Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 3, No. 3, December 2017.

A BINARY CLASSIFIER BASED ON FIREFLY ALGORITHM

Raed Z. Al-Abdallah[1], Ameera S. Jaradat[1], Iyad Abu Doush[2] and Yazan A. Jaradat[1]

(Received: 27-Jul.-2017, Revised: 23-Sep.-2017, Accepted: 11-Oct.-2017)

ABSTRACT

This work implements the Firefly algorithm (FA) to find the best decision hyper-plane in the feature space. The proposed classifier uses a cross-validation of a 10-fold portioning for the training and the testing phases used for classification. Five pattern recognition binary benchmark problems with different feature vector dimensions are used to demonstrate the effectiveness of the proposed classifier. We compare the FA classifier results with those of other approaches through two experiments. The experimental results indicated that FA classifier is a competitive classification technique. The FA shows better results in three out of the four tested datasets used in the second experiment.

KEYWORDS

Swarm-based algorithms, Binary classification problems, Firefly algorithms.

1. INTRODUCTION

Classification means using the characteristics of an object to identify to which set of predefined classes it belongs. The classification problem has many applications, such as: medical diagnosis, news filtering, document retrieval, opinion mining, email classification and spam filtering. Binary classification is the problem of classifying a new input instance to be in one of two classes. For example, a received email is classified into either a spam or a non-spam. Another example of binary classification is when a patient is diagnosed either to be infected or not infected with a specific disease [1].

A classification technique can use a training dataset to learn how it can classify new instances. The training dataset consists of a set of training examples. Each example is a pair of an input vector of features and the desired output class value. The classification has two phases; a learning phase and a testing phase. In the learning phase, the category of an instance is identified according to its closeness of instances in the training data. A classification model is usually generated by identifying the feature values in the training data instance to one of the predefined class labels. In the testing phase, this classification model is used to explicitly select a particular class for the new instance data.

A classification algorithm can be used to identify a mathematical model to classify a new instance. A linear classification is a type of classification that uses a polynomial function of degree one to classify the new data.This linear degree function is a hyper-plane. In general, a hyper-plane of n-1 dimension is the separator in the n-dimensional space. For example, in the 3-dimensional space, the hyper-plane becomes the 2-dimensional plane. In the 2-dimensional space, the hyper-plane becomes the1-dimensional line. This hyper-plane in binary classification is used to separate the data samples in two different places in the feature space. Equation (1) below shows the general hyper-plane with n dimensions [2]:

$$w_1x_1 + w_2x_2 + \cdots + w_nx_n + w_{n+1}x_{n+1} = 0\,; \tag{1}$$

where $\mathrm{w_j} = (\mathrm{w_1}, \mathrm{w_2}, \dots, \mathrm{w_n}, \mathrm{w_{n+1}})$ is the weight vector and $\mathrm{x} = (\mathrm{x_1}, \mathrm{x_2}, \dots, \mathrm{x_n}, \mathrm{x_{n+1}})$ is the feature vector. Note that n is the dimensions of the hyper-plane and it represents the number of features. In the linear classification, we use a hyper-plane to split the data between two classes.The points in the feature space of n dimensions which are located above the hyper-plane belong to one class and the

1. R. Z. Al-Abdallah, A. S. Jaradat and Y. A. Jaradat are with the Department of Computer Science,Yarmouk University,Irbid, Jordan.
2. I. Abu Doush is with the Department of Computer Science and Information Systems, American University of Kuwait, Salmiya, Kuwait.

other points that are located below the hyper-plane belong to the second class. All points in the feature space of n dimensions that are located in half spaces above the hyper-plane are mathematically greater than zero when their values are substituted in Equation (1). In other words, they are all points that satisfy the inequality (2) to true while all other points that are located in half spaces below the hyper-plane are mathematically lesser than zero when their values are substituted in Equation 1.

$$w_1x_1 + w_2x_2 + \cdots + w_nx_n + w_{n+1}x_{n+1} > 0 \tag{2}$$

A binary linear classifier is formulated mathematically using Equation 3.

$$h(x, w, w_0) = sign(x.w + w_0)\ ; \tag{3}$$

where x is the features vector, w is the weights vector and w_0 is the base of the hyper-plane. Note that *sign* is a function which returns 1 when $x.w + w_0 > 0$ and -1 when $x.w + w_0 < 0$. The (.) is dot vectors multiplication. Algorithm (1) shows a pseudo-code to classify data with two categories using the hyper-plane equation function. Class A represents one class and class B represents the other class in the binary classifier.

Algorithm 1: Linear Classifier

```
1:    $x = \sum_{i=1}^{n} (w_i x_i)$
2:    If x > 0 then
3:    Return Class A
4:    Else If x < 0 then
5:    Return Class B
6:    End if
```

Algorithm 1. Linear classifier.

