Pattern Recognition Letters 77 (2016) 21–27

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

A binary clonal flower pollination algorithm for feature selection☆

Safinaz AbdEl-Fattah Sayed*, Emad Nabil, Amr Badr

Department of Computer Science, Cairo University, Cairo, Egypt

ARTICLE INFO

Article history:
Received 19 September 2015
Available online 25 March 2016

Keywords:
Feature selection
Clonal Selection Algorithm
Flower Pollination Algorithm
Optimum Path Forest

ABSTRACT

Feature selection problem has been detected essentially in the last years. It is a step that is considered the prerequisite of the classification step. For the feature selection problem, the goal is to find out the most important subset of features that represent the original features in a certain domain. The selected features are used in optimization of a certain fitness function, so the feature selection problem can be seen as an optimization problem. This paper presents a new hybrid algorithm that combines Clonal Selection Algorithm (CSA) with Flower Pollination Algorithm (FPA) to compose Binary Clonal Flower Pollination Algorithm (BCFA) to solve the feature selection problem. The accuracy of the Optimum-Path Forest (OPF) classifier is used as an objective function. The experiments were implemented on three public datasets and demonstrated that the proposed hybrid algorithm achieved remarkable results in comparison with other well-known algorithms such as Binary Cuckoo Search Algorithm (BCSA), Binary Bat Algorithm (BBA), Binary Differential Evolution Algorithm (BDEA) and Binary Flower Pollination Algorithm (BFPA).

1. Introduction

Nowadays, many applications, that are used in medicine and industry, depend on large datasets with a lot of features, some of these features may be considered irrelevant, high dimensional or noisy that will degrade the performance of the machine learning tasks. Feature Selection is an important technique to find out the most important subset of features to sort out this issue. This task can be extremely useful in reducing the dimensional data to be processed by the classifier, reducing the execution time and enhancing the recognition rate of the classifier. There are several studies that have addressed the feature selection problem as an optimization problem where the fitness function is the accuracy of the given classifier that may be maximized by the selected features.

Nature inspired metaheuristic algorithms are now among the most widely used algorithms for solving the optimization problems. Optimization is a process of searching for the optimal solutions to a particular problem. There are many natural inspired metaheuristic algorithms that are worked on the feature selection problem, some of these algorithms are combined to compose hybrid algorithms and others are used alone to solve this problem. Several previous studies used hybrid algorithms for the feature selection problem, for instance [1] who hybridized Ant Colony Optimization (ACO) with Cuckoo Search (CS) to solve the feature selection problem in the digital mammogram, while [2] combined Particle Swarm Optimization (PSO) with Genetic Algorithm (GA) to select the most important features subset for gender classification. Rough Set Theory (RST), as a feature selector tool, was combined with the Fire Fly Algorithm (FFA) [3], Bee Colony Optimization [4] and ACO [5] to improve the performance of feature selection procedure.

On the contrary, there are numerous algorithms that applied separately to solve the feature selection problem, such as Binary Bat Algorithm (BBA) in [6], Binary Cuckoo Search Algorithm (BCSA) in [7], Binary Flower Pollination Algorithm (BFPA) in [8], where they evaluated the accuracy of Optimum-Path Forest (OPF) classifier as an objective function. Additionally, Binary Differential Evolution Algorithm (BDEA) was suggested by He et al. [9] to select the best features' subset and this selection step is used as a preprocessing step to Support Vector Machine (SVM) and Radial Basis Function (RBF) network.

Regarding Optimum-Path Forest (OPF) classifier, which was suggested by Papa et al. [10,11], it was reported to have faster-training capabilities (from ten to thousand times) than the other classifiers without affecting the accuracy [6,11].

Flower Pollination Algorithm (FPA), which was suggested by [12], is inspired from flower pollination process of flowering plants. In the study of Rodrigues et al. [8], the binary version of FPA for feature selection task that is called Binary Flower

☆ This paper has been recommended for acceptance by G. Borgefors.

* Corresponding author. Tel.: +20 82 221 1255.
E-mail addresses: safyfci@gmail.com, safy_1988_8_10@yahoo.com (S. AbdEl-Fattah Sayed), e.nabil@fci-cu.edu.eg, emadnabilcs@gmail.com (E. Nabil), a.badr.fci@gmail.com (A. Badr).

http://dx.doi.org/10.1016/j.patrec.2016.03.014
0167-8655/

Pollination Algorithm (BFPA) was proposed, in which the search space is modeled as a d-dimensional boolean lattice, where each possible solution or a string of bits denotes whether a feature will belong to the final set of features or not.

The Clonal Selection Algorithm (CSA) is an efficient metaheuristic algorithm that is inspired from the human body immune system, the CSA was hybridized with many other metaheuristic algorithms and the resultant algorithm proved to be better than the original algorithms separately. Successful hybridization of CSA with SVM was performed by Ding and Li [13] and Zhao et al. [14]. Additionally, CSA was combined with ACO to solve TSP problem in [15], and with Harmony Search for fuzzy classification systems in [16], as well as with Differential Evolution for the training cascade correlation neural network in [17].

