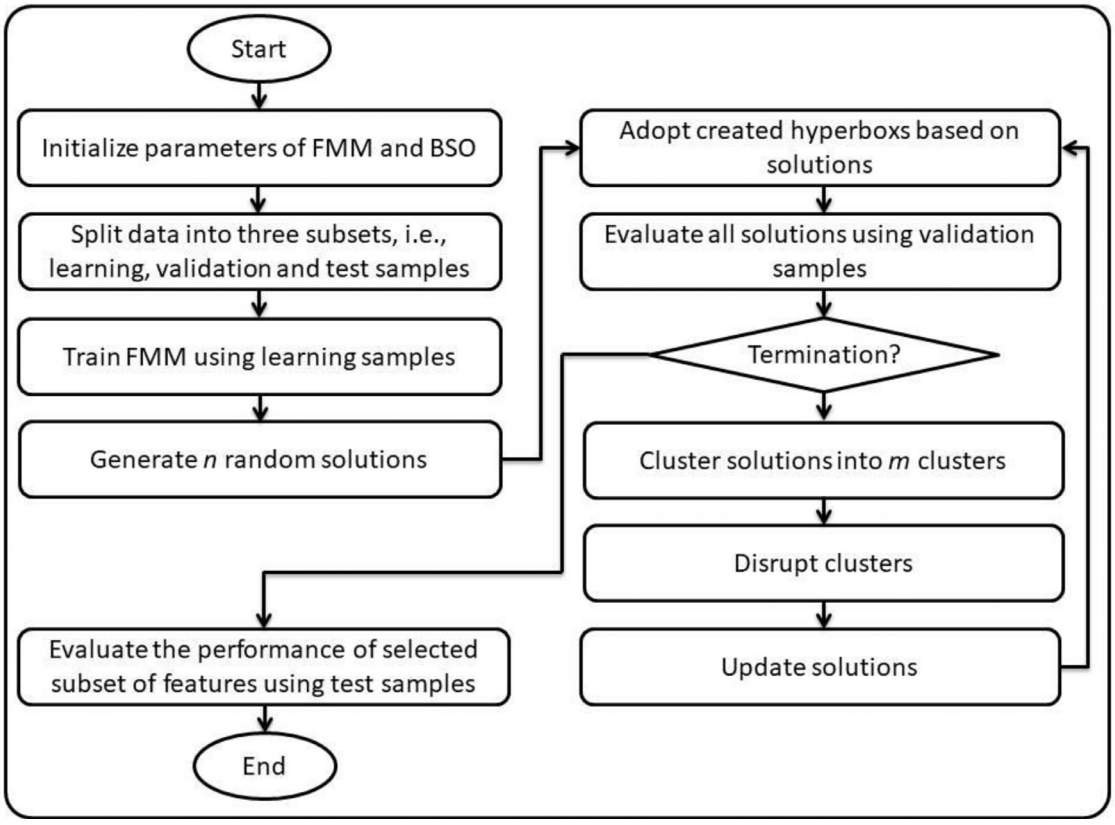

Fig. 6. The proposed FMM-BSO model.

Algorithm 3 The procedure of the FMM.

For training sample $d = 1: D$ **do**
For hyperbox $m = 1: M$ **do**
Compute membership value of those hyperboxes belonging to the class of current sample using Eq. (6).
Select the hyperbox with highest membership value as wining hyperbox.
If the wining hyperbox satisfied the condition in Eq. (7) **then**
Expand the hyperbox
For hyperbox of other classes $k = 1: K$ **do**
Check overlap between the winning hyperbox and those hyperboxes from the other classes
If there exist overlap **then**
For hyperbox dimension $l = 1: L$ **do**
Contract hyperboxes
else
Add new hyperbox to encode current input sample

case scenario when all variables extend to infinity, the computational complexity of FMM is of $O(M^*K^*L)$, as can be deduced from Algorithm 3. According to [47,48], FMM requires $(19^*L + 36)K$ steps for the key "overlap-contraction" operations for each input sample. With M hyperboxes, the computational complexity of $O(M^*K^*L)$ is in line with that in [47,48]. Given D training samples, and when $K \approx M$, the computational complexity of FMM becomes $O(M^2LD)$, when all variable extend to infinity.

In BSO, as shown in Algorithm 2 (lines 4.1–4.4), after generating n random solutions with P dimensions, there are a few further steps, namely (1) clustering solutions using the k-mean algorithm, (2) disrupting the cluster center, (3) generating new solutions, and (4) determining fitness value. According to Patel and Mehta [49], the computational complexity of the k-mean algorithm for n samples (solutions) is $O(n^2)$. The maximum step for disrupting the cluster center is $O(P)$. To update a solution, BSO adds a random value to each dimension of the selected solution. Therefore, the maximum step to update the solution is $O(P)$. The maximum step to determine a fitness value is measuring the membership value for all hyperboxes, which is $O(M)$. In the worst-case, the time complexity of BSO is $O(n^2)$ for the k-mean clustering algorithm, $O(P)$ for the disrupting the cluster center sub-procedure, $O(P)$ for the updating solutions sub-procedure, and $O(M)$ for determining the fitness value sub-procedure, which is of $O(n^2)$ when all variables extend to infinity. To sum up, the computational complexity of FMM-BSO, which runs sequentially, is of $O(M^2LD)$ for FMM and of $O(n^2)$ for BSO, when all variables extend to infinity.

Table 1
Details of UCI data set.