The algorithm first calculates the instance features values using Equation (1). The obtained result is used to locate the instance on the hyper-plane by comparing it with zero as a threshold.
The main classification methods can be categorized as follows:

- **Decision Trees**: a decision tree is a hierarchical decomposition of training data. A decision tree is made of decision nodes and leaf nodes. Each decision node has a condition over an attribute value. This condition is used to divide the data space into a number of branches. A leaf node represents a class that represents the decision result. The decision tree is used to classify testing data [3].
- **Rule-based Classifiers**: in a rule-based classifier, a set of IF-THEN rules are used for which the left hand side (LHS) is a condition on the feature set and the right hand side (RHS) is the predicted class label. For a given test instance, we determine the set of rules for which the test instance satisfies the condition on the LHS of the rule, then use them to determine the predicted class. Sequential Covering Algorithms (SCA) strategy is the most used strategy to induce rules from the training data. It learns a rule from a training set (conquer step), then removes from the training set the examples covered by the rule (separate step) and recursively learns another rule which covers the remaining examples [4].
- **Support Vector Machine (SVM) Classifiers**: SVM classifiers attempt to partition the data space with the use of linear or non-linear drawing between the different classes. The goal in such classifiers is to find the optimal boundaries between the different classes. In linear SVM, the optimal hyper-plane is the one that minimizes the accuracy error and maximizs the geometric margin. The geometric margin represents the minimum distance of the training samples of both classes from the separating hyper-plane [5].
- **Neural Network Classifiers**: a neural network (NN) classifier consists of units arranged in layers. Each unit takes an input, applies a liner or nonlinear function to it and then passes the output to the next layer. Each node is consisting of a set of input values (xi) and associated weights (wi). These weightings are tuned in the training phase to adapt a neural network in the learning phase [6].
- **Bayesian Classifiers:** Bayesian classifiers are probabilistic classifiers which apply Bayes' theorem. This model is then used to predict the classification of a new instance. The simplest

Bayesian classifier is the naive Bayesian classifier (NBC), which assumes that the input features are conditionally independent of each other [7].

Swarm intelligence algorithms imitate the behaviour of the swarm to obtain the optimal solution for different kinds of problems [1]. It is inspired by the collective behavior of swarms (e.g., ants, bees and a flock of birds). In the swarm, each agent interacts with other agents in a self-organizing behavior. Examples of such algorithms are: particle swarm optimization, firefly algorithm, bat algorithm and ant colony optimization [8], [22] and [32].

One of the recent swarm intelligence algorithms is firefly algorithm (FA). In this paper, FA is used as a binary linear classifier. It is used as a search algorithm to find the best weight vector of the hyper-plane classifier. The proposed algorithm is compared with five of the state of the art algorithms in terms of accuracy. The rest of this paper is organized as follows: Section 2 presents firefly algorithm. Section 3 shows a literature review of some of the classification techniques. The methodology used and the proposed firefly classifier are described Section 4. Section 5 considers the experimental results using five binary datasets. The discussion and a comparison with the state of the art methods are presented in section 6. Finally, we sketch the conclusion and the future work in Section 7.

2. FIREFLY ALGORITHM

FA is inspired by the flashing behaviour in the matting phase of fireflies' life cycle in nature. It is developed by Xin-She Yang at Cambridge University in late 2008 [9]. The fundamental function of flashing light in fireflies is to attract a mate. A male or female firefly light glows brighter in order to make itself more attractive for a mate. The FA algorithm is presented in algorithm (2). FA uses the following three rules [11]:

- A firefly is attracted to other fireflies regardless of their sex, because all fireflies are unisex.
- Attractiveness is proportional to their brightness, thus for any two flashing fireflies, the less bright one will move towards the brighter one. Both attractiveness and brightness are decreasing as the distance between the two fireflies increases. If no one is brighter than a particular firefly, then it moves randomly.
- The brightness or light intensity of a firefly is determined by the objective function of the optimization problem.

```
Algorithm 2: Firefly Algorithm
1:  Initialize parameters α, β, $\gamma$, t=0, Bs=0
2:  $P^{(0)} = InitializeFA()$  // Initialize Randomly Firefly population
3:  While t < Max-Iteration do
4:     FitnessFA($P^{(t)}$)  // calculate fitness value for each solution
5:  Bs = BestFA($P^{(t)}$)   // order population then find best solution
6:  $P^{(t+1)} = MoveFA(P^{(t)})$  // Firefly movement
7:      t= t+1
8:  $\alpha^{(t)} = NewAlpha$ //calculate new alpha value
9:  End While
10: Output Bs
```

Algorithm 2. Firefly algorithm [10].

In Algorithm 2, alpha (α) is the random movement parameter that controls the step length of the random movement, γ is the fixed light absorption coefficient, β is the brightness, t is the iteration number and Bs is the best solution.
InitializeFA() function in line (2) is used for initializing the fireflies' population randomly, where each individual contains two attributes; a position and a fitness. The while-loop (lines 3-9) starts with the *FitnessFA* function in line (3) which is used to calculate the quality of all population solutions. Then, *BestFA* function in line (5) is used to sort the population of fireflies according to their fitness values. After that, the *MoveFA* function in line (6) is used to perform a move of the firefly position (the details are presented in algorithm (3)) [11]. Finally, the *NewAlpha* function in line (8) is used to decrease the initial value of parameter alpha (α) as the iteration increases. The firefly search process is repeated until we reach Max-Iteration steps. After the loop is terminated, the best solution is obtained [10].

```
Algorithm 3: $MoveFA(P^{(t)})$
1:   For i = 1 To n do // n is population size
2:     For j = 1 To n do
3:     $x_i = P_i^{(t)}$.position // array of positions for firefly I at t iteration
4:     $x_j = P_j^{(t)}$. Position
5:     $P_i^{(t+1)} = P_i^{(t)}$
6:     If(f($x_i$)<f($x_j$)) then //f is attractiveness function
7:       $r_{ij} = \sqrt{\sum_{k=1}^{n}(x_{ik} - x_{jk})^2}$
8:       // move firefly i towards j
9:     $x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(x_j - x_i) + \alpha_t(rand - 0.5)$ // rand is a random number
10:      $P_i^{(t+1)}.position = x_i^{t+1}$
11:    End if
12:   End For
13:  End For
14: Return($P^{(t+1)}$)
```

Algorithm 3. Firefly movement.