The current study has investigated the efficiency of combining CSA with FPA to compose a new hybrid algorithm called Binary Clonal Flower Pollination Algorithm (BCFA) to solve the feature selection problem. The accuracy of OPF classifier was used as an objective function to be maximized. The experiments were applied on three public datasets from UCI machine learning repository [18]. The results of the proposed algorithm were compared with the results of four well-known algorithms, namely: BCSA, BBA, BDEA, and BFPA.

The remaining of the paper is organized in four sections, where Section (2) introduces a theoretical background about CSA, FPA, BFPA, and OPF classifier, while Section (3) presents the proposed BCFA, as well, Section (4) presents the experimental results. Finally, conclusions and future work are stated in Section (5).

2. Theoretical background

2.1. The Clonal Selection Algorithm

CSA is used by the natural immune system to define the basic features of an immune response to an antigenic stimulus, as proposed in [19]. The algorithm is shown to be an evolutionary strategy, and capable of solving complex machine learning tasks like pattern recognition and multi-modal optimization tasks. The main features of the clonal selection theory [20] are as follows:

- The new cells are copies of their parents (clones) subjected to a mutation mechanism with high rates (somatic hypermutation).
- Elimination of newly differentiated lymphocytes carrying self-reactive receptors.
- Proliferation and differentiation on contact of mature cells with antigens.

The algorithm can be described as follows [19]:

1. Generate a set P of candidate solutions, composed of the subset of memory cells M added to the remaining Pr population $P = Pr + M$;
2. Determine (Select) the n best individuals of the population Pn, based on an affinity measure;
3. Reproduce (Clone) these n best individuals of the population, giving rise to a temporary population of clones C. The clone size is an increasing function of the affinity with the antigen;
4. Submit the population of clones to a hyper mutation scheme, where the hyper mutation is proportional to the affinity of the antibody with the antigen. A maturated antibody population is generated in C^*;
5. Re-select the improved individuals from C^* to compose the memory set M. Some members of P can be replaced by other improved members of the set C^*;
6. Replace d antibodies by novel ones (diversity introduction). The lower affinity cells have higher probabilities to be replaced;

Table 1
Parameters used for each technique.

Algorithm	Parameters
BCFA	$P = 0.8$, Best solutions to clone $= 10$, $\lambda = 1.5$
BFPA	$P = 0.8$, $\lambda = 1.5$
BCSA	$Pa = 0.25$, $\alpha = 0.1$, $\lambda = 1.5$
BBA	$\alpha = 0.9$, $\gamma = 0.9$, $f_{min} = 0$, $f_{max} = 2$, $A = 0.5$, $r = 0.5$
BDEA	$\beta = 0.5$, $Cr = 0.5$

2.2. Flower Pollination Algorithm

In 2012, Yang proposed a natural inspired algorithm called FPA [12], which is inspired by the pollination process of flowering plants. The objective of this algorithm is the survival of the best and the optimal reproduction of plants, which is, in fact, an optimization process of plant species. The FPA is governed by the four basic rules [12] bellow:

1. Biotic and cross-pollination are considered as global pollination process with pollen-carrying pollinators performing Lévy flights.
2. Abiotic and self-pollination are considered as local pollination.
3. Flower constancy can be considered as the reproduction probability is proportional to the similarity of two flowers involved.
4. Local pollination and global pollination are controlled by the switching probability $p \in [0, 1]$. Due to the physical proximity and other factors such as the wind, local pollination has a significant fraction p in the overall pollination activities.

In the global pollination step (Rules 1 and 3), the flower pollens are carried by pollinators such as insects. The pollens can travel over a long distance using Lévy flights distribution. The global pollination can be represented mathematically as,

$$x_i^{(t+1)} = x_i^t + \alpha L(\lambda)(g^* - x_i^t) \tag{1}$$

where,

$$L(\lambda) = \frac{\lambda.\Gamma(\lambda).\sin(\lambda)}{\pi} \cdot \frac{1}{s^{1+\lambda}}, \quad s > 0, \tag{2}$$

where x_i^t is the pollen i (solution vector) at the iteration t and g^* is the current best solution found among all solutions at the current generation, while α is the scaling factor to control the step size, s is the step size, $L(\lambda)$ is the Lévy flight step size corresponding to the strength of the pollination and $\Gamma(\lambda)$ stands for the gamma function, where the value of λ is in the range of $1 \leq \lambda \leq 2$. In our study, we have used $\lambda = 1.5$ as described in Table 1.

The Local Pollination (Rule 2) can be represented mathematically as,

$$x_i^{(t+1)} = x_i^t + \varepsilon(x_j^t - x_k^t), \tag{3}$$

where x_j^t and x_k^t stand for the pollen from different flowers j and k of the same plant species. In order to mimic the local and the global flower pollination, the switching probability P is used (Rule 4). Algorithm 1 presents the pseudo-code of FPA [12].