Data set	Number of input features	Number of data samples	Number of classes
Australian	14	690	2
Bupa liver	6	345	2
Cleveland Heart	13	303	2
Diabetes	8	768	2
German	24	1000	2
Ionosphere	34	351	2
Sonar	60	208	2
Vowel	13	990	10
Thyroid	5	215	3
Yeast	8	1848	10

5. Experimental studies

Ten benchmark data sets from the UCI machine learning repository [50] and a real-world case study, i.e., motor fault detection, were used to evaluate the effectiveness of FMM-BSO. Table 1 shows the details of the UCI data sets. These data sets were selected to compare the performance of FMM-BSO with those of other EC-based feature selection methods in the literature. Each

Table 2
Parameters of BSO (adopted from [16]).

N	m	P_{5a}	P_{6b}	$P6biii$	P_{6C}	K	Max-iteration	σ	μ
100	5	0.2	0.8	0.4	0.5	20	1000	1	0

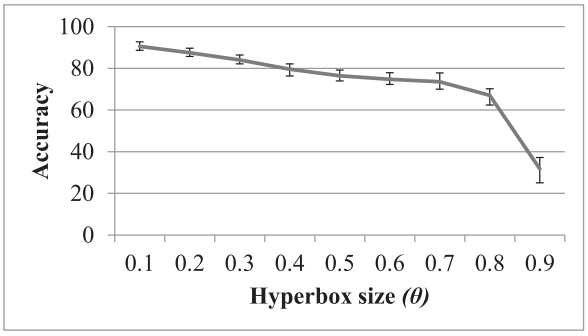

Fig. 7. Accuracy rates of FMM-BSO with different θ setting for German data set.

data set contains different characteristic in terms of difficulty and numbers of features and samples. Australian, Bupa liver and Diabetes data samples overlap each other, which is a challenging task for classification. Ionosphere has fewer numbers of overlapped samples. Sonar, Thyroid and Cleveland heart data samples have fewer numbers of samples. Sonar and German contain moderate imbalance data samples. Yeast and Vowel are multi-class data sets.

The experimental parameters were set as follows. The parameters of BSO are listed in Table 2. These parameters were adopted from [16]. According to other SI-based feature selection methods (e.g., PSO [51]), δ is usually set to 0.5 or slightly larger. Since the evolutionary-based algorithms automatically update the solutions, setting δ within [0.5, 0.7] would not significantly influence the feature selection process [51]. In this research, after several trials, δ was set to 0.6. All experiments were conducted using Matlab 2018a with 4 GHz CPU and 8GB memory.

5.1. UCI data sets

In this section, the performance of FMM-BSO was compared with those from the original FMM model and other state-of-the-art methods reported in the literature. For each data set, the 10-fold cross validation was used. To quantify the results statistically, each fold was repeated 10 times. Each experiment was repeated 5 times, giving a total of 500 runs for each data set. The bootstrap method [23] was employed to measure the 95% confidence intervals. A total of 90% and 10% of data samples were used for training (80% for learning and 10% for validation) and test, respectively. The validation samples were used to extract the optimal feature subset. Note that the test samples were not used to find the optimal feature subset. All data samples were normalized between 0 and 1.

The German data set was employed to find an optimal θ setting for producing the best performance. Fig. 7 shows the accuracy rates of FMM-BSO with different θ settings. As can be seen, the accuracy rates of FMM-BSO decrease when the hyperbox size (i.e., θ) is increased from 0.1 to 0.9. Note that setting θ=0.1 increases the model complexity, but reduces the classification error, as shown in Fig. 8. This setting was adopted throughout the experiments in this research.

Table 3 shows the accuracy rates with 95% confidence intervals of FMM and FMM-BSO. As indicated by the overlap of the 95% confidence intervals, FMM-BSO performed better or similar to the original FMM for nine out of ten data sets (except the Australian data set, which FMM outperformed FMM-BSO). Nonetheless, FMM-BSO managed to select fewer numbers of features, as compared

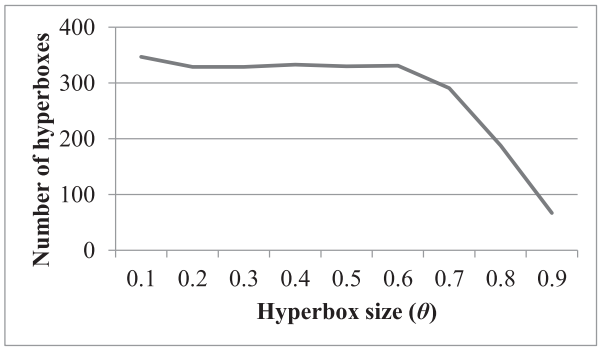

Fig. 8. Number of created hypeboxes by FMM with different θ setting for German data set.

Table 3
The accuracy rates (%) of FMM and FMM-BSO for UCI data sets ("Upper", "Mean" and "Lower" indicate the upper, mean and lower bounds of the 95% confidence intervals, respectively).

Data sets	FMM			FMM-BSO			
	Lower	Mean	Upper	Lower	Mean	Upper	Best
Australian	79.44	**80.44**	81.34	72.56	73.46	75.37	89.85
Bupa liver	68.30	70.60	72.80	69.30	**70.90**	71.70	77.10
Cleveland heart	76.70	78.96	80.71	83.40	**85.17**	87.38	92.32
Diabetes	80.38	81.31	82.35	80.74	**82.24**	85.65	90.78
German	86.37	88.73	91.20	88.78	**89.83**	90.53	98.67
Ionosphere	87.76	88.25	88.74	88.84	**89.59**	90.98	97.22
Sonar	90.41	**91.18**	92.21	88.90	89.93	91.19	100
Vowel	92.21	**92.72**	93.28	91.84	92.52	93.09	98.98
Thyroid	92.01	94.59	95.91	94.06	**94.72**	95.76	100
Yeast	67.64	69.13	71.17	67.43	**69.46**	72.34	79.05

Table 4
Numbers of selected features and computational time of FMM-BSO.