Algorithm 3 shows the steps for the function *MoveFA*, where line (6) tests the attractiveness (brightness) between two fireflies using the fitness function to determine which firefly is moving and to which one. The details about the fitness function are shown in algorithm (4). The firefly with less brightness will move towards the brighter firefly. The r_{ij} in line (7) is the distance between any two fireflies i and j at $x_i$and x_j positions which is calculated using the Cartesian distance. x_{ik} is the kth component of the spatial coordinate x_i vector of ith firefly and x_{jk} is the kth component of the spatial coordinate x_j vector of jth firefly.

The new position x_i^{t+1} of the moving firefly i at t+1 iteration is calculated by line (9) where the step size of the moving firefly i depends on the last two terms which are added to the current position for firefly i at t iteration. The second term is used to control the step size due to the attraction of a firefly towards the intensity of the light (brightness) by neighboring fireflies. Brightness here is inversely proportional to the distance between the two fireflies due to exponential function characteristics.

The brightness is decreasing as the two firefly distance increases. The third term is a randomization vector of random variables, where α is the random movement parameter that controls the step length of the movement. Note that β_0 is the attraction factor at $r_{ij} = 0$ and γ is the light absorption coefficient. For most cases $\beta_0 = 1$, α Є [0, 1] and γ = 1 [11]. Finally, line 14 returns the new population after the movement phase is completed.

In this paper, we propose a novel FA to be used as a binary linear classifier. The hyper-plane that separates the data into two classes by searching for the best values of a weight vector using Equation (1) is used for the classification decision. The proposed classifier is evaluated on five datasets. Then, it is compared with other classification techniques [12,13]. We use FA for many reasons. Firstly, FA is efficient, because it does not need complex computations and has a limited number of parameters. Moreover, FA is a stochastic meta-heuristic algorithm that can be applied for solving the optimization problems. Thirdly, since firefly algorithm is population-based meta-heuristic, it improves multiple candidate solutions to guide the search [10].

3. Literature Review

There are many classification techniques to generate classifiers: particle swarm classifiers [12], decision tree classifiers [14] and artificial neural network (ANN) classifiers [15]. In this literature review, we present a summary of selected work on using swarm intelligence and ANN for classification.

Sousa et al. [12] proposed the use of Particle Swarm Optimizer (PSO) as a new tool for classification. Three different particle swarm algorithms were implemented and tested against genetic algorithm and tree induction algorithm (J48). The results proved that PSO is competitive when compared with the other techniques.

Zahiri and Seyedin [16] proposed an Intelligent Particle Swarm classifier (IPS classifier) for finding the decision hyper-plane to classify patterns of different classes in the feature space using PSO algorithm. The IPS classifier used an intelligent fuzzy controller which was designed to improve the performance and efficiency of the proposed classifier by adapting three important parameters of PSO (swarm size, neighbourhood size and constriction coefficient). Three pattern recognition problems with different feature vector dimensions were used to demonstrate the effectiveness of the proposed classifier. The experimental results showed that the performance of the IPS-classifier is comparable to or better than the k-nearest neighbour (k-NN) and multi-layer perception (MLP) classifiers. In another work, Martens et al. [17] proposed a new ant-based classification technique named AntMiner+. The key difference between the proposed AntMiner+ and the previous AntMiner versions is the use of a better performing MAX-MIN ant system. Furthermore, AntMiner+ controlled the commonly encountered problem in Ant Colony Optimization (ACO), which is setting the parameters as the new method automatically sets the algorithm parameters. The experiments showed that AntMiner+ accuracy is superior when compared to the other AntMiner versions and that its results are competitive or better than the results achieved by other classification techniques.

Assarzadeh et al. [18] introduced Harmony Search Algorithm-based classifier. The experimental results showed that the performance of the HS-classifier is better than the k-nearest neighbour classifier, particle swarm, genetic algorithm and imperialist competitive algorithm-based classifier. Mantas and Abellán [13] presented a modified version of C4.5, called Credal-C4.5. The modified version of C4.5 used an imprecise probability based on mathematical theory and uncertainty measures. Credal-C4.5 estimated the features probabilities and the class variable using imprecise probabilities. It used imprecise information gain ratio which is a new split criterion. Credal-C4.5 built smaller and better performance trees than the classic C4.5 classifier.

Gandomi et al. [19] introduced FA for solving mixed continuous/discrete structural optimization problems taken from the literature regarding welded beam design, pressure vessel design, helical compression spring design, reinforced concrete beam design, stepped cantilever beam design and car side impact design. The optimization results indicated that FA is more efficient than other meta-heuristic algorithms, such as particle swarm optimization, genetic algorithms, simulated annealing and differential evolution.