2.3. Binary Flower Pollination Algorithm

BFPA was proposed by Rodrigues et al. [8], in which the search space is modeled as a d-dimensional boolean lattice where the solutions are updated across the corners of a hypercube. Since feature selection problem is to select a specific feature or not, so the solution is represented as a binary vector, where 1 indicates a feature will be selected to compose the new dataset and 0 otherwise. Sigmoid function is used to build this binary vector by the following equation:

$$S(x_i^j(t)) = \frac{1}{1 + e^{-x_i^j(t)}}, \tag{4}$$

Algorithm 1 Flower Pollination Algorithm.

```
1: Objective min or max f(x), x = (x_1, ..., x_n);
2: Initialize a population of n flowers/pollen gametes with random solutions;
3: Find the best solution g* in the initial population;
4: Define a switch probability p ∈ [0, 1];
5: while (t < MaxGeneration) do
6:   for i = 1 : n (all n flowers in the population) do
7:     if rand < p, then
8:       Draw a (d-dimensional) step vector L which obeys a Lévy flight distribution;
9:       Global pollination via Eq. (1);
10:    else
11:      Draw ϵ from a uniform distribution in [0,1];
12:      Randomly choose j and k among all the solutions;
13:      Do local pollination via Eq. (3);
14:    end if
15:    Evaluate new solutions;
16:    if new solutions are better, update them in the population;
17:  end for
18:  Find the current best solution g*;
19: end while
```

Thus, Eqs. (1) and (3) will be replaced by the following equation:

$$x_i^j(t) = \begin{cases} 1 & \text{if } S(x_i^j(t)) > \sigma, \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

In which $x_i^j(t)$ represents the new pollen (solution) i with the jth feature vector, where $i = 1, 2, \ldots m$ and $j = 1, 2, \ldots d$, at the iteration t and $\sigma \sim U(0, 1)$.

2.4. The Optimum Path Forest classifier

OPF [10,11] is considered a supervised classification method, in which the training set is interpreted as a complete graph. It assigns to any path in the graph the maximum arc weight along it. Each node in the graph is represented as a feature vector, and each edge connects a pair of nodes, constituting a fully connected graph.

In our study, the dataset was divided into Z_1, Z_2, Z_3 and Z_4 subsets representing the training, learning, evaluating and testing subsets, respectively. Let (Z_1, A) be a complete graph whose nodes are samples in Z_1 and any pair of samples defines an arc in $A = Z_1 x Z_1$ as was explained in Fig. 1a. $\lambda(s)$ is the function that assigns the correct class label i to any sample $s \in Z_2 \bigcup Z_3 \bigcup Z_4$. $S \subset Z_1$ is a set of prototype samples of all classes.

As regard to the steps OPF classifier, they include the training, learning, and testing steps. The purpose of the training step is to find out the optimum-path forest in the complete graph (Z_1, A). The found forest is rooted in a special set $S \subset Z1$ of prototypes. In this respect, a Minimum Spanning Tree (MST) could be computed over the original graph and the samples that share arcs between different classes are marked as the prototypes (Fig. 1b). The data illustrated in Fig. 1c revealed that the resulting tree is divided into a group of trees rooted in these prototypes to compute the forest. For the path value function, f_{max} is given by the following equations:

$$f_{max} = \begin{cases} 0, & \text{if } s \in S, \\ +\infty, & \text{otherwise,} \end{cases}$$

$$f_{max}(\pi_s . < s, t >) = \max\{f_{max}(\pi_s), d(s, t)\}, \tag{6}$$

where $d(s, t)$ denotes the distance between samples s and t, π_s is a path in the graph that ends in sample $s \in Z_1$, where a path is a sequence of distinct samples $\pi_s =< s1, s2, \ldots, s >$ with terminus at a sample s and a path is said trivial if $\pi_s =< s >$, and $(\pi_s . < s, t >)$ is the concatenation between π_s and the arc $< s, t >$, $t \in Z_1$. In such a way, $f_{max}(\pi_s . < s, t >)$ computes the maximum distance between adjacent samples along the path $\pi_s . < s, t >$. During the learning step, the classifier can improve its performance by learning from its errors along time of use. This step uses the learning set Z_2 to improve the composition samples of the training set Z_1. The algorithm projects an instance I of a given classifier from Z_1 and evaluates it on Z_2. The misclassified samples of Z_2 are selected and replaced by random selected non-prototype samples of Z_1. This procedure assumes that the most informative samples can be obtained by learning from the errors. The new sets Z_1 and Z_2 are then used to repeat the process during the few iterations T. The instance of the classifier that achieved the highest accuracy along the iterations is selected to be used in the testing step. The accuracy function $L(I)$ of the classifier instance I can be computed by the following equations:

$$L(I) = 1 - \frac{\sum_{i=1}^{c} E(i)}{2c} \tag{7}$$

where c is the number of classes and $E(i)$ is computed as follows:

$$E(i) = e_{i,1} + e_{i,2}, \tag{8}$$

where,

$$e_{i,1} = \frac{FP(i)}{|Z_2| - |NZ_2(i)|}, \quad and, \quad e_{i,2} = \frac{FN(i)}{|NZ_2(i)|}, \quad i = 1, \ldots, c \tag{9}$$