Data sets	All features	Selected	Time (s)
Australian	14	3.71	409
Bupa liver	6	3.56	194.70
Cleveland heart	13	6.17	59.03
Diabetes	8	5.07	198
German	24	14.35	543
Ionosphere	34	11.88	358
Sonar	60	17.94	149
Vowel	13	9.53	394
Thyroid	5	3.68	40.16
Yeast	8	4.63	564.70

with those from the original FMM model. Note that FMM used all features. The average numbers of selected features for each hyperbox by the original FMM model and FMM-BSO, and the execution time of FMM-BSO are presented in Table 4.

FMM-BSO was compared with particle swarm optimization (PSO) [52], Genetic algorithm (GA) [53], Simulated annealing (SA) [54], binary bat algorithm (BBA) [33], ant lion optimization (ALO) [14], Cuckoo search (CS) [55], adaptive chemotaxis bacterial foraging optimization algorithm (ACBFO) [38] and improved swarming and elimination dispersal bacterial foraging optimization (ISEDBFO) [38]. Note that all results related to PSO, GA, SA, BBA, ALO, CS, ACBFO and ISEDBFO were obtained from [38]. To have a fair compression the same experimental procedure in [38] was followed. Table 5 shows the accuracy rates of FMM-BSO, PSO, GA, SA, ALO, BBA, CS, ACBFO and ISEDBFO. Based on Table 5, FMM-BSO outperformed four out of ten benchmark data sets, i.e., Diabetes, German, Vowel and Yeast. For the Cleveland heart data set, FMM-BSO performed statistically similar to ISEDBFO, which ISEDBFO outperformed other methods, as indicated by the overlap between the 95% confidence intervals of FMM-BSO and accuracy of ISEDBFO.

Table 5
The accuracy rates (%) for UCI data sets ("Upper", "Mean" and "Lower" indicate the upper, mean and lower bounds of the 95% confidence intervals, respectively).

Data sets	PSO	GA	SA	ALO	BBA	CS	ACBFO	ISEDBFO	FMM-BSO		
									Lower	Mean	Upper
Australian	85.5	86.1	86.2	86.1	86.5	85.4	86.9	**87.3**	72.56	73.46	75.37
Bupa liver	71.1	70.1	72.3	71.2	69.2	68.9	74.2	**74.8**	69.30	70.90	71.70
Cleveland heart	84.1	82.4	84.2	82.5	82.5	84.1	85.8	**86.1**	83.40	85.17	87.38
Diabetes	76.2	77.1	77.3	76.5	77.5	77.3	77.9	77.6	80.74	**82.24**	85.65
German	74.4	75.7	76.7	75.5	76.2	76.7	76.8	77.4	88.78	**89.83**	90.53
Ionosphere	94.8	95.3	93.7	94.1	95.9	92.8	96.2	**96.6**	88.84	89.59	90.98
Sonar	85.2	87.1	85.8	88.8	90.2	89.7	**93.5**	92.8	88.90	89.93	91.19
Vowel	57.2	59.6	58.3	61.6	64.8	63.9	64.9	66.6	91.84	**92.52**	93.09
Thyroid	95.1	95.1	95.2	95.5	94.2	95.9	96.3	**97.2**	94.06	94.72	95.76
Yeast	56.5	57.3	57.4	61.4	60.3	62.7	63.3	65.3	67.43	**69.46**	72.34
Mean	78.0	78.6	78.7	79.3	79.7	79.7	81.6	82.2	82.6	**83.8**	85.4

Table 6
Average number of selected features.

Data sets	PSO	GA	SA	ALO	BBA	CS	ACBFO	ISEDBFO	FMM-BSO
Australian	9.8	9.0	9.7	9.3	10.1	9.5	8.6	8.2	**3.7**
Bupa liver	5.9	5.8	5.6	5.8	5.7	5.6	5.5	5.4	**3.6**
Cleveland heart	8.5	8.7	9.2	8.1	7.9	8.3	7.2	6.9	**6.2**
Diabetes	6.3	6.6	5.5	5.1	4.8	5.4	**4.2**	4.6	5.1
German	16.8	15.7	14.3	13.9	15.2	14.8	13.1	**12.3**	14.3
Ionosphere	19.2	19.5	18.9	17.3	18.2	17.8	16.8	16.1	**11.9**
Sonar	29.4	27.7	28.4	28.1	30.0	27.2	26.1	25.4	**18.0**
Vowel	9.2	8.8	8.0	7.4	8.1	7.8	6.9	**6.5**	9.5
Thyroid	4.1	4.3	3.4	4.0	3.6	3.7	**2.8**	3.0	3.7
Yeast	6.6	5.7	6.2	5.0	5.3	5.1	4.8	**4.6**	**4.6**
Mean	11.6	11.2	10.9	10.4	10.9	10.5	9.6	9.3	**8.1**

Table 7
Sensitivity rates for data sets from UCI machine learning repository.

