

An Efficient Binary Clonal Selection Algorithm with Optimum Path Forest for Feature Selection

Emad Nabil¹

Department of Computer Science
Faculty of Computers and Artificial Intelligence
Cairo University, Giza, Egypt
Department of Computer Science
Faculty of Computer and Information Systems
Islamic University of Madinah
Madinah, Saudi Arabia

Safinaz Abdel-Fattah Sayed²

Department of Computer Science, Faculty of Computers and
Information, Luxor University, Luxor, Egypt
Department of Computer Science, Faculty of Computers and
Artificial Intelligence, Cairo University, Giza, Egypt

Hala Abdel Hameed³

Faculty of Computers and Information
Fayoum University, Egypt, Khaybar Community College
Taibah University, Saudi Arabia

Abstract—Feature selection is an important step in different applications such as data mining, classification, pattern recognition, and optimization. Until now, finding the most informative set of features among a large dataset is still an open problem. In computer science, a lot of metaphors are imported from nature and biology and proved to be efficient when applying them in an artificial way to solve a lot of problems. Examples include Neural Networks, Human Genetics, Flower Pollination, and Human Immune system. Clonal selection is one of the processes that happens in the human immune system while recognizing new infections. Mimicking this process in an artificial way resulted in a powerful algorithm, which is the Clonal Selection Algorithm. In this paper, we tried to explore the power of the Clonal Selection Algorithm in its binary form for solving the feature selection problem, we used the accuracy of the Optimum-Path Forest classifier, which is much faster than other classifiers, as a fitness function to be optimized. Experiments on three public benchmark datasets are conducted to compare the proposed Binary Clonal Selection Algorithm in conjunction with the Optimum Path Forest classifier with other four powerful algorithms. The four algorithms are Binary Flower Pollination Algorithm, Binary Bat Algorithm, Binary Cuckoo Search, and Binary Differential Evolution Algorithm. In terms of classification accuracy, experiments revealed that the proposed method outperformed the other four algorithms and moreover with a smaller number of features. Also, the proposed method took less average execution time in comparison with the other algorithms, except for Binary Cuckoo Search. The statistical analysis showed that our proposal has a significant difference in accuracy compared with the Binary Bat Algorithm and the Binary Differential Evolution Algorithm.

Keywords—Feature selection; artificial immune system; clonal selection algorithm; optimization; optimum path forest

I. INTRODUCTION

Artificial Immune System (AIS) uses ideas inspired by the immune system of the human body for solving different kinds of problems in various research areas like pattern recognition, data mining, machine learning, and optimization. Clonal

selection is an important branch in AIS that is responsible for the response of the immune system to harmful antigens. It selects the cells (antibodies) that identified the antigens (Ag) to proliferate. Then, the procedure of affinity maturation is applied to the selected cells to improve their affinities to be suitable for the selective Ag[1]. The mentioned characteristic of the Clonal Selection Algorithm helped to make it very appropriate for solving multidimensional optimization tasks, where the optimization can be defined as a searching process for the best solution among the available solutions to a specific problem.

The target of the feature selection problem is to compose a subset that contains the best features selected among all features in a particular domain. The obtained features subset can be used to optimize an objective function of a certain problem, so the problem of feature selection can be categorized as an optimization problem. Solving this problem can be very useful in decreasing the dimensionality of the data and removing irrelevant and noisy data, subsequently, it will have a good effect on the implementation and execution of many applications.

Many natural inspired algorithms were used to find a solution to the problem of feature selection, such as Particle Swarm Optimization (PSO) [2, 3], Binary Ant Colony Optimization (ACO) [4, 5], Gravitational Search Algorithm (GSA) [6], Binary Differential Evolution (BDE) [7], Cloning Algorithm [8, 9], Artificial Fish Swarm (AFS) [10, 11], Harmony Search Algorithm (HSA) [12], Binary FireFly Algorithm (FFA) [13], Binary Cuckoo Search (BCS) [14], Binary Bat Algorithm (BBA) [15], Binary Flower Pollination Algorithm (BFPA) [16], and Binary Clonal Flower Pollination Algorithm (BCFA) [17].

The natural immune system uses the clonal selection to select the best cells that can recognize the antigens. The chosen cells are proliferated and then matured to improve their affinity to the particular antigens. The clonal selection concept has a serious role in the success of the human immune system and has an excellent ability of selection at work. As the feature

selection problem can be defined as selecting the optimal subset of features to improve the fitness function of a particular problem, thus, the clonal selection algorithm was chosen in the current study to find a solution for the feature selection problem, as it has achieved good results in solving many problems of different applications. Such as function optimization [18], pattern recognition [19], scheduling [20] and industrial engineering (IE) related problems [21].

In BCSA, the search domain is designed as an n -dimensional, where n refers to the features number. The concept of the algorithm is representing each solution as a binary set of coordinates that indicate where a feature will be selected or not. The fitness function to be maximized is the accuracy of the Optimum-Path Forest (OPF) classifier [22, 23]. The point is to train the classifier and calculate its accuracy every time the solution is mutated; so fast and robust classifier should be used to handle this task like OPF that has been used in many applications and achieved good results similar to that of Support Vector Machine (SVM) classifier but OPF is faster than SVM in the training phase [15].