where $NZ_2(i)$ is the number of samples in Z_2 from each class i, and $FP(i)$ and $FN(i)$ are the false positives and false negatives, respectively. $FP(i)$ is the number of samples from other classes that were classified as being from the class i in Z_2, and $FN(i)$ is the number of samples from the class i that were incorrectly classified as being from other classes in Z_2. In the testing/classification step, the classification of a new sample $t \in Z_3 \bigcup Z_4$ is done based on the distance $d(s,t)$ between t and each training node $s \in Z1$, see Fig. 1d. This distance is computed and used to weight the edges. The training node s that has offered the minimum path-cost will conquer the test sample t as in Fig. 1e. The path can be identified incrementally, by computing the optimum cost $C(t)$ of each path between the test sample t and a training node s as follows:

$$C(t) = \min\{\max\{C(s), d(s, t)\}\}, \quad \forall s \in Z_1 \tag{10}$$

According to Eq. (10), it can be assumed that $s^* \in Z_1$ is the node that offered the minimum path-cost with the test sample t. In this case, this node is the predecessor $P(t)$ in the optimum path $P^*(t)$). Respectively, $L(s^*) = \lambda(R(t))$, where $\lambda(t)$ is the function that assigns the correct class label, and $R(t)$ is the function that get the root of t and this root is one of the prototypes $R(t) \in S$. Classification simply assigns $L(s^*)$ as the class of t. An error occurs when $L(s^*) \neq \lambda(t)$.

Algorithm 2 represents the pseudo-code of OPF classifier that is used in calculating the accuracy (fitness function) of each solution in the population, where this accuracy should be measured every time the solution changes its position.

The algorithm starts with the main loop in Lines 1–10 to learn the classifier to generate the best instance classifier. For this purpose, the classifier is trained over $\acute{Z}_1$ and then evaluated over $\acute{Z}_2$. The recognition accuracy over $\acute{Z}_2$ is stored in *acc* variable and then compared with the last best accuracy *bestAcc*. The best instance classifier will be updated if the accuracy of the current classifier is better than the last, otherwise, the last is kept. Then, the miss-classified samples from $\acute{Z}_2$ are replaced with random non-prototypes samples from $\acute{Z}_1$, this step makes the classifier learn from its errors and increase its performance. In Line 11, the best instance of the classifier is detected to use its accuracy as the fitness function of the solution.

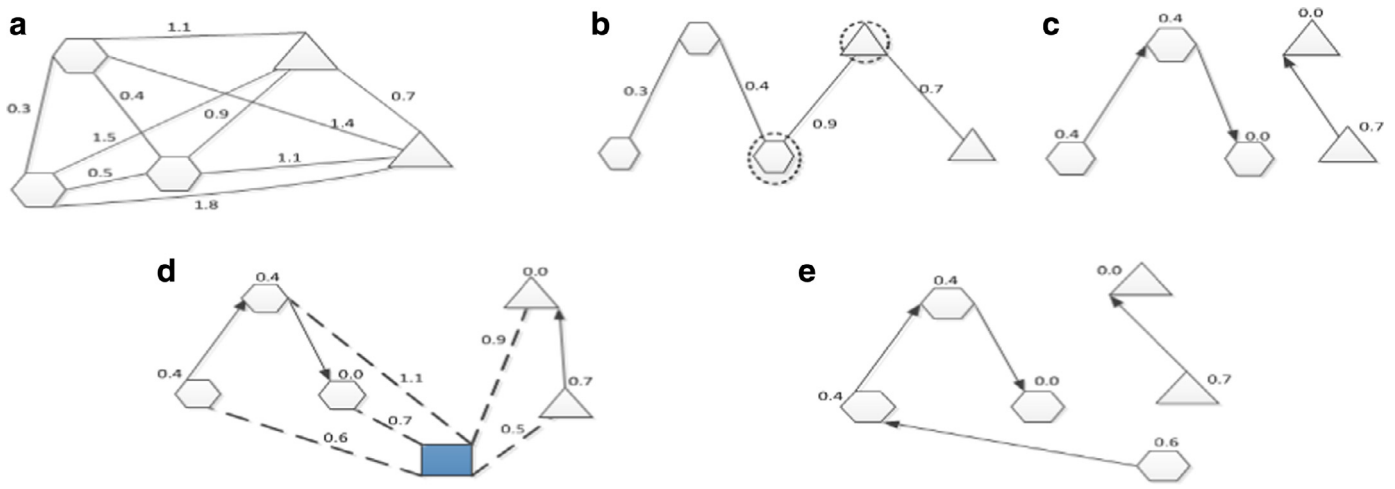

Fig. 1. (a) Complete weighted graph for a simple training set where the shapes hexagons and triangles denote examples of samples of different classes, (b) MST and prototypes marked with dashed circle, (c) optimum-path forest generated at the final of the training step, (d) Test sample (square shape) and its connections (dashed lines) with the training nodes for the classification phase and (e) The optimum path from the most strongly connected prototype, its label is the hexagon class, so the square sample (test sample) is associated to the hexagon class although its nearest training sample is the class triangle. The values above the nodes are their costs after training, and the values above the edges stand for the distance between their corresponding nodes.

Algorithm 2 OPF classifier pseudo-code.