Data sets	PSO	GA	SA	ALO	BBA	CS	ACBFO	ISEDBFO	FMM-BSO
Australian	0.79	0.84	0.87	0.83	0.84	0.87	0.87	0.88	0.77
Bupa liver	0.51	0.44	0.52	0.49	0.48	0.57	0.53	0.58	**0.62**
Cleveland heart	0.84	0.83	0.58	0.53	0.81	0.81	0.81	0.85	**0.87**
Diabetes	0.51	0.53	0.55	0.56	0.53	0.53	0.59	0.54	**0.84**
German	0.86	0.82	0.54	0.54	0.83	0.82	0.85	**0.87**	0.86
Ionosphere	0.92	0.92	0.91	0.88	0.93	0.93	0.95	0.96	**0.97**
Sonar	0.62	0.66	0.77	0.74	0.73	0.66	0.82	0.79	**0.87**
Mean	0.72	0.72	0.68	0.65	0.73	0.74	0.77	0.78	**0.83**

While FMM-BSO could not achieve the highest accuracy rates for the Australian, Bupa liver, Ionosphere, Sonar and Thyroid data sets, it selected fewer numbers of features for the Australian, Bupa liver, Cleveland heart, Ionosphere, Sonar and Yeast data sets. Table 6 shows the average numbers of selected features. Overall, FMM-BSO outperformed other methods in terms of mean accuracy and selected number of features, i.e., 83.8% and 8.1, respectively, for all data sets. The detailed results are presented in Tables 5 and 6.

In addition to accuracy and numbers of selected features, sensitivity and specificity were computed to compare the performance of FMM-BSO with those from other state-of-the-art feature selection methods. Sensitivity is the ratio of correctly classified positive samples to the total number of positive samples, and specificity is the ratio of correctly classified negative samples to the total number of negative samples [56]. Note that sensitivity and specificity are applicable to two-class classification problems. Tables 7 and 8 show the sensitivity and specificity rates of FMM-BSO, GA, SA, ALO, CS, ACBFO and ISEDBFO, respectively. According to Table 7, FMM-BSO was able to classify high rates of positive samples for the Bupa liver, Cleveland heart, Diabetes, Ionosphere and Sonar data sets. However, FMM-BSO could produce high rates for negative samples only for the German and Sonar data sets (as shown in Table 8). Overall, FMM-BSO was able to produce balanced sensitivity and specificity rates for the Australian, Cleveland heart, German, and Sonar data sets. FMM-BSO outperformed other methods in terms of mean sensitivity and ranked the second best method in terms of mean specificity for all data sets, as highlighted in Tables 7 and 8, respectively.

To have a fair comparison, FMM-GA and FMM-PSO were implemented. For all methods, the hyperbox size (θ), δ, population/particle size and maximum iteration were set to 0.1, 0.6, 100, and 1000, respectively. For PSO, both c_1 and c_2 were set to 1.49. For the GA, crossover and mutation probabilities were set to 0.9 and 0.1, respectively, and in each population, 20 prototypes were replaced. Fig. 9 shows the accuracy rates of FMM-GA, FMM-PSO, and FMM-BSO. For all data sets except Sonar, FMM-BSO performed similar to, if not better than, FMM-GA and FMM-PSO. While FMM-BSO could not produce good results as compared with those of other methods, it used significantly fewer numbers of features (18.0) in comparison with those of FMM-GA (29.89) and FMM-PSO (29.82), as shown in Table 9. Overall, FMM-BSO selected fewer numbers of features for five out of ten data sets, i.e., Australian, Cleveland heart, Ionosphere, Sonar and yeast.

5.2. Real-world case study

In this section, a real-world case study, i.e., motor fault detection, was used to evaluate the performance FMM-BSO. A total of 20 cycles, equivalent to 0.4 seconds of the unfiltered three-phase stator currents (i.e., phase A, phase B, and phase C), were transformed using fast Fourier transform to the respective Power Spectral Density for feature extraction. During feature

Table 8
Specificity rates for data sets from UCI machine learning repository.

Data sets	PSO	GA	SA	ALO	BBA	CS	ACBFO	ISEDBFO	FMM-BSO
Australian	0.83	0.85	0.86	0.86	0.88	0.58	0.83	**0.87**	0.70
Bupa liver	0.84	0.84	0.87	0.86	0.81	0.84	0.81	**0.88**	0.73
Cleveland heart	0.79	0.81	0.82	0.84	0.75	0.77	**0.85**	0.77	0.82
Diabetes	0.88	0.88	0.77	0.77	0.87	0.88	0.88	**0.89**	0.74
German	0.45	0.42	0.69	0.69	0.33	0.49	0.34	0.45	**0.83**
Ionosphere	0.62	0.87	0.42	0.42	0.89	0.70	0.85	0.87	0.76
Sonar	0.82	0.65	0.87	0.87	0.70	0.67	0.80	0.89	**0.91**
Mean	0.75	0.76	0.76	0.76	0.75	0.70	0.76	**0.80**	0.78

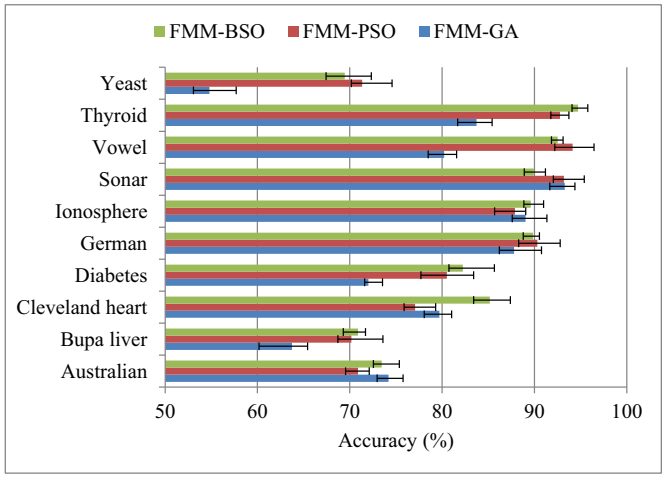

Fig. 9. Accuracy rates for UCI data sets.