Durkota [20] used a modified version of FA to solve the class of discrete problems named Quadratic Assignment Problems (QAP), where the solutions are represented as permutations of integers. In this algorithm, the continuous functions like attractiveness, distance and movement are mapped into newly developed discrete functions. The experimental results were obtained on 11 different QAP problems.

Sayadi et al. [21] proposed a new discrete firefly meta-heuristic for minimizing the make span for the permutation shop scheduling problem. They compared the results of the proposed algorithm with those of other existing ant colony optimization techniques. The results indicated that firefly algorithm outperforms the ant colony for some well-known benchmark problems.

Jati [23] applied FA on the symmetric traveling salesman problem. In this algorithm, a permutation representation is used, where an element of the array represents a city and the index represents the order of a tour. The firefly move is generated using inversion mutation. The simulation results indicated that the proposed algorithm performed very well for some traveling salesman problem instances when compared with other memetic algorithms.

Alweshah [24] proposed a hybrid firefly algorithm with artificial neural network (FA-ANN) for time series problems. The hybrid approach is tested on 6 benchmark UCR time series data sets. The experimental results revealed that the proposed FA-ANN can effectively solve time series classification problems. In another work, Alweshah and Abdullah [25] proposed a method that hybridizes the firefly algorithm with simulated annealing (SFA). They also investigated the effectiveness of using Lévy flight within the firefly algorithm (LFA) to better explore the search space. Moreover, they integrated SFA with Lévy flight (LSFA) to improve the algorithm performance. The algorithm was tested on 11 standard benchmark datasets. The experimental results indicated that the LSFA shows better performance than the SFA and LFA. Moreover, the LSFA is able to obtain better results in terms of classification accuracy when compared with other algorithms from the literature.

Alweshah et al. [26] proposed an improved probabilistic neural network model that employs biogeography-based optimization to enhance the accuracy of classification. Their proposed approach was tested on 11 standard benchmark medical datasets. The results showed that the classification accuracy of the proposed model outperforms that of the traditional probabilistic neural network model.

Faris et al. [27] investigated the efficiency of the Lightning Search Algorithm (LSA) in training Neural Network. The investigated LSA-based trainer was evaluated on 16 popular medical diagnosis problems. The algorithm was compared to BP, LM and 6 other evolutionary trainers. The statistical test conducted proved that the LSA-based trainer is significantly superior in comparison with current algorithms on the majority of datasets.

Aljarah et al. [28] proposed a new training algorithm based on whale optimization algorithm (WOA). A set of 20 datasets with different levels of difficulty have been chosen to test the proposed WOA-based trainer. The obtained results were compared with those of a back-propagation algorithm and six evolutionary algorithms. The results proved that the proposed trainer is able to outperform current algorithms on the majority of datasets.

4. METHODOLOGY

In this section, we give a brief description of the dataset used in evaluating the proposed classifier. After that, we provide details on how the dataset is partitioned to generate the training and testing subsets. Finally, we list the steps to develop the firefly-based classifier. Figure 1 shows the overall research design.

4.1 Datasets

We use the following binary class datasets to test the proposed firefly classifier. The datasets are obtained from the University of California at Irvine (UCI) Machine Learning Repository as follows:

1. Wisconsin Breast Cancer (Original) (WBC) dataset: this dataset is collected from the University of Wisconsin Hospitals between 1989 and 1991. It is commonly used among researchers who use machine learning methods in order to classify patients with breast cancer using a set of attributes. WBC contains 699 instances, 241 instances are malignant class and 458 instances are benign class. Each instance has 9 attributes and each attribute is represented as an integer between 1 and 10.
2. Haberman's Survival dataset: this dataset contains cases from a study that was conducted between 1958 and 1970 at the University of Chicago's Billings Hospital on the survival of patients who had undergone surgery for breast cancer. It contains 306 instances and each instance has 3 attributes. The class of an instance is either survived or died.
3. Statlog (Heart) is a multivariate dataset: it has 13 categorical attributes. It contains 270 instances. Each instance has a class which is either the absence or the presence of heart disease.
4. Liver Disorders dataset: it has 6 attributes. The first 5 variables are all blood tests which are considered to be sensitive to liver disorders. The last attribute is the drinks number of half-pint equivalents of alcoholic beverages drunk per day. The dataset contains 345 instances.
5. Connectionist Bench (Sonar, Mines vs. Rocks) dataset: this dataset contains 208 instances and 111 patterns obtained by bouncing sonar signals off a metal cylinder at various angles and under various conditions. It also contains 97 patterns obtained from rocks under similar conditions.