Although there are many algorithms that were applied to find the solutions to the problem of feature selection as mentioned above, the accuracy of the classification and the speed of execution still need to be enhanced. In this paper, we introduced a modified binary clonal selection algorithm to improve the accuracy and the speed of solving the feature selection problem, taking into consideration reducing the number of features. The proposed algorithm was compared with BFPA, BCS, BBA, and BDEA. Public UCI datasets [24] were used in the experimental results. Experiments include sensitivity analysis and execution time comparison.

The paper is sectioned as follows; Clonal Selection Algorithm is presented in Section 2. The Optimum Path Forest (OPF) classifier is explained in Section 3. The proposed Binary Clonal Selection Algorithm (BCSA) is presented in Section 4. Experimental results and discussion are demonstrated in Section 5. Finally, Section 6 contains conclusions and future work.

II. THE CLONAL SELECTION ALGORITHM

The biological immune system is partitioned to innate immunity and adaptive immunity, mostly, the researchers propose ideas that are based on the latter. The natural immune system has an ability to protect the human body from the attack of harmful microorganisms; it can discriminate between the normal inhabitant microorganisms and harmful ones. The harmful organisms are foreign bodies that can stimulate our immune system, so they are called antigens (Ag). The immune system of the human body produces another component called antibody (Ab) to attack each antigen, and each Ag has a specific Ab to recognize it.

There are many important models that are based on the natural immune system, like Artificial Immune Network [25], Danger Theory Inspired Algorithms [26], Negative Selection Algorithms [27–29], Clonal Selection Algorithm [1] and Dendritic Cell Algorithms [30, 31].

The clonal selection is responsible for the adaptive response of the immune system to the foreign antigens as

proposed in [1], where the cells (antibodies) that detect these antigens are stimulated, cloned, and divided into plasma and memory cells. The algorithm of clonal selection can be considered as one of the evolutionary strategies that have the ability to solve the complicated problems in different areas. The features of clonal selection theory [32] can be listed as follows:

- All antibodies are mutated for maturation. The mutation can be seen as genetic changes for better recognition of antigens.
- The antibodies that carry self-reactive receptors are removed from the repertoire (the set of antibodies).
- Proliferation and differentiation for the most-stimulated antibodies.
- The best set of antibodies is chosen as memory cells for any future attacks.

The AIS algorithm [1] is explained as follows:

1) Initialize population of solutions P , some of these solutions are stored as a memory M and others are the remaining solutions P_r , so $P = P_r + M$;

2) The solutions n that achieved the highest affinity measure are selected to compose the population P_n ;

3) The selected solutions n are cloned (reproduced), generating a clone population C . The rate of cloning is proportional to the solution affinity with the objective function (antigen);

4) The clone population C is submitted to an affinity mutation process, the rate of mutation is inversely proportional to the solution affinity with the objective function (antigen), generating a matured population C^* ;

5) From the matured population C^* , reselect the highest affinity solutions to be stored in the memory part M , also some solutions of P can be exchanged with other matured solutions of C^* ;

6) The lowest affinity solutions d are replaced by new initialized solutions to increase the population diversity.

In optimization problems, the objective is to look for the best solution among all available solutions, the role of AIS is to develop the solutions depending on the mechanisms of the natural immune system like clonal selection, immune network theory, or other immune system concepts. The algorithm of clonal selection optimization consists of a set of candidate solutions (antibodies) and a set of objectives (antigens), where the antibody tries to match or catch (optimize) the antigen [33].

III. THE SUPERVISED OPTIMUM-PATH FOREST (OPF) CLASSIFIER

OPF is a supervised classifier that can deal with the labeled samples, it has faster training advantage than other classifiers like SVMs and ANNs [22, 23], so it is expected to be useful in the current study. OPF is a graph-based classifier in which the features are represented as graph nodes and these nodes are connected by using some adjacency relation, where the arc between two nodes can be defined as a sequence of

adjacent nodes. Euclidean norm is calculated to weight the arc between every two nodes, defining a complete graph.

In the current research, the dataset is partitioned into four subsets training set Z_1 , learning set Z_2 , evaluating set Z_3 , and testing set Z_4 . The graph is represented by (Z_1, A) , where the samples in Z_1 can be considered as the graph nodes and each pair of samples in $A = Z_1 \times Z_1$ can be performed as the graph arcs, as explained in Fig. 1(a), where there is a complete graph of different class samples (circles and stars). Function $\lambda(s)$ is responsible for assigning the correct class label i to any sample $s \in Z_2 \cup Z_3 \cup Z_4$. The set of prototype samples of all classes is represented by $S \subset Z_1$.

OPF classifier has three phases in its procedure [22, 23]: the training phase, the learning phase, and the classification/testing phase. In the training phase, the purpose is to generate an optimum-path forest that contains a group of discrete trees with optimum paths (OPTs) rooted in a special set $S \subset Z_1$ called prototypes. The algorithm minimum spanning tree (MST) should be applied to generate these prototypes that are samples with different class labels and have the same arcs, as explained in Fig. 1(b), where the dashed circles and stars are samples with different class labels, then the arcs that connect these samples are removed to produce a group of trees rooted in the generated prototype samples as described in Fig. 1(c), where there are two trees; one is rooted by a circle sample and the other is rooted by star sample.