Input: Training set $\acute{Z}_1$ and evaluating set $\acute{Z}_2$, number of iterations T for learning.
Output: The best instance of the classifier.

```
1: while (t < T) do
2:   Train the classifier over Ź1;
3:   Evaluate the classifier over Ź2;
4:   Calculate the accuracy of the classifier instance and stores in
     acc;
5:   if (acc > bestAcc) then
6:     Update the current classifier instance as the best instance;
7:     Update bestAcc with acc;
8:   end if
9:   Replace miss-classified samples from Ź2 with random non-
     prototypes samples from Ź1;
10: end while
11: return The best instance of the classifier to be the accuracy of
    the solution;
```

3. A Binary Clonal Flower Pollination Algorithm

The CSA has good exploitation characteristics via cloning of good solutions, in the other hand, the FPA has good exploration characteristics via Lévy flight. In the proposed algorithm, we gathered the characteristics of CSA and FPA to compose a new hybrid algorithm that utilizes these features. The main idea is to use the switch probability p to control the switching of local pollination and global pollination [12]. In the global pollination, the flower pollens are carried by pollinators such as insects. These pollens can fly to the next position using Lévy flight distribution [21] as mentioned before in Eqs. (1) and (5) for the binary position. However, in the local pollination, CSA is used to select the best pollens and then cloned according to their affinity function (fitness function). The cloned pollens are matured using the local pollination properties according to Eq. (3) and then Eq. (5) is applied for the binary position. Finally, the best pollens are selected for the next population. Algorithm 3 represents the pseudo-code of the proposed algorithm.

The algorithm starts with the first loop in Lines 1–6, in which the population is initialized where each solution position is represented as a binary string of random values and its fitness value f_i is defined. The loop in Lines 7–10 is responsible for creating the new training set $\acute{Z}_1$ and the evaluating set $\acute{Z}_2$ from the input data Z_1 and Z_2, respectively, where both sets contain only features in the solution $x_i^j(t) \neq 0,\ \forall j = 1, 2, \ldots d$. Then, OPF, that is explained in Algorithm 2, is trained over $\acute{Z}_1$ and evaluated over $\acute{Z}_2$ to calculate the fitness function f_i of each solution. The main loop in Lines 11–34 is the core of the proposed algorithm, in which Line 13 uses the switching probability P to indicate the type of the pollination that will be followed in calculating the next position. In the case of the global pollination (Lines 14–20), Eq. (1), will be applied on each solution and restricted by Eq. (5) for the binary position. For each position, the OPF in Algorithm 2 is applied with the new sets $\acute{Z}_1$ and $\acute{Z}_2$ to get the fitness function and store the result in acc variable. Then, acc is compared with f_i of the solution i: if the latter is better than acc, the old fitness value is kept; otherwise, the fitness value is then updated.

On the contrary, the local pollination with the clonal selection will be applied (Lines 22–32). In Lines 22-23, best solutions are selected and cloned according to their affinity (fitness function) where the clone size is an increasing function of their affinity. In the loop (Lines 24–30) the local pollination rules, according to Eq. (3), will be applied on each solution and then restricted by Eq. (5) for the binary position. Then, the OPF Algorithm 2 is used for calculating the fitness function. In Lines 31-32, the best solutions are selected from the matured and the original populations to generate the next one. At the end of the algorithm in Line 35, the best solution over all iterations is detected that maximizes the accuracy of the OPF classifier. Then, the selected features of the obtained best solution will be used to test the classifier (testing step).

4. Experimental results

In order to assess the performance of the proposed algorithm, we compared the results of BCFA against those of (BFPA), (BCSA), (BBA), and (BDEA). The used parameters for each technique were illustrated in Table 1.

The experiments were carried out using a PC Intel(R)Core (TM)i5-2430M CPU 2.40 GHz with 8 GB RAM, Windows 7 operating system and eclipse java Luna development editor.

Three public labeled datasets from UCI machine learning repository [18] were used in the experiments, namely Australian, Breast

Algorithm 3 Binary Clonal Flower Pollination Algorithm.

Input: Training set Z_1 and evaluating set Z_2, population size n, number of features d, number of iterations T and switch probability p.

Output: Subset of features that gives the maximum accuracy over Z_2.