4.2 Dataset Partitioning Criteria

A ten-fold cross-validation procedure is used to supply the testing and training datasets. The original dataset is partitioned into ten data subsets. Each partition T_i is used as a testing set and the remaining 9 partitions are grouped together to build a training set. Then, we run the FA 30 times. In each time, we

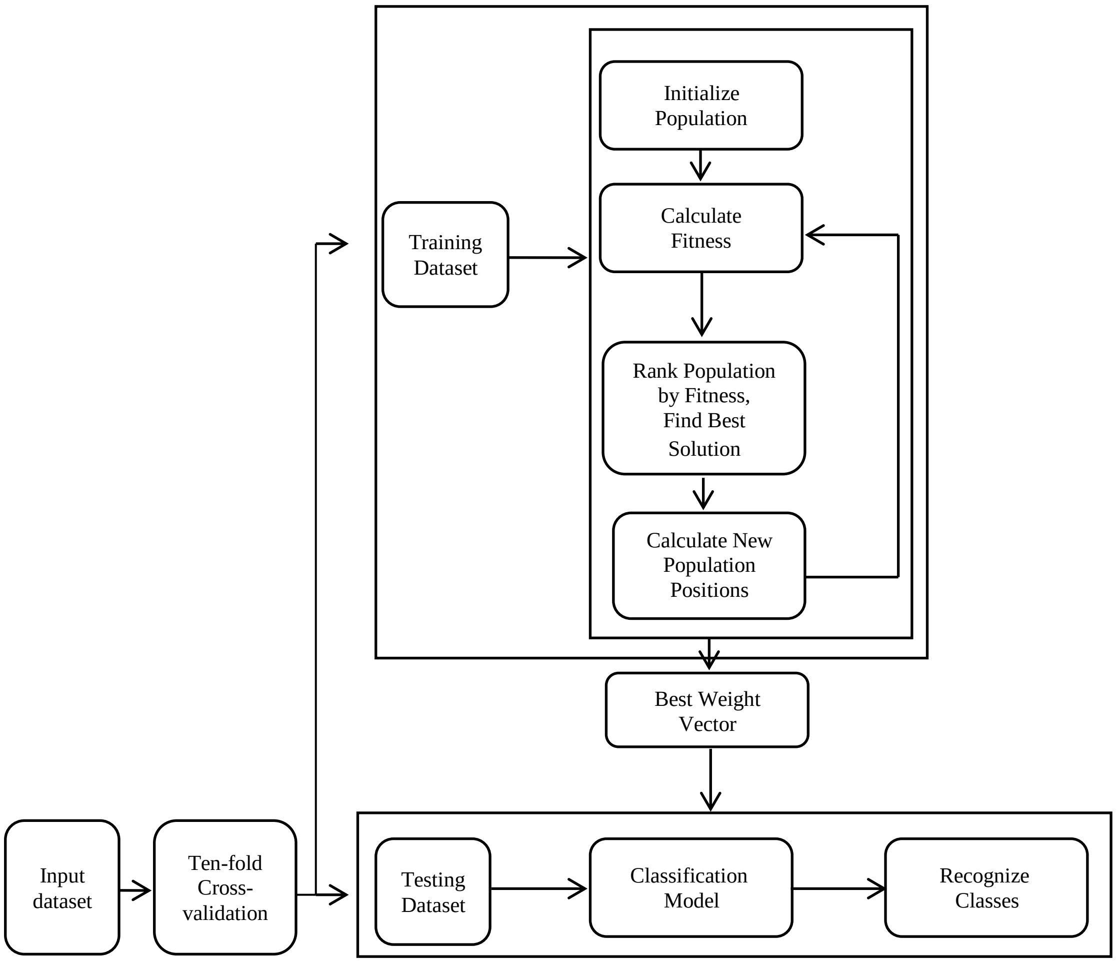

Figure 1. Framework of firefly algorithm classifier.

use T_i as a test set and the union of the others T_{ij} where $i \neq j$ as training set [28]. We use the number of partitions to be 10 to avoid costly computation for higher portioning values. The total number of models calculated in our work is 100 x 10-fold for each run.

4.3 Fitness Function

The objective function is used to score the quality of a solution. Our Fitness Function is the classifier accuracy, which is calculated using Equation (4).

$$Accuracy = \frac{\#\, of\ correctly\ classified\ objects\ in\ test\ set}{\#\, of\ objects\ in\ test\ dataset} * 100 \qquad (4)$$

4.4 Learning Phase

In this phase, we apply firefly algorithm on the training dataset to infer a hyper-plane classifier model by searching for the weights vector that constructs the hyper-plane. This weights vector is the learned knowledge from learning phase and it is reused in the testing phase. The following are the steps:

A. **Initialize Population:** each firefly is a candidate solution, each candidate solution $w = (w_1, w_2, \dots, w_n, w_{n+1})$ is the weights vector. These weights vectors are initialized randomly.

B. **Calculate Fitness:** in this step, we determine the fitness (attractiveness) of each of the fireflies in the population using fitness function. The fitness calculation pseudo-code is presented in algorithm 4, where the position of each firefly is the weight vector for a candidate solution and the fitness is really the accuracy of the hyper-plane that the firefly holds its weights vector.