The connectivity function for a path-cost F_{max} is calculated as follows [22, 23]:

$$f_{max}(\langle x \rangle) = \begin{cases} 0, & \text{if } x \in S \\ +\infty, & \text{Otherwise} \end{cases}$$

$$f_{max}(\pi_x \cdot \langle x, y \rangle) = \max\{f_{max}(\pi_x), d(x, y)\}, \quad (1)$$

In which, $d(x, y)$ represents the distance between the nodes x and y , and π is the path that can be defined as a sequence of adjacent samples where π_x is the path that ends in sample $x \in Z_1$. The path is defined as a trivial if $\pi_x = \langle x \rangle$. The function $f_{max}(\pi_x \cdot \langle x, y \rangle)$ calculates the maximum distance of the path $\pi_x \cdot \langle x, y \rangle$.

Now, the classifier has been initialized by the training set Z_1 during the first phase, then it will use the learning set Z_2 with Z_1 in the learning process during the learning phase. The target of this phase is learning OPF from its errors for enhancing its performance. The process starts with the training set Z_1 to initialize the classifier and generate an initial instance I that can be evaluated over the set Z_2 . Then, Z_2 samples that were misclassified are selected and exchanged with random selected non-prototype samples from Z_1 to generate the new sets Z_1 and Z_2 . The learning over the generated new sets Z_1 and Z_2 continues until the few iterations T are met. This technique helps in increasing the effective samples in the training set Z_1 and this is very important because this set will be used with the testing set in the final phase. Finally, the best classifier instance with the best accuracy will be employed in the final testing phase. The OPF accuracy is calculated by this equation:

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

In which, the function $L(I)$ is the classifier accuracy of the instance I , c refers to the classes number and the error function $E(i)$ is calculated as follows:

$$E(i) = e_{i,1} + e_{i,2} \quad (3)$$

Where,

$$e_{i,1} = \frac{FP(i)}{|Z_2| - |NZ_2(i)|}, \quad \text{and} \quad e_{i,2} = \frac{FN(i)}{|NZ_2(i)|}, \quad i = 1, 2, \dots, c \quad (4)$$

Where $NZ_2(i)$ is the samples number N in Z_2 for each class i in which $i=1, 2, 3, \dots, c$. $FP(i)$ is the false positive, which refers to the samples number in Z_2 that were incorrectly classified as class i but they belong to the other classes. While, $FN(i)$ is the false negative, which indicates the number of Z_2 samples that were wrongly classified as being from other classes but they belong to class i . As regards to the classification/testing phase, the target is to classify the new sample by assigning a class label to it, and this is done by connecting a new sample $y \in Z_3$ (or Z_4) to all training nodes, as similar to the square shape (test sample) in Fig. 1(d). Then, the distance $d(x, y)$ between the test node y and each training node $x \in Z_1$ is calculated and used to weight the arcs. The test sample will be classified by assigning to it the class label of the training sample that achieved the minimum path-cost with it, as shown in Fig. 1(e). The path-cost $C(y)$ between the samples is computed as follows:

$$C(y) = \min\{\max\{C(x), d(x, y)\}\}, \quad \forall x \in Z_1 \quad (5)$$

It can be supposed that $x^* \in Z_1$ is the training sample that achieved the optimum cost with the test sample y depending on Eq. 5. In this context, $L(x^*) = \lambda(R(y))$, where; $R(y)$ is used to get the root of sample y and the function $\lambda(y)$ is used to assign the correct class label. The classification assigns the class label of x^* as the class label of the test sample y . The classification error happens in the case that $L(x^*) \neq \lambda(y)$. The same procedure of the classification is applied to the learning samples in Z_2 . However, some samples of Z_2 that were misclassified are used to teach the classifier and improve its classification performance.

OPF is explained in the following pseudo-code of Algorithm 1 that is applied to measure the solution fitness function and also when the solution is mutated to measure its new fitness function.

The algorithm receives the training and evaluating sets as input data to learn through them along with the iterations of the loop (1-10), where the classifier is trained over Z_1 and then evaluated over Z_2 . The last best accuracy ($bestAcc$) is compared with the obtained accuracy over Z_2 (acc) and in case that the latter (acc) is higher than the last $bestAcc$, the best instance classifier is updated with the current classifier; otherwise, the last best is kept. Then, the misclassified samples of Z_2 are selected and exchanged with random non-prototypes samples from Z_1 . These steps continue until the stopping condition is met. By following these steps, the classifier can increase its classification quality by learning from its classification errors. Finally, the accuracy of the best classifier instance is returned and used as the solution fitness function.