for each solution i ($\forall i = 1, 2, \ldots n$) **do**
  for each dimension j ($\forall j = 1, 2, \ldots d$) **do**
    $x_i^j(0) \leftarrow Random\{0, 1\}$;
  end for
  $f_i \leftarrow -\infty$, f_i stores a fitness value of each solution;
end for
for each solution i ($\forall i = 1, 2, \ldots n$) **do**
  Create new $\acute{Z}_1$ and $\acute{Z}_2$ from Z_1 and Z_2 respectively, such that both contains only features in the solution $x_i^j(t) = 1$, $\forall j = 1, 2, \ldots d$;
  $f_i \leftarrow$ Get the fitness function by applying Algorithm 2 with $\acute{Z}_1$ and $\acute{Z}_2$;
end for
for each iteration t ($t = 1, 2, \ldots T$) **do**
  $rand \leftarrow Random[0, 1]$;
  if $rand > p$, **then**
    for i = 1 : n (each solution in the population) **do**
      Apply global pollination with Lévy flight by using Eq. (5);
      Calculate the fitness function of a solution by applying Algorithm 2 with the new $\acute{Z}_1$ and $\acute{Z}_2$ of new position and store it in Acc;
      if ($Acc > f_i$) **then**
        Update the solution position to a new position;
      end if
    end for
  else
    Select the best solutions;
    Clone the best solutions according to their fitness function;
    for i = 1 : n (all n solutions in the clones population) **do**
      Apply local pollination on the solution using Eq. (5) to compose the matured population;
      Calculate the fitness function of the solution by applying Algorithm 2 with the new $\acute{Z}_1$ and $\acute{Z}_2$ of new position that are created from Z_1 and Z_2 respectively, and store it in Acc;
      if ($Acc > f_i$) **then**
        Update the solution position to the matured position;
      end if
    end for
    Select the best solutions from the original population and the matured population for the next generation;
    Replace the current population by the new population;
  end if
end for
return the selected features of the best solution that maximizes an accuracy of the OPF classifier, these selected features would be used in a testing phase.;

Cancer and German Number datasets. The main characteristics of the three datasets are presented in Table 2.

Each dataset is randomly divided into four disjoint subsets Z_1, Z_2, Z_3, and Z_4, which correspond to the training, learning, validation and testing sets with percentages of 30%, 20%, 20% and 30%, respectively. The training set Z_1 is a base set to construct the classifier and the accuracy of the classifier is little affected by this set, while the learning set Z_2 is used to learn the classifier by randomly interchanging samples of Z_1 with misclassified samples of Z_2 to improve the composition of samples in Z_1. The training and learning sets are applied inside the optimization fitness. The validation set Z_3 is used to validate the quality of the selected features before using them in the testing step. The testing set Z_4 is kept hidden for both the classifier and the optimizer for the final evaluation of the whole feature selection and classification system.

Table 2
Characteristics of datasets used in this work.

Dataset	# Samples	# Samples per class	# Features	# Classes
Australian	690	+:307 Class 1 -:383 Class 2	14	2
Breast Cancer	699	+:458 Class 1 -:241 Class 2	10	2
German Number	1000	+:700 Class 1 -:300 Class 2	24	2

In the current experiment, we applied the threshold approach, that was used in [6], to retrieve the most informative features. The values of the threshold ranged from 10% to 90%, and for each threshold value of the running time, the selected features of the solution that achieved the highest accuracy over Z_2 were stored in the vector until the training and learning steps are finished. Then, the solutions, that are stored in the vector, were used over the validation set Z_3, and finally, the best solution that achieved the highest accuracy over Z_3 would be used to test the testing set Z_4.

In details, to guarantee the best selected features, we suggested a validation step before the testing step. For this purpose, during each 10% of the running time, the selected features of the solution that achieved the best accuracy are stored in the vector, then selected features in this vector are used to validate the unknown validation set Z_3. Finally, the selected features that achieved the best accuracy over Z_3 will be used in the final test step over Z_4 as depicted in Fig. 2.

We have used a population of 20 solutions, the number of internal iterations for optimization equals 1000, and the number of times repeating the stochastic optimization equals 10.

Table 3 displays the mean accuracy results over the testing set that are obtained from the different optimizers for the three datasets. Through out the rest of the paper, the best values are formatted in bold. As it is shown in Table 3, BCFA achieved the best accuracy results in Breast Cancer and German Numeric datasets and surpasses BFPA, BBA and BDEA in Australian dataset, also BCFA has the best accuracy mean.

For testing the performance of the optimizers, the standard deviation of each optimizer is calculated through the 10 different runs with different initial random solutions. The results that are represented in Table 4 show that BCFA has the minimum standard deviation in case of German Numeric dataset and the minimum mean of the standard deviations of the three datasets.

Table 5 shows the classification error of the different optimizers for the different datasets. We can remark that the mean error of BCFA is better than other algorithms.

The statistical Wilcoxon Signed-Rank Test [22] is performed to verify whether there is a significant difference between BCFA and the other four techniques used in our work. Table 6 displays the p-values, the bold values indicates a statistical difference between the BCFA and the respective technique, taking significance level of $\alpha = 0.05$. It can be seen that there is a statistical difference between BCFA and BCSA for German Numeric dataset, also a difference between the BCFA and BBA for Australian, Breast Cancer and German Number datasets, moreover there is a difference between the BCFA and the BDEA for the Breast Cancer dataset.