   In algorithm 4, the $P_L^{(t)}$ position in line (2) is the weights vector for the firefly L at the t iteration. These positions or weights are initialized randomly. In line (4), we substitute both

```
Algorithm 4: FitnessFA($P^{(t)}$)
1:   For L = 1 To n do // n is population size
2:   $w = P_L^{(t)}$.position // position is array of weights initialized randomly of firefly L at t
3:   For k = 1 To n do // n is population size
4:         $y = \sum_{k=1}^{n} (w_k x_k)$
5:   If y > 0 Then
6:           Predict = Class A
7:   Else
8:           Predict = Class B
9:       End if
10:  If Predict == Actual_Class Then
11:          Correct = Correct + 1
12:      End if
13:    End For
14:  $P_L^{(t)}$. Fitness= Correct/n * 100
15   End For
```

Algorithm 4. Calculate fitness.

weight vector of the current firefly and the x features vectors of the training dataset to obtain a value for each instance in the training dataset. Then, we use the obtained value to predict where this instance is located in the space of the hyper-plane by comparing it with zero as a threshold (lines 5-9). We use class A to represent one class and class B to represent the other class. We compare the obtained prediction with the actual one in order to increase correct counter of the prediction (lines 10-12). Then, we use this counter to calculate the fitness for each firefly. The fitness is the accuracy of the hyper-plane.

C. **Rank Population:** in this step, we rank the fireflies according to their fitness value.

D. **Find Current Best Solution**: after ranking the fireflies according to fitness, we return the current best solution which is the weights vector that has the maximum fitness.

E. **Calculate New Population Positions:** in this step, we calculate the new positions of the moving fireflies, where fireflies with the less fitness move towards fireflies with greater fitness (brightness). Equation 5 is used to calculate the new position of the firefly when it performs this move.

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(\mathrm{x_j} - \mathrm{x_i}) + \alpha_t(rand - 0.5)\,; \tag{5}$$

where α is the random movement parameter that controls the step length of the random movement, γ is a fixed light absorption coefficient and β is the brightness that depends on the distance between the two fireflies i and j. The $\mathrm{r_{ij}}$ is the distance between the two fireflies i and j at $\mathrm{x_i}$ and $\mathrm{x_j}$ positions, which is calculated using the Cartesian distance. The new position $\mathrm{x_i^{t+1}}$ of the moving firefly i at t+1 iteration depends on the step size of the moving firefly i and the current position for firefly i at t iteration. The step size of the moving firefly i relies on the attraction of firefly i towards firefly j. The brightness is inversely proportional to the distance between the two fireflies due to exponential function characteristics. The brightness is decreasing as the distance between the two fireflies increases. α is the random movement parameter that controls the step length of the random movement to make firefly i jump not too far away from firefly j. In the Equation, rand is a random number generator which is uniformly distributed between [0, 1]. In order to make the numbers within the range [-0.5, 0.5], 0.5 value is subtracted from rand. The pseudo-code which shows the steps of the new move is presented in algorithm (3).

F. **Checking for Stopping Criteria**: if the stopping criteria are satisfied, return the best weight Vector; otherwise repeat from step B.

4.5 Testing Phase

In this phase, we use the best weight vector learned from the learning phase to construct the binary linear classification model that satisfies Equation (1) to predict the class of instaces in the dataset to be tested using algorithm 1.

5. Parameter Setup and Results

5.1 Parameter Settings

Firefly classifier is implemented using MATLAB 2014a. The experiments are executed on an Intel core i5 processor running with 8 GB of RAM under Microsoft Windows 7. Table 1 presents the parameter setting that we used for the experiments.

Table 1. Firefly parameter setting.

Parameter	value
Termination condition	100
Number of fireflies	100
Attractiveness B_0	2
Light absorption coefficient (Gamma)	1
Randomization parameter (alpha)	0.2
Alpha constant	0.98
Number of times we run the algorithm on each dataset	30

5.2 Performance Metrics

The results obtained by the firefly-based classifier are evaluated using two performance metrics. These are: classification accuracy and k-fold cross-validation (KCV) accuracy. Classification accuracy is calculated using the previous formula (4) [29].

For the k-fold cross-validation (KCV) accuracy evaluation, the original sample is randomly partitioned into k sub-samples. A single sub-sample of the k sub-samples is used as validation data for testing and the remaining k-1 sub-samples are used as training data. The cross-validation process is then repeated k times; each one of the k sub-samples is used exactly once as the validation data. The average of the k results from the k-folds gives the KCV test accuracy of the algorithm [30]-[31] . Our k-fold cross-validation is a 10-fold cross-validation. The proposed algorithm runs 30 times on each dataset using a 10-fold cross-validation. The result from each run is reported and the average of the 30 runs is calculated.

5.3 Experimental Results

We perform two different experiments to evaluate our proposed classifier. The first experiment is applied on Wisconsin Breast Cancer (Original) dataset, while the second experiment is applied on Haberman's Survival, Statlog (Heart), Liver Disorders and Sonar datasets.

The two experiments are applied using a 10-fold and the accuracy of the 30 runs is averaged. Then, these results are compared with those of different algorithms from the state of the art, which are: DPSO, CPSO, LDW PSO, GA, J48, MID3 and CCDT [12]-[13].

In the first experiment, we applied the FA classifier using different population sizes: 25, 50, 100, 200 and 300 on Wisconsin Breast Cancer (WBC) dataset. After that, we calculated the average of the accuracy of the obtained results. Table 2 presents the experimental results of the 10-fold average accuracy. The accuracy of the classifier increases as the population size increases. A population size larger than 100 makes the algorithm exploration not well focused and the algorithm performance is then degraded.

The results from the proposed algorithm when applied on WBC dataset are compared with those of five algorithms presented in [12] which are: DPSO, CPSO, LDW PSO, GA and J48. Table 4 and Figure 2 show the comparison between the proposed classifier accuracy when compared with these techniques.