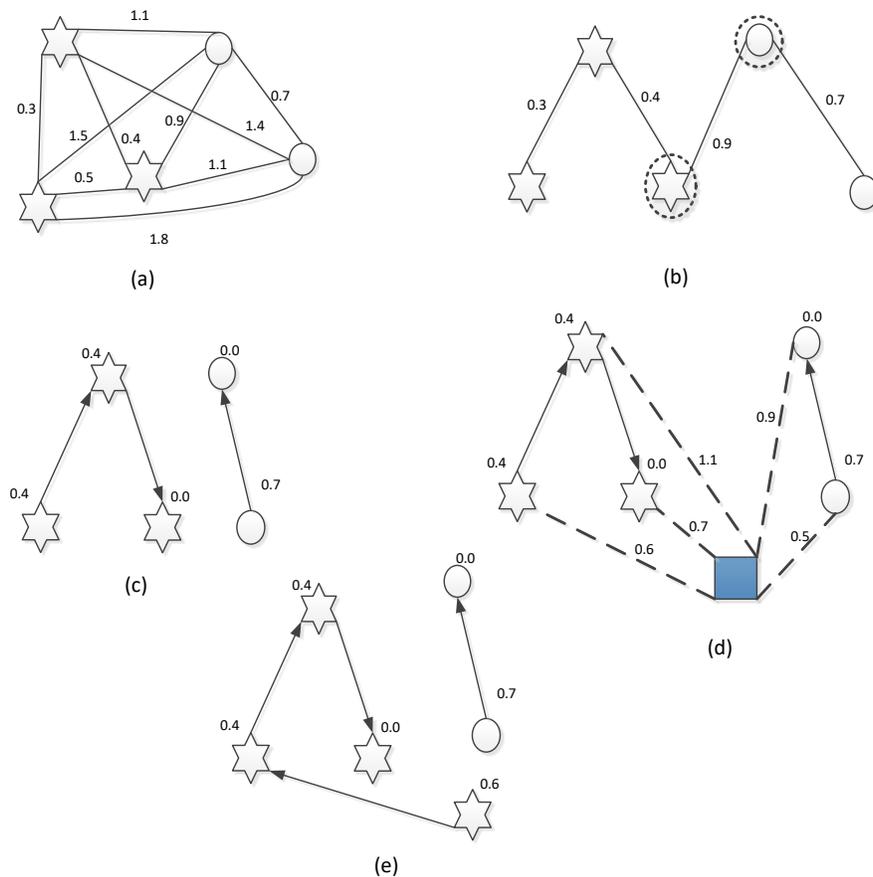


Fig. 1. A Complete Weighted Graph is Constructed by Samples with different Class Labels (Stars and Circles) of the Training Set, (b) Minimum Spanning Tree (MST) is Applied and the Prototype Samples Marked with a Dashed Circle, (c) Optimum-Path Trees (OPTs) are Generated and Rooted with Prototype Samples at the Final of the Training Phase, (d) Test Sample (Square Shape) is Connected to all the Training nodes as Expressed with Dashed Lines to Calculate the Path-Costs (e) Test Sample is Classified with the Label of the Sample that Offered the Minimum Path-Cost, so its Label is the Star Class. Note: The Values above the nodes are their Costs after Training, and the Values above the Arcs Denote the Distance between their Corresponding Nodes.

Algorithm 1: OPF Classifier Pseudocode

Input: Training set Z_1 , evaluating set Z_2 and number of iterations N .

Output: The best classifier instance I .

1. **While** ($n < N$) **do**
2. OPF is trained over Z_1 ;
3. OPF is evaluated over Z_2 ;
4. The accuracy of the classifier instance I is calculated using Eq. 2 and stores it in acc variable;
5. **If** ($acc > bestAcc$) **then**
6. The current classifier instance is updated to be the best instance;
7. $bestAcc = acc$;
8. **End if**
9. The misclassified samples of Z_2 are exchanged with random non-prototypes samples from Z_1 ;
10. **End while**
11. **Return** the best classifier instance where its accuracy represents the fitness function of the solution;

Algorithm 2: BCSA for Feature Selection

Input: Training set (Z_1), Evaluating set (Z_2), Population size (p), Number of features (f) and number of iterations (T).

Output: The best solution with the selected features that achieved the best fitness value over Z_2 .

1. **For** each solution s ($\forall s=1,2,\dots,p$) **do**
2. **For** each dimension d ($\forall d=1,2,\dots,f$) **do**
3. $x_s^d(0) \leftarrow \text{Random}\{0,1\}$;
4. **End For**
5. $F_s \leftarrow -\infty$;
6. **End For**
7. **For** each solution s ($\forall s=1,2,\dots,p$) **do**
8. Compose the new training set Z_1 and evaluating set Z_2 from the original sets Z_1 and Z_2 respectively, with the selected features $x_s^d(t)=1, \forall d=1,2,\dots,f$;
9. $F_s \leftarrow$ Calculate the solution fitness function by applying Algorithm 1 with the new sets ;
10. **End For**
11. **While** ($t < T$) **do**
12. Select the best solutions;
13. Proliferate (clone) the selected solutions according to their affinity (fitness function);
14. **For** $c = 1: n$ ($n \rightarrow$ all solutions in the clones population) **do**
15. Apply a random walk on each solution (mutation) in clones population to compose matured population;
16. The fitness function of the solution is measured by applying Algorithm 1 with the new sets of mutation ;
17. **End for**
18. The highest solutions are chosen from the original and the matured populations to compose the memory set for a next-generation ;
19. Replace solutions by novel ones (diversity introduction) ;
20. **End while**
21. **Return** the highest solution that fulfilled the best accuracy (fitness function) and its selected features will be used in testing OPF classifier (Testing phase) ;

IV. THE PROPOSED BINARY CLONAL SELECTION ALGORITHM (BCSA)

In the original procedure of CSA, the solutions are updated to continuous positions, while in BCSA, the search domain is modeled as an n -dimensional, where n denotes the features number. The algorithm represents each solution (individual) in the population as a string of binary in which 1 refers that the feature will be chosen to construct a new dataset with selected features but 0 otherwise, which means that each solution encodes subset of features. The affinity function of the solution is calculated by measuring the accuracy of the OPF classifier in Algorithm 1 where each solution may have a different subset of features, so the training and evaluating subsets may be different among the solutions. The mutation process is done on each solution by applying a random walk like the distribution of Lévy flights as follows:

$$x_i^j(t+1) = x_i^j(t) + L(\lambda) \quad (6)$$

Where

$$L(\lambda) = \frac{\lambda \Gamma(\lambda) \cdot \sin(\lambda)}{\pi} \cdot \frac{1}{s^{1+\lambda}}, \quad s > 0 \quad (7)$$