Table 7 shows the average number of the selected features for each dataset. It is clear that BCFA selects the minimum number of the features in comparison with other algorithms, while it keeps

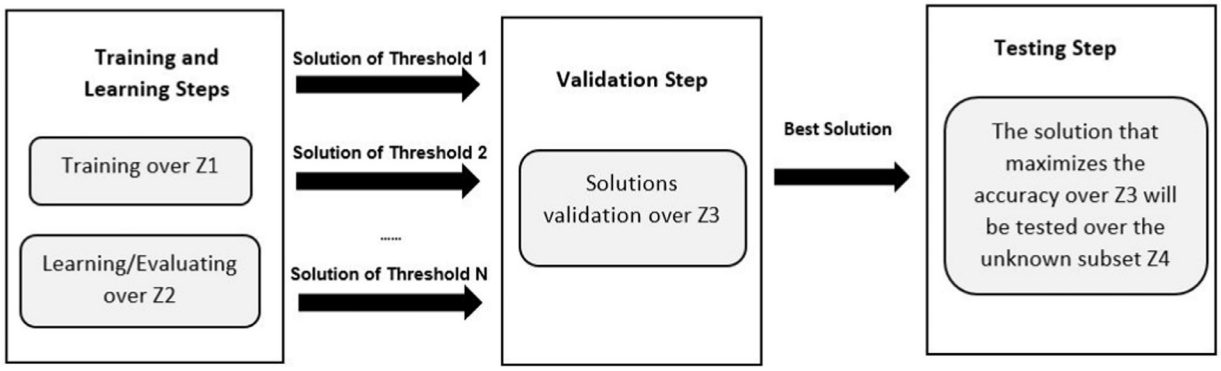

Fig. 2. The used methodology in our experiments.

Table 3
Average classification accuracy obtained from the different optimizers for different datasets.

Dataset	BCFA(%)	BFPA(%)	BCSA(%)	BBA(%)	BDEA(%)
Australian	76.86	72.55	**77.32**	71.10	74.55
Breast Cancer	**96.31**	95.08	95.13	89.17	95.29
German Numeric	**60.34**	56.31	57.09	55.91	56.32
Mean	**77.84**	74.65	76.51	72.06	75.39

Table 4
Standard deviation of the obtained accuracy of different optimizers for the different datasets.

Dataset	BCFA	BFPA	BCSA	BBA	BDEA
Australian	5.21	6.21	5.77	**4.18**	7.04
Breast Cancer	1.42	1.96	**1**	7.93	1.54
German Numeric	**2.63**	4.89	3.82	3.37	4.04
Mean	**3.09**	4.35	3.53	5.16	4.21

Table 5
Classification error on test data for different optimizers in comparison with the data with all features.

Dataset(%)	BCFA(%)	BFPA(%)	BCSA(%)	BBA(%)	BDEA(%)
Australian	23.14	27.45	**22.68**	28.90	25.45
Breast Cancer	**3.69**	4.92	4.87	10.83	4.71
German Numeric	**39.66**	43.69	42.91	44.09	43.68
Mean error	**22.16**	25.35	23.49	27.94	24.61

Table 6
Wilcoxon Signed-Rank test evaluation: *p*-values computed between the proposed algorithm and the BFPA, BCS, BBA and BDE algorithms, significance level $\alpha = 0.05$.

Dataset	BFPA	BCSA	BBA	BDEA
Australian	0.241	0.721	**0.028**	0.508
Breast Cancer	0.114	0.169	**0.017**	**0.009**
German Numeric	0.059	**0.047**	**0.028**	0.059

better classification performance as outlined in Table 5 where it has the minimum mean classification error and the best mean classification accuracy as described in Table 3.

Table 8 displays the execution time for all considered optimization techniques. We can notice that BCFA is the fastest technique in Breast Cancer and German Numeric datasets and faster than BFPA, BBA and BDEA in Australian dataset, in the same time BCFA has the best average of execution time.

From the experimental results, it has been approved that BCFA outperforms four well-known algorithms and achieved better results in the accuracy and the number of the selected features. The secret behind surpassing the proposed algorithm over the other four algorithms is that CSA has a good exploitation property and FPA with Lévy flight behavior has a good exploration property. The proposed algorithm, that is composed of two efficient tools, led to reach these remarkable results.

Table 7
Average number of selected features obtained from the different optimizers for the different datasets.

Dataset	BCFA	BFPA	BCSA	BBA	BDEA
Australian	**8.3**	10.2	9.4	9.4	10.4
Breast Cancer	**6.7**	7.5	7.5	7.5	8.2
German Numeric	15.6	16	16	**15.5**	16.8
Mean	**10.2**	11.2	11	10.8	11.8

Table 8
Average execution time in seconds of the five optimizers for the three datasets.

Dataset	BCFA	BFPA	BCSA	BBA	BDEA
Australian	3943	4826.4	**3858**	6138	3939.6
Breast Cancer	**2990.8**	7628.5	4913.3	4917.2	7612
German Numeric	**13234.4**	21636.7	21663.9	27059	26715.9
Average	**6722.8**	11363.9	10145.1	12704.7	12755.8

5. Conclusions

In this paper, we propose a new hybrid algorithm, that combines CSA with FPA to compose BCFA. The proposed algorithm is used to solve the feature selection problem. The experiments are implemented on three public benchmark datasets from UCI datasets. The experimental results show that the proposed algorithm has achieved remarkable results against four well-known algorithms, namely BFPA, BCSA, BBA, and BDEA. The results proved that BCFA is able to achieve the best classification accuracy using the smallest number of selected features in less time.