Table 2. Effect of different population sizes on the performance of FA when applied on WBC dataset.

Population	Accuracy
25	92 ± 1
50	92 ± 2
100	95 ± 3
200	93 ± 2
300	91 ± 2
Average	93

Table 3. FA result for a population size of 100 on WBC dataset.

	Accuracy	Sensitivity	Specificity	Positive Predictive	Negative Predictive
Minimum	94.14	95.01	88.47	93.18	93.3
Maximum	97	97.92	94.25	96.28	97
Average	95.41333	96.489	91.20533	94.55733	95.44033
Standard deviation	0.728282	0.777927	1.413973	0.816693	1.088744

Table 4. Classification accuracy of FA compared with other techniques on WBC dataset.

Population Size	DPSO	CPSO	LDW PSO	GA	J48	FA
25	92 ± 3	92 ± 3	92 ± 5	92 ± 4	93	92 ± 1
50	92 ± 3	92 ± 4	92 ± 6	92 ± 3		92 ± 2
100	92 ± 3	92 ± 4	92 ± 2	92 ± 4		92 ± 3
200	92 ± 5	92 ± 4	92 ± 4	92 ± 3		92 ± 2
300	92 ± 3	92 ± 3	92 ± 4	92 ± 3		92 ± 2
AVG	94	93	93	93		93

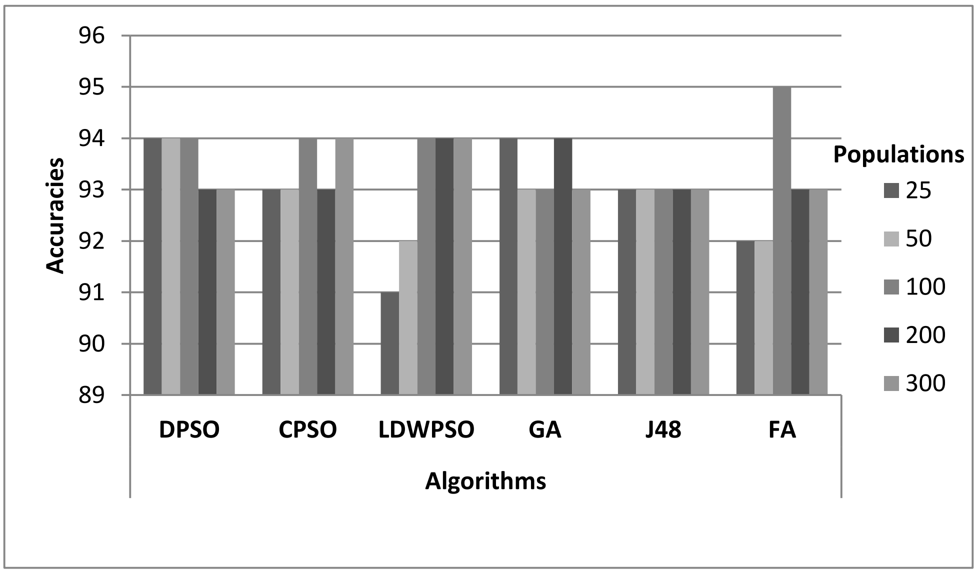

Figure 2. Accuracies of Firefly compared with other techniques on WBC dataset.

According to this experiment, we set the population size of the proposed algorithm to be 100 and the algorithm is repeated 30 times. We calculate the accuracy, sensitivity, specificity, positive predictive and negative predictive obtained from the classifier for each run. The sensitivity of each class is calculated as TP/(TP+FN) and the specificity of each class is calculated as TN/(TN+FP). Note that TP is true positive, FN is false negative, TN is true negative and FP is false positive. The minimum, maximum, average and standard deviation values are reported. Table 3 shows these results.

The second experiment was applied on Haberman's Survival dataset, Statlog (Heart) dataset, Liver

Disorders dataset and connectionist Bench dataset. We compared our results with the results from [13]. Table 4 and figure 3 show the accuracy of the proposed algorithm when compared with [13]. The parameter settings are kept as in the first experiment. We repeat the second experiment using a population size of 100 for 30 runs. We calculate the accuracy, sensitivity, specificity, positive predictive and negative predictive obtained from the classifier for each run. The minimum, maximum, average and standard deviation values are reported. Table 5, Table 6, Table 7 and Table 8 show the results for the Haberman, Heart-tatlog, Liver Disordes and Sonar datasets respectively.

Table 5. FA result for a population size of 100 on Haberman dataset.

	Accuracy	Sensitivity	Specificity	Positive Predictive	Negative Predictive
Minimum	71.55	80.09	26.79	86.7	24.09
Maximum	77.08	83.85	50.12	92.29	43.48
Average	74.9807	82.3093	39.698	89.3153	33.9067
Standard deviation	1.3321	0.9891	5.9915	1.6653	5.6787

Table 6. FA result for a population size of 100 on Heart-tatlog dataset.

	Accuracy	Sensitivity	Specificity	Positive Predictive	Negative Predictive
Minimum	76.3	80.47	66.07	75.7	64.71
Maximum	82.22	86.69	75.74	85.85	81.67
Average	79.6547	83.7443	70.2407	79.465	74.608
Standard deviation	1.5897	1.6157	2.3346	2.646	4.3328

Table 7. FA result for a population size of 100 on Liver Disordes dataset.