In which $x_i^j(t)$ represents the solution i with the j^{th} feature vector, where $i=1,2,\dots,n$ and $j=1,2,\dots,d$, at the iteration t , $L(\lambda)$ is the Lévy flight step size and $\Gamma(\lambda)$ refers to the gamma function. Each mutated solution is converted to a binary vector by employing Eq. 8 which can provide only binary values:

$$x_i^j(t+1) = \begin{cases} 1, & \text{if } S(x_i^j(t)) > \sigma \\ 0, & \text{Otherwise} \end{cases} \quad (8)$$

Where

$$S(x_i^j(t)) = \frac{1}{1 + e^{-x_i^j(t)}} \quad (9)$$

Where $x_i^j(t)$ is the mutated solution i with its feature vector j^{th} and $\sigma \sim U(0,1)$.

The proposed algorithm deals with the selected features problem as an optimization problem where it searches for the best solution (antibody) with the best features subset that achieves the highest accuracy of OPF classifier (antigen). The pseudo-code of the proposed algorithm BCSA is explained in detail in Algorithm 2.

The algorithm initializes a population of solutions through the loop in Lines 1-6. The position of each solution is

initialized by a vector of random binary values (Lines 2-4). Then, the loop in Lines 7-10 constructs the new training set Z_1 and evaluating set Z_2 with the selected features. After that, Algorithm 1 is applied to the newly constructed training and evaluating sets to measure the OPF accuracy to be the solution fitness function f_s . The main functionality of the proposed algorithm is explained through the iterations in Lines 11-20, where the highest solutions n are selected and then proliferated (cloned) according to their affinities where the cloning is proportional to the affinity, as explained through Lines 12-13. Moreover, lines 14-17 contain a loop on the clones' population, in which the mutation process (Eq. 6) is applied for each solution and this mutation is restricted by Eq. (8) for the binary values. The fitness function of each mutated solution is calculated by applying OPF classifier Algorithm 1 with the new training and evaluating sets. In line 18, the improved solutions are chosen from the original and maturated populations to produce the new population for the next iteration. In line 19, the lower affinity solutions are replaced by the newly generated random solutions to increase the diversity in the population. The algorithm ends in line 21 where the solution that got the best OPF accuracy overall runtime iterations is returned and its selected features will be used in the testing phase.

V. RESULTS AND DISCUSSIONS

The proposed techniques were developed by Java language through a PC Intel Core i5 with 8GB RAM and Windows 7 operating system. The procedure of the experiment run each algorithm 10 times to get the average values, each technique in the experiment used 20 solutions for the population size, and the internal iteration number for each technique was 1000. The results of BCSA were compared to the results of BFPA, BCS, BBA, and BDEA. Table I presents the parameters of the employed optimizers.

The experiments were executed using UCI public datasets which are called the Australian dataset, Breast Cancer dataset, and German Number dataset. Table II illustrates the details that are related to the used datasets. In the current study, the datasets were randomly partitioned into four disjoint subsets Z_1 , Z_2 , Z_3 , and Z_4 . Z_1 is the training set that was used in the experiment with a percentage of 30% to initialize and train the classifier. While Z_2 is the learning set with a percentage of 20%. This learning has a serious impact on improving the composition of samples of Z_1 . However, Z_3 is the validating set that was used with a percentage of 20% to ensure the efficiency of the subset of features selected. Z_4 is the testing set that was used with a percentage of 30% for finally calculating OPF accuracy with the features selected.

The methodology of the experiment depends on the threshold approach that was presented in [15]. This approach divides the running times into values range from 10% to 90%, for each period of the running time, the best solution that got the highest fitness function over Z_2 was stored in a vector. Then, the features subset of the stored solutions is used to test the validation set Z_3 . After that the best-stored features subset

that maximized the accuracy over Z_3 will be used to evaluate the testing set Z_4 . The purpose of the validation step is to guarantee the best-selected features and ensure the quality of them before their using in the test step over Z_4 , as explained in Fig. 2 [17].

Throughout the remaining of the research paper, the bold format represents the best values. Table III displays the average results of classification accuracy (fitness function) of the compared techniques over testing sets of all datasets. The obtained results revealed that BCSA surpassed the other four algorithms in all datasets.

Table IV shows the calculated standard deviation of the classification accuracy results of the compared techniques. It was demonstrated that BCSA had better results than BBA, BFPA, and BDEA in the Breast Cancer dataset. Besides, it was evident that the standard deviation of BCSA was higher than those of BFPA and BDEA in the German Numeric dataset.

The average classification errors of the different algorithms are represented in Table V. It was remarked that BCSA achieved the least classification error compared with other algorithms in the three datasets. Wilcoxon rank test [34] was calculated between the proposed algorithm BCSA and the compared techniques. The results in Table VI indicate that the BCSA outperforms the BBA over the Australian dataset (0.022) and Breast Cancer dataset (0.007) and also, surpasses BDEA over the Breast Cancer dataset (0.047), taking into account that significance level is $\alpha = 0.05$. The results in Table VII are the average of features selected by the compared algorithms over the used datasets. It is remarked that although BCSA selected the smallest number of features compared with the other techniques, it achieved the highest accuracy as shown in Table III and Fig. 3, also it achieved the lowest classification error as outlined in Table V. Table VIII shows the execution time of the used algorithms. The obtained results revealed that BCSA was executed in less time over the Breast Cancer dataset and German Numeric dataset, also it had the best mean execution time over the three datasets. The experimental results proved that BCSA outperformed the compared techniques where the best results were obtained through the classification accuracy and the number of the features selected.