For future work, BCFA will be applied to many other public datasets, real world problems, and will be used with more classifiers like SVM, Artificial Neural Networks (ANN) and k-Nearest Neighbors (k-NN) to verify and extend this approach. The hybrid BCFA will be applied in different applications like machine learning, data mining, pattern recognition, job scheduling and text processing.

References

[1] J.B. Jona, N. Nagaveni, Ant-cuckoo colony optimization for feature selection in digital mammogram, Pak. J. Biol. Sci. 17 (2) (2014) 266–271.
[2] M. Nazir, A. Majid-Mirza, Ali-Khan, Pso-ga based optimized feature selection using facial and clothing information for gender classification, J. Appl. Res. Technol. 12 (1) (2014) 145–152.
[3] H. Banati, M. Bajaj, Fire fly based feature selection approach, Int. J. Comput. Sci. Issues(IJCSI) 8 (4) (2011) 473–480.

[4] N. Suguna, K. Thanuskodi, A novel rough set reduct algorithm for medical domain based on bee colony optimization, J. Comput. 2 (6) (2010) 49–54.
[5] M. Mafarja, D. Eleyan, Ant colony optimization based feature selection in rough set theory, Int. J. Comput. Sci. Electr. Eng. 1 (2) (2013) 244–247.
[6] R. Nakamura, L. Pereira, K. Costa, D. Rodrigues, J. Papa, X.-S. Yang, A binary bat algorithm for feature selection, in: Proceedings of the Conference on Graphics, Patterns and Images (SIBGRAPI), IEEE, Ouro Preto, 2012, pp. 291–297.
[7] D. Rodrigues, L. Pereira, T. Almeida, J. Papa, A. Souza, C. Ramos, X.-S. Yang, Bcs: A binary cuckoo search algorithm for feature selection, in: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), IEEE, Beijing, 2013, pp. 465–468.
[8] D. Rodrigues, X. Yang, A. Souza, J. Papa, Binary flower pollination algorithm and its application to feature selection, in: X. Yang (Ed.), Recent Advances in Swarm Intelligence and Evolutionary Computation, Studies in Computational Intelligence, Springer International Publishing, 2015, pp. 85–100.
[9] X. He, Q. Zhang, N. Sun, Y. Dong, Feature selection with discrete binary differential evolution, in: Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence, 4, IEEE, Shanghai, 2009, pp. 327–330.
[10] J.P. Papa, A.X. Falcao, C.T.N. Suzuki, Supervised pattern classification based on optimum-path forest, Int. J. Imaging Syst. Technol. 19 (2) (2009) 120–131.
[11] J.P. Papa, A.X. Falcao, V.H.C. Albuquerque, J.M.R. Tavares, Efficient supervised optimum-path forest classification for large datasets, Pattern Recognit. 45 (1) (2012) 512–520.
[12] X. Yang, Flower pollination algorithm for global optimization, in: J. Durand-Lose, N. Jonoska (Eds.), Unconventional Computation and Natural Computation, Springer, Berlin Heidelberg, 2012, pp. 240–249.
[13] S. Ding, S. Li, Clonal selection algorithm for feature selection and parameters optimization of support vector machines, in: Proceedings of the Second International Symposium on Knowledge Acquisition and Modeling, 2, IEEE, Wuhan, 2009, pp. 17–20.
[14] M. Zhao, C. FU, L. JI, K. Tang, M. Zhou, Clonal selection algorithm based on feature antibodies for feature selection and parameter optimization of support vector machines, J. Comput. Inf. Syst. 7 (4) (2011) 1163–1172.
[15] L. Yong, L. Sunjun, A hybrid model for solving tsp based on artificial immune and ant colony, in: Proceedings of the International Conference on Information Engineering and Computer Science (ICIECS), IEEE, Wuhan, 2009, pp. 1–5.
[16] X. Wang, X. Gao, S.J. Ovaska, Fusion of clonal selection algorithm and harmony search method in optimisation of fuzzy classification systems, Int. J. Bio Inspir. Comput. 1 (1/2) (2009) 80–88.
[17] X.Z. Gao, X. Wang, S.J. Ovaska, Fusion of clonal selection algorithm and differential evolution method in training cascade correlation neural network, Neurocomputing 72 (10–12) (2009) 2483–2490.
[18] C. Blake, E. Keogh, C. Merz, Uci repository of machine learning databases, 1998, (www.ics.uci.edu/mlearn/MLRepository.html/) (accessed 02.04.16).
[19] L.N.D. Castro, F.J.V. Zuben, The clonal selection algorithm with engineering applications, in: Proceedings of the Workshop on Artificial Immune Systems and Their Applications, Las Vegas, USA, 2000, pp. 36–37.
[20] F.M. Burnet, Clonal selection and after, in: G.I. Bell, A.S. Perelson, G.H. Pimbley (Eds.), Theoretical Immunology, Marcel Dekker Inc., New York, 1978, pp. 63–85.
[21] I. Pavlyukevich, Lévy flights, non-local search and simulated annealing, J. Comput. Phys. 226 (2) (2007) 1830–1844.
[22] F. Wilcoxon, Individual comparisons by ranking methods, Biom. Bull 1 (6) (1945) 80–83.