Table 9
Average number of selected features.

Data sets	FMM-GA	FMM-PSO	FMM-BSO
Australian	6.98	7.01	**3.7**
Bupa liver	**3.03**	3.04	3.6
Cleveland heart	6.54	6.55	**6.2**
Diabetes	4.08	**4.01**	5.1
German	12.2	**12.01**	14.3
Ionosphere	16.95	16.99	**11.9**
Sonar	29.89	29.82	**18.0**
Vowel	6.61	**6.46**	9.5
Thyroid	2.53	**2.51**	3.7
Yeast	4.9	4.7	**4.6**

extraction, the selected pairs of harmonic magnitudes (i.e., the 3rd, 5th, 7th, and 13th harmonics) from the frequency spectrum were used as the input features of FMM-BSO. To further evaluate the robustness of the proposed FMM-BSO model with respect to tolerance of noise, white Gaussian noise was injected to the training samples [57]. While the data samples of real-world motor current contained noise, we added white Gaussian noise into 5% of the training samples, in order to ensure that the training set contained a certain minimum level of noise (5%) for evaluation in our experiment. Table 10 shows the data samples for each fault.

A design-of-experiment method, i.e., center composite design (CCD) [58], was used to analyze the impact of different settings of two parameters, i.e., probability (P_{one}) and number of clusters (m), on the FMM-BSO performance. As shown in Table 11, each parameter was set to three levels. Table 12 shows the results of five possible combinations of parameters. As can be seen, FMM-BSO with p_{one}=0 and $m=7$ (Exp. 5) outperformed other possible combinations of parameters for both noise-free and noisy data samples.

Table 10
Details of the real-world case study.

Type of faults	Number of samples
Normal	29
Broken Rotor Bars	58
Supply Unbalanced	29
Stator Winding Faults	28
Eccentricity	56

Table 11
The parameters and levels of center composite design.

Parameter	Level		
	Low	Medium	High
Number of clusters (m)	3	5	7
Probability P_{one}	0	0.5	1

Table 12
Effects of probability (P_{one}) and numbers of clusters (m) on the FMM-BSO performance (Accuracy).

Experiment	Noise-free			Noisy		
	Lower	Mean	Upper	Lower	Mean	Upper
Exp. 1 ($m=3$, $P_{one}=0$)	95.00	95.91	97.40	93.86	95.58	96.30
Exp. 2 ($m=5$, $P_{one}=0.5$)	93.50	94.51	95.60	90.80	92.30	94.60
Exp. 3 ($m=7$, $P_{one}=1$)	91.80	93.89	95.10	90.60	92.80	94.54
Exp. 4 ($m=3$, $P_{one}=1$)	93.50	94.20	94.70	91.45	93.24	94.53
Exp. 5 ($m=7$, $P_{one}=0$)	96.90	97.49	98.10	94.90	96.67	97.28

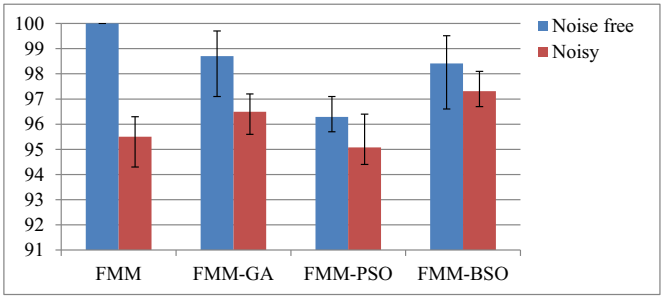

Fig. 10. The accuracy rates for motor fault detection.

The parameters of FMM-BSO, after several trials, were set to $m=5$ and P_{one}=0.8. As shown in Fig. 10, original FMM classified all test samples correctly for the noise-free data samples. For noisy samples FMM-BSO outperformed other methods in terms of mean accuracy. In addition, FMM-BSO managed to select fewer numbers of features in comparison with FMM-GA and FMM-PSO (as shown in Table 13). Both FMM-GA and FMM-BSO selected almost similar

Table 13
Number of selected features.

Method	Noise free	Noisy
FMM-GA	10.50	10.53
FMM-PSO	12.71	14.33
FMM-BSO	8.48	8.54

numbers of features for noise-free and noisy samples, i.e., approximately 10.50 and 8.50, respectively.

5.3. Discussion

GA and PSO models use the best solution and global best solution to generate new individual solutions. Comparatively, BSO uses all possible solutions to generate new ones. In addition, BSO randomly disrupts a cluster center to generate a new solution, which is different from the existing ones. This updating mechanism helps BSO to escape from the local optima and produce better results as compared with those of GA and PSO. In general, FMM-BSO is able to produce promising results, in terms of both accuracy and the number of selected features, as compared with those from the other feature selection methods. Nonetheless, BSO requires longer execution durations to find the optimal feature subset. This is mainly due to the use of distance-based *k*-mean clustering in each iteration. This problem can be solved by replacing *k*-mean clustering with objective space solutions. In addition, feature selection can be formulated as a binary problem. As such, instead of generating solutions between 0 and 1, binary values can be employed, which is more effective in finding an optimal solution.