The special characteristics of the clonal selection are the reason behind the exceeding of BCSA over the compared algorithms. An adaptive cloning technique, so that the high-affinity solutions are cloned by a low cloning rate, and the low-affinity solutions are cloned by a high cloning rate, this step enhances exploitation. In order not to locate in local optima, the worst affinity individuals are exchanged by randomly newly generated individuals, therefore, the algorithm always maintains the population diversity that is very important for exploration property. Also, the receptor editing helps to achieve population diversity, exploring new search regions, and avoiding local optima.

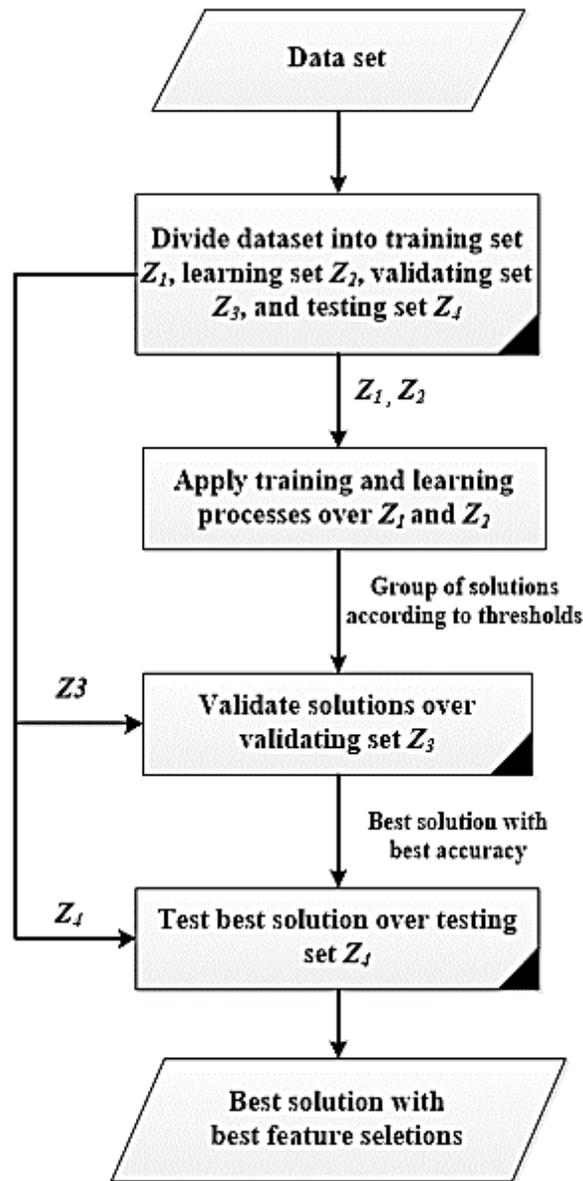


Fig. 2. Flowchart of the used Methodology in Our Experiments.

TABLE I. THE PARAMETERS OF THE EMPLOYED OPTIMIZERS

Algorithm	Parameters
BCSA	Clone population size= 10, Diversity percentage= 25 %, $\lambda = 1.5$
BFPA	$\lambda = 1.5$, $P=0.8$
BCS	$\lambda = 1.5$, $\alpha=0.1$, $Pa=0.25$
BBA	$f_{\min}=0$, $f_{\max}=2$, $A=0.5$, $\alpha=0.9$, $r=0.5$, $\gamma=0.9$
BDEA	$Cr=0.5$, $B=0.5$

TABLE II. ATTRIBUTES OF THE USED DATASETS

Dataset Name	Number of Samples	Number of Features	Number of Classes
Australian	690	14	2
Breast Cancer	699	10	2
German Numeric	1000	24	2

TABLE III. THE CLASSIFICATION ACCURACY OF THE COMPARED ALGORITHMS FOR THREE DATASETS

Dataset Name	BCSA	BFPA	BCS	BBA	BDEA
Australian	77.562%	72.554%	77.315%	71.098%	74.551%
Breast Cancer	96.262%	95.077%	95.132%	89.175%	95.292%
German Numeric	58.543%	56.314%	57.094%	55.905%	56.325%
Mean	77.456%	74.649%	76.514%	72.059%	75.389%

TABLE IV. THE STANDARD DEVIATIONS OF THE CLASSIFICATION ACCURACY OF THE COMPARED ALGORITHMS FOR ALL DATASETS

Dataset Name	BCSA	BFPA	BCS	BBA	BDEA
Australian	7.597	6.212	5.773	4.178	7.040
Breast Cancer	1.171	1.960	0.997	7.927	1.536
German Numeric	3.959	4.889	3.823	3.366	4.042
Mean	4.243	4.354	3.531	5.157	4.206

TABLE V. THE RESULTS OF CLASSIFICATION ERRORS THAT WERE OBTAINED FROM THE DIFFERENT ALGORITHMS OVER TESTING SETS FOR ALL DATASETS