	Accuracy	Sensitivity	Specificity	Positive Predictive	Negative Predictive
Minimum	62.95	68.71	60.14	38.34	81.75
Maximum	71.71	77.05	65.23	56.7	89.74
Average	68.2203	72.8753	62.809	48.256	85.7943
Standard deviation	2.530924	2.5029	1.3808	4.7153	2.2573

Table 8. FA result for a population size of 100 on Sonar dataset.

	Accuracy	Sensitivity	Specificity	Positive Predictive	Negative Predictive
Minimum	95.8	95.79	95.95	95.87	96.04
Maximum	98.26	97.87	97.66	98.09	98.15
Average	97.01	96.9933	97.065	97	96.945
Standard deviation	0.639	0.4865	0.4635	0.5010	0.5898

Table 9. Classification accuracies of FA compared to those of different classification algorithms.

Dataset	FA	C4.5	Credal-C4.5	MID3	CCDT
Haberman	**74.98**	70.52	73.89	70.62	73.59
Heart-tatlog	79.65	76.78	80.04	77.93	**82.11**
Liver Disordes	**68.22**	65.37	64.18	65.75	56.85
Sonar	**97.01**	73.42	71.47	73.53	73.92

6. DISCUSSION

We tested the FA algorithm with different population sizes 25,50,100,200 and 300. But, the one that gives the best result is 100, because the algorithm exploration is focused as the population has enough diversity.

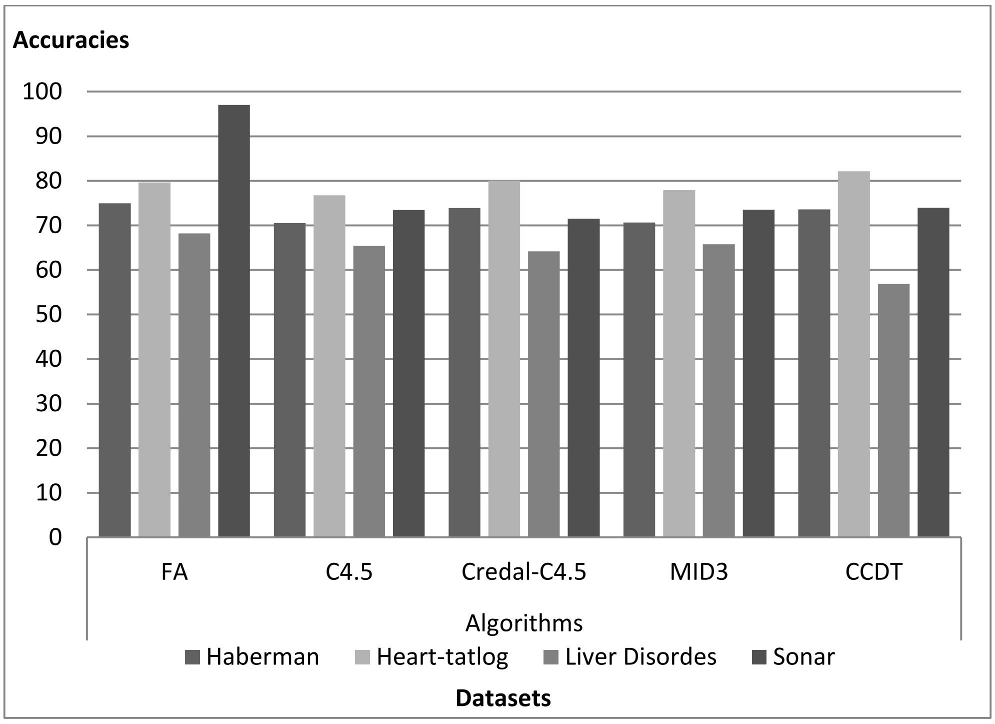

Figure 3. Accuracies of Firefly with a population of 100 compared with those of other techniques on four different datasets.

The experimental results and the accuracy values show that the classification accuracy of the proposed FA when applied on Wisconsin Breast Cancer data using different population sizes reached an average accuracy of 93%, which is similar to the results obtained from CPSO, LDW PSO, GA and J48 algorithms presented in [12]. On the other hand, the DPSO algorithm has a slightly better average accuracy value of 94%. So, our FA results are competitive when compared to those of other algorithms from the state of the art.

Table 9 shows that the classification accuracy of FA is competitive when compared with those of the algorithms presented in [13] when applied to different datasets. The FA showed better results when compared with [13] in three out of the four tested datasets, which are: Haberman, Liver Disordes and Sonar.

The results of applying FA classifier are promising. FA has advantages over other mentioned algorithms, as it speeds up the convergence rate in a small number of iterations. Applying FA leads to achieve good accurate results in an efficient manner. This is because it attracts low-quality solutions to be merged with good solutions, which makes the whole population to be automatically sub-divided into subgroups. Each group is swarmed around its local best and that speeds up the convergence rate. Then, among all these local best solutions, the best solution from the population is quickly found. Another reason that makes FA more efficient is that the parameters in the algorithm can be tuned to control the randomness as the iterations proceed. This makes convergence speed up [10].

7. Conclusion and Future Work

In this work, we presented