6. Summary

In this paper, we have presented a hybrid model of FMM-BSO to solve feature selection problems for data classification. Firstly, FMM is used as a supervised learning algorithm to create hyperboxes incrementally. Then, BSO is adopted as the underlying technique to extract the best feature subset, in order to maximize classification accuracy and minimize the model complexity. Ten benchmark classification problems and a real-world case study, i.e., motor fault detection, have been used to evaluate the effectiveness of the proposed FMM-BSO model. The performance of FMM-BSO, in terms of classification accuracy and number of selected features, has been compared with those from the original FMM and other methods reported in the literature. Overall, FMM-BSO is able to produce promising results, which are similar to, if not better than, those from other state-of-the-art methods. However, FMM-BSO requires longer execution durations as compared with FMM-PSO and FMM-GA. It needs further investigation to improve its robustness. Our future work will focus on enhancing the performance of FMM-BSO using multi-objective fitness function and formulating a binary BSO variant to reduce the model complexity. To avoid the overfitting problem of FMM, a pruning strategy can be incorporated to help maintain a parsimonious network structure. In addition to classification, there are FMM-based variants for tackling regression problems. As such, FMM-BSO can be modified to solve regression problems, which is another direction of our further research.

Declarations of interest

None.

Acknowledgment

This work is partially supported by the National Natural Science Foundation of China (Grant Nos. 61772344, 61811530324 and 61732011), and the Natural Science Foundation of Shenzhen University (Grant nos. 827-000140, 827-000230, and 2017060).

References

[1] P. Shunmugapriya, S. Kanmani, A hybrid algorithm using ant and bee colony optimization for feature selection and classification (AC-ABC Hybrid), Swarm Evolut. Comput. 36 (2017) 27–36.
[2] E. Hancer, B. Xue, M. Zhang, D. Karaboga, B. Akay, Pareto front feature selection based on artificial bee colony optimization, Inf. Sci. (Ny). 422 (2018) 462–479.
[3] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn. Res. 3 (2003) 1157–1182.
[4] L.E.A. Santana, L. Silva, A.M.P. Canuto, F. Pintro, K.O. Vale, A comparative analysis of genetic algorithm and ant colony optimization to select attributes for an heterogeneous ensemble of classifiers, in: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE, 2010, pp. 1–8.
[5] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, J. R. Stat. Soc. Ser. B (Stat. Methodol.) 67 (2005) 301–320.
[6] Y. Zhang, D. Gong, Y. Hu, W. Zhang, Feature selection algorithm based on bare bones particle swarm optimization, Neurocomputing 148 (2015) 150–157.
[7] K.-J. Wang, K.-H. Chen, M.-A. Angelia, An improved artificial immune recognition system with the opposite sign test for feature selection, Knowl. Based Syst. 71 (2014) 126–145.
[8] T. Marill, D. Green, On the effectiveness of receptors in recognition systems, IEEE Trans. Inf. Theory. 9 (1963) 11–17.
[9] B. Ma, Y. Xia, A tribe competition-based genetic algorithm for feature selection in pattern classification, Appl. Soft Comput. 58 (2017) 328–338.
[10] P. Pudil, J. Novovičová, J. Kittler, Floating search methods in feature selection, Pattern Recognit. Lett. 15 (1994) 1119–1125.
[11] H. Ishibuchi, T. Murata, I.B. Türkşen, Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems, Fuzzy Sets Syst. 89 (1997) 135–150.
[12] B. Xue, M. Zhang, W.N. Browne, Particle swarm optimization for feature selection in classification: a multi-objective approach, IEEE Trans. Cybern. 43 (2013) 1656–1671.
[13] M.H. Aghdam, N. Ghasem-Aghaee, M.E. Basiri, Text feature selection using ant colony optimization, Expert Syst. Appl. 36 (2009) 6843–6853.
[14] S. Mirjalili, The ant lion optimizer, Adv. Eng. Softw. 83 (2015) 80–98.
[15] P. Moradi, M. Gholampour, A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy, Appl. Soft Comput. 43 (2016) 117–130.
[16] Y. Shi, Brain storm optimization algorithm, Advances in Swarm Intelligence, Springer, Berlin Heidelberg, 2011, pp. 303–309.
[17] Shi Cheng, Yifei Sun, Junfeng Chen, Quande Qin, Xianghua Chu, Xiujuan Lei, Yuhui Shi, A comprehensive survey of brain storm optimization algorithms, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), IEEE, 2017, pp. 1637–1644.
[18] P.K. Simpson, Fuzzy min–max neural networks. I. Classification, IEEE Trans. Neural Netw. 3 (1992) 776–786.
[19] P.K. Simpson, Fuzzy min–max neural networks - Part 2: clustering, IEEE Trans. Fuzzy Syst. 1 (1993) 32.
[20] G.A. Carpenter, S. Grossberg, A massively parallel architecture for a self-organizing neural pattern recognition machine, Comput. Vis. Graph. Image Process. 37 (1987) 54–115.
[21] A. Quteishat, C.P. Lim, S.T. Kay, A modified fuzzy min–max neural network with a genetic-algorithm-based rule extractor for pattern classification, IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans 40 (2010) 641–650.
[22] J. Liu, Z. Yu, D. Ma, An qdaptive fuzzy min–max neural network classifier based on principle component analysis and adaptive genetic algorithm, Math. Probl. Eng. 2012 (2012) 1–21.
[23] B. Efron, Bootstrap methods: another look at the jackknife, Ann. Stat. 7 (1979) 1–26.
[24] W. Siedlecki, J. Sklansky, A note on genetic algorithms for large-scale feature selection, Pattern Recognit. Lett. 10 (1989) 335–347.
[25] F. Pourpanah, C.P. Lim, Q. Hao, A reinforced fuzzy ARTMAP model for data classification, Int. J. Mach. Learn. Cybern. (2018) 1–13, doi:10.1007/s13042-018-0843-4.
[26] F. Pourpanah, C.P. Lim, J.M. Saleh, A hybrid model of fuzzy ARTMAP and genetic algorithm for data classification and rule extraction, Expert Syst. Appl. 49 (2016) 74–85.
[27] F. Pourpanah, C.J. Tan, C.P. Lim, J. Mohamad-Saleh, A Q-learning-based multi-agent system for data classification, Appl. Soft Comput. 52 (2017) 519–531.
[28] M. Schiezaro, H. Pedrini, Data feature selection based on artificial bee colony algorithm, EURASIP J. Image Video Process. 2013 (2013) 47.
[29] M.S. Uzer, N. Yilmaz, O. Inan, Feature selection method based on artificial bee colony algorithm and support vector machines for medical datasets classification, Sci. World J. 2013 (2013) 419187.
[30] R.K. Sivagaminathan, S. Ramakrishnan, A hybrid approach for feature subset selection using neural networks and ant colony optimization, Expert Syst. Appl. 33 (2007) 49–60.
[31] S. Nemati, M.E. Basiri, N. Ghasem-Aghaee, M.H. Aghdam, A novel ACO–GA hybrid algorithm for feature selection in protein function prediction, Expert Syst. Appl. 36 (2009) 12086–12094.