Dataset Name	BCSA	BFPA	BCS	BBA	BDEA
Australian	22.437%	27.445%	22.685%	28.902%	25.449%
Breast Cancer	3.738%	4.923%	4.868%	10.825%	4.708%
German Numeric	41.456%	43.685%	42.906%	44.095%	43.675%
Mean	22.544%	25.351%	23.486%	27.941%	24.611%

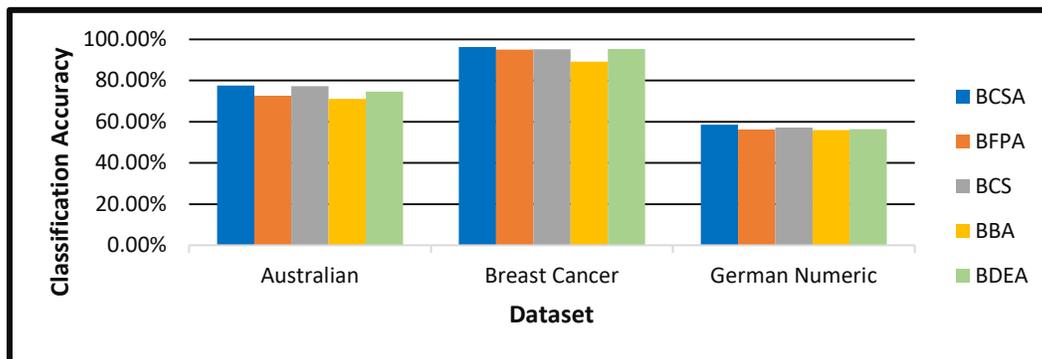


Fig. 3. The Classification Accuracy of the Compared Algorithms for All Datasets.

TABLE VI. P-VALUES OF THE WILCOXON TEST FOR THE BCSA AGAINST THE COMPARED ALGORITHMS FOR ALL DATASETS

Dataset Name	BFPA	BCS	BBA	BDEA
Australian	0.169	0.721	0.022	0.285
Breast Cancer	0.114	0.074	0.007	0.047
German Numeric	0.333	0.386	0.333	0.203

TABLE VII. THE AVERAGE SELECTED FEATURES OF THE COMPARED ALGORITHMS FOR ALL DATASETS

Dataset Name	BCSA	BFPA	BCS	BBA	BDEA
Australian	6.7	10.2	9.4	9.4	10.4
Breast Cancer	6.1	7.5	7.5	7.5	8.2
German Numeric	15.5	16	16	15.5	16.8
Mean	9.4	11.2	10.9	10.8	11.8

TABLE VIII. THE EXECUTION TIME OF THE COMPARED ALGORITHMS IN SECONDS FOR ALL DATASETS

Dataset Name	BCSA	BFPA	BCS	BBA	BDEA
Australian	4909.579	4826.419	3858.050	6137.972	3939.559
Breast Cancer	3679.145	7628.455	4913.337	4917.187	7612.023
German Numeric	16373.967	21636.725	21663.917	27059.006	26715.927
Mean	8320.898	11363.866	10145.101	12704.722	12755.836

VI. CONCLUSION

From the current research, it could be concluded that the suggested Binary Clonal Selection Algorithm (BCSA) has an ability to solve optimization problems such as feature selection problem and get notable results against four powerful techniques. The target of this problem is to look for the most informative subset of features that represent all features in a specific domain. The proposed BCSA surpassed famous techniques like BFPA, BCS, BBA, and BDEA and got the best results through the accuracy of classification, the number of features selected, and the execution time.

Therefore, it is suggested that BCSA is tested against many public datasets and real-world problems. It is proposed to be used with different classifiers like Support Vector Machine (SVM), k-Nearest Neighbors (k-NN) and Artificial Neural Networks (ANN) in order to assure its reliability. Moreover, we intend to apply it to big data mining and to solve other problems like feature weighting, job scheduling, and text processing.

Additionally, there are some ideas related to the Clonal Selection Algorithm where the ratio of the mutation can be adapted according to the individual affinity and the number of iterations. If we assumed that the population converges by times, so the mutation can have a large value at first and then decrease with time. The same concept can be applied to the ratio of cloning but in the reverse order, the ratio of the cloning can start with a small value and increase with time.