[32] E. Emary, H.M. Zawbaa, A.E. Hassanien, Binary ant lion approaches for feature selection, Neurocomputing 213 (2016) 54–65.
[33] R.Y.M. Nakamura, L.A.M. Pereira, K.A. Costa, D. Rodrigues, J.P. Papa, X.-S. Yang, BBA: a binary bat algorithm for feature selection, in: Proceedings of the Twenty Fifth SIBGRAPI Conference on Graphics Patterns Images, IEEE, 2012, pp. 291–297.
[34] L. Zhang, K. Mistry, C.P. Lim, S.C. Neoh, Feature selection using firefly optimization for classification and regression models, Decis. Support Syst. 106 (2018) 64–85.
[35] E. Emary, H.M. Zawbaa, A.E. Hassanien, Binary grey wolf optimization approaches for feature selection, Neurocomputing 172 (2016) 371–381.
[36] D.K. Agrafiotis, W. Cedeno, Feature selection for structure- activity correlation using binary particle swarms, J. Med. Chem. 45 (2002) 1098–1107.
[37] Y. Liu, G. Wang, H. Chen, H. Dong, X. Zhu, S. Wang, An improved particle swarm optimization for feature selection, J. Bionic Eng. 8 (2011) 191–200.
[38] Y.-P. Chen, Y. Li, G. Wang, Y.-F. Zheng, Q. Xu, J.-H. Fan, X.-T. Cui, A novel bacterial foraging optimization algorithm for feature selection, Expert Syst. Appl. 83 (2017) 1–17.
[39] M.A. Tawhid, K.B. Dsouza, Hybrid binary bat enhanced particle swarm optimization algorithm for solving feature selection problems, Appl. Comput. Inf. (2018) 1–10.
[40] H. Faris, M.M. Mafarja, A.A. Heidari, I. Aljarah, A.M. Al-Zoubi, S. Mirjalili, H. Fujita, An efficient binary salp swarm algorithm with crossover scheme for feature selection problems, Knowl. Based Syst. (2018) 43–67.
[41] S.P. Rajamohana, K. Umamaheswari, Hybrid approach of improved binary particle swarm optimization and shuffled frog leaping for feature selection, Comput. Electr. Eng. (2018) 497–508.
[42] N. Zeng, Z. Wang, H. Zhang, F.E. Alsaadi, A novel switching delayed PSO algorithm for estimating unknown parameters of lateral flow immunoassay, Cognit. Comput. 8 (2016) 143–152.
[43] N. Zeng, H. Zhang, W. Liu, J. Liang, F.E. Alsaadi, A switching delayed PSO optimized extreme learning machine for short-term load forecasting, Neurocomputing 240 (2017) 175–182.
[44] N. Zeng, H. Qiu, Z. Wang, W. Liu, H. Zhang, Y. Li, A new switching-delayed-PSO-based optimized SVM algorithm for diagnosis of Alzheimer's disease, Neurocomputing 320 (2018) 195–202.
[45] H. Ishibuchi, T. Nakashima, T. Murata, Three-objective genetics-based machine learning for linguistic rule extraction, Inf. Sci. (Ny) 136 (2001) 109–133.
[46] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, MIT press, 2009.
[47] A.V. Nandedkar, P.K. Biswas, A fuzzy min–max neural network classifier with compensatory neuron architecture, IEEE Trans. Neural Netw. 18 (2007) 42–54.
[48] Huaguang Zhang, Jinhai Liu, Dazhong Ma, Zhanshan Wang, Data-core-based fuzzy min–max neural network for pattern classification, IEEE Trans. Neural Netw. 22 (2011) 2339–2352.
[49] V.R. Patel, R.G. Mehta, Impact of outlier removal and normalization approach in modified k-means clustering algorithm, Int. J. Comput. Sci. Issues 8 (2011) 331.
[50] K. Bache, M. Lichman, {UCI} Machine learning repository. http://archive.ics.uci.edu/ml, 2013.
[51] B. Tran, B. Xue, M. Zhang, Variable-length particle swarm optimisation for feature selection on high-dimensional classification, IEEE Trans. Evolut. Comput. (2018) 1–1, doi:10.1109/TEVC.2018.2869405.
[52] A. Unler, A. Murat, R.B. Chinnam, mr2PSO: a maximum relevance minimum redundancy feature selection method based on swarm intelligence for support vector machine classification, Inf. Sci. (Ny). 181 (2011) 4625–4641.
[53] C.-L. Huang, C.-J. Wang, A GA-based feature selection and parameters optimizationfor support vector machines, Expert Syst. Appl. 31 (2006) 231–240.
[54] J.C.W. Debuse, V.J. Rayward-Smith, Feature subset selection within a simulated annealing data mining algorithm, J. Intell. Inf. Syst. 9 (1997) 57–81.
[55] X.-S. Yang, S. Deb, Cuckoo search: recent advances and applications, Neural Comput. Appl. 24 (2014) 169–174.
[56] J.L. Melsa, D.L. Cohn, Decision and estimation theory, McGraw-Hill series in electrical engineering: Communications and information theory, McGraw-Hill (1978).
[57] M. Seera, C.P. Lim, D. Ishak, H. Singh, Fault detection and diagnosis of induction motors using motor current signature analysis and a hybrid FMM-CART model., IEEE Trans. Neural Netw. Learn. Syst. 23 (2012) 97–108.
[58] Q.-K. Pan, M. Fatih Tasgetiren, Y.-C. Liang, A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem, Comput. Oper. Res. 35 (2008) 2807–2839.

Farhad Pourpanah received his Ph.D. degree in computational intelligence from the University Science Malaysia in 2015. He is currently an associate researcher at the College of Mathematics and Statistics, Shenzhen University (SZU), China. Before joining to SZU, he was a postdoctoral research fellow with the Department of Computer Science and Engineering, Southern University of Science and Technology (SUSTech), China. His research interests include evolutionary algorithms, pattern recognition and deep learning.

Chee Peng Lim received his Ph.D. degree in intelligent systems from the University of Sheffield, UK, in 1997. His research interests include computational intelligence-based systems for data analytics, condition monitoring, optimization, and decision support. He has published more than 350 technical papers in journals, conference proceedings, and books, received 7 best paper awards, edited 3 books and 12 special issues in journals, and served in the editorial board of 5 international journals. He is currently a professor at Institute for Intelligent Systems Research and Innovation, Deakin University.

Xizhao Wang (M'03-SM'04-F'12) served in Hebei University as a professor and the dean of school of Mathematics and Computer Sciences before 2014. After 2014 Prof. Wang worked as a professor in Big Data Institute of ShenZhen University. Prof. Wang's major research interests include uncertainty modeling and machine learning for big data. Prof. Wang has edited 10+ special issues and published 3 monographs, 2 textbooks, and 200+ peer-reviewed research papers. As a Principle Investigator (PI) or co-PI, Prof. Wang has completed 30+ research projects. Prof. Wang has supervised more than 100 Mphil and PhD students.

Prof. Wang is an IEEE Fellow, the previous BoG member of IEEE SMC society, the chair of IEEE SMC Technical Committee on Computational Intelligence, the Chief Editor of Machine Learning and Cybernetics Journal, and associate editors for a couple of journals in the related areas. He was the recipient of the IEEE SMCS Outstanding Contribution Award in 2004 and the recipient of IEEE SMCS Best Associate Editor Award in 2006. He is the general Co-Chair of the 2002-2017 International Conferences on Machine Learning and Cybernetics, cosponsored by IEEE SMCS. Prof. Wang was a distinguished lecturer of the IEEE SMCS.

Choo Jun Tan received his Ph.D. degree from School of Computer Sciences, University Science of Malaysia in 2014. He has more than 10-year experience in software design and development, as well as research and development in computational intelligence. He has designed and developed a number of intelligent software systems that have received national and international awards. His research interests include computational intelligence, especially evolutionary algorithms and their application to multi-objective optimization problems.

Manjeevan Seera received the B.Eng. (Hons) degree in Electronics and Electrical Engineering from the University of Sunderland, U.K., in 2007 and Ph.D. degree in Computational Intelligence from the University Science Malaysia in 2012. He is currently an Adjunct Research Fellow with Swinburne University of Technology (Sarawak Campus), Malaysia. His research interests include soft computing, pattern recognition, fault detection and diagnosis, and human-robot interaction.

Dr. Yuhui Shi is a Chair Professor in the Department of Computer Science and Engineering, Southern University of Science and Technology (SUSTech), Shenzhen, China. Before joining SUSTech, he was with the Department of Electrical and Electronic Engineering at the Xi'an Jiaotong-Liverpool University (XJTLU), Suzhou, China, from January 2008 to August 2017, was with the Electronic Data Systems Corporation (EDS), Indiana, USA, from October 1998 to December 2007, and was with Purdue School of Engineering and Technology, Indiana, USA, from October 1995 to September 1998. He is an IEEE Fellow, the Editor-in-Chief of the International Journal of Swarm Intelligence Research, and an Associate Editor of the IEEE Transactions on Evolutionary Computation. He co-authored a book on Swarm Intelligence together with Dr. James Kennedy and Dr. Russell C. Eberhart, and another book on Computational Intelligence: Concept to Implementation together with Dr. Russell C. Eberhart.