REFERENCES

- [1] De Castro, L.N., Von Zuben, F.J.: The clonal selection algorithm with engineering applications. In: Proceedings of GECCO. pp. 36–39 (2000).
- [2] Yang, Xin-She: Nature-inspired optimization algorithms: Challenges and open problems, *Journal of Computational Science*, (2020).
- [3] Inbarani, H.H., Azar, A.T., Jothi, G.: Supervised hybrid feature selection based on PSO and rough sets for medical diagnosis. *Comput. Methods Programs Biomed.* 113, 175–185 (2014).
- [4] Kashaf, S., Nezamabadi-pour, H.: A new feature selection algorithm based on binary ant colony optimization, (2013).
- [5] Mafarja, M., Eleyan, D.: Ant colony optimization based feature selection in rough set theory. *Int J Comput Sci Electron Eng.* 1, 244–247 (2013).
- [6] Papa, J.P., Pagnin, A., Schellini, S.A., Spadotto, A., Guido, R.C., Ponti, M., Chiachia, G., Falcão, A.X.: Feature selection through gravitational search algorithm. In: Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on. pp. 2052–2055 (2011).
- [7] He, X., Zhang, Q., Sun, N., Dong, Y.: Feature selection with discrete binary differential evolution. In: Artificial Intelligence and Computational Intelligence, 2009. AICI'09. International Conference on. pp. 327–330 (2009).
- [8] Ding, S., Li, S.: Clonal selection algorithm for feature selection and parameters optimization of support vector machines. In: Knowledge Acquisition and Modeling, 2009. KAM'09. Second International Symposium on. pp. 17–20 (2009).
- [9] Zhao, M., FU, C., Ji, L., Tang, K., Zhou, M.: Clonal Selection Algorithm Based on Feature Antibodies for Feature Selection and Parameter Optimization of Support Vector Machines. *J. Comput. Inf. Syst.* 7, 1163–1172 (2011).
- [10] Zhang, M., Shao, C., Li, F., Gan, Y., Sun, J.: Evolving neural network classifiers and feature subset using artificial fish swarm. In: Mechatronics and Automation, Proceedings of the 2006 IEEE International Conference on. pp. 1598–1602 (2006).
- [11] Wang, G., Dai, D.: Network intrusion detection based on the improved artificial fish swarm algorithm. *J. Comput.* 8, 2990–2996 (2013).
- [12] Ramos, C.C.O., Souza, A.N., Chiachia, G., Falcão, A.X., Papa, J.P.: A novel algorithm for feature selection using harmony search and its application for non-technical losses detection. *Comput. Electr. Eng.* 37, 886–894 (2011).
- [13] Banati, H., Bajaj, M.: Firefly based feature selection approach. *IJCSI Int. J. Comput. Sci. Issues.* 8, (2011).
- [14] Rodrigues, D., Pereira, L.A.M., Almeida, T.N.S., Papa, J.P., Souza, A.N., Ramos, C.C.O., Yang, X.-S.: BCS: A Binary Cuckoo Search algorithm for feature selection. In: Circuits and Systems (ISCAS), 2013 IEEE International Symposium on. pp. 465–468 (2013).
- [15] Nakamura, R.Y.M., Pereira, L.A.M., Costa, K.A., Rodrigues, D., Papa, J.P., Yang, X.-S.: BBA: a binary bat algorithm for feature selection. In: Graphics, Patterns and Images (SIBGRAPI), 2012 25th SIBGRAPI Conference on. pp. 291–297 (2012).
- [16] Yang, X.-S.: Recent Advances in Swarm Intelligence and Evolutionary Computation. Springer (2015).
- [17] Sayed, S.A.-F., Nabil, E., Badr, A.: A Binary Clonal Flower Pollination for Feature Selection. *Pattern Recognit. Lett.* 77, 21–27 (2016).
- [18] Liu, X., Shi, L., Chen, R., Chen, H.: A novel clonal selection algorithm for global optimization problems. In: Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference on. pp. 1–4 (2009).
- [19] Li, J., Gao, H., Wang, S.: A novel clone selection algorithm with reconfigurable search space ability and its application. In: Natural Computation, 2008. ICNC'08. Fourth International Conference on. pp. 612–616 (2008).
- [20] Lu, H., Yang, J.: An improved clonal selection algorithm for job shop scheduling. In: Intelligent Ubiquitous Computing and Education, 2009 International Symposium on. pp. 34–37 (2009).
- [21] Ulutas, B.H., Islier, A.A.: A clonal selection algorithm for dynamic facility layout problems. *J. Manuf. Syst.* 28, 123–131 (2009).
- [22] Papa, J.P., Falcao, A.X., Suzuki, C.T.N.: Supervised pattern classification based on optimum-path forest. *Int. J. Imaging Syst. Technol.* 19, 120–131 (2009).
- [23] Papa, J.P., Falcão, A.X., De Albuquerque, V.H.C., Tavares, J.M.R.S.: Efficient supervised optimum-path forest classification for large datasets. *Pattern Recognit.* 45, 512–520 (2012).
- [24] Blake, C., Merz, C.J.: UCI Repository of machine learning databases. (1998).
- [25] Timmis, J., Neal, M., Hunt, J.: An artificial immune system for data analysis. *Biosystems.* 55, 143–150 (2000).
- [26] Edelman, G.M., Gally, J.A.: Degeneracy and complexity in biological systems. *Proc. Natl. Acad. Sci.* 98, 13763–13768 (2001).
- [27] Forrest, S., Perelson, A.S., Allen, L., Cherkuri, R.: Self-nonself discrimination in a computer. In: null. p. 202 (1994).
- [28] Gonzalez, F., Dasgupta, D., Niño, L.F.: A randomized real-valued negative selection algorithm. In: Artificial Immune Systems. pp. 261–272. Springer (2003).
- [29] Dasgupta, D., Forrest, S.: Tool breakage detection in milling operations using a negative-selection algorithm. (1995).
- [30] Rezvanian, A., Meybodi, M.R.: LACAIS: learning automata based cooperative artificial immune system for function optimization. In: Contemporary Computing. pp. 64–75. Springer (2010).
- [31] Dasgupta, D., Yu, S., Nino, F.: Recent advances in artificial immune systems: models and applications. *Appl. Soft Comput.* 11, 1574–1587 (2011).
- [32] Burnet, F.M.: Clonal selection and after. *Theor. Immunol.* 63, 85 (1978)
- [33] Bernardino, H.S., Barbosa, H.J.C., Fonseca, L.G.: A faster clonal selection algorithm for expensive optimization problems. In: Artificial Immune Systems. pp. 130–143. Springer (2010).
- [34] Wilcoxon, F.: Individual comparisons by ranking methods. *Biometrics Bull.* 1, 80–83 (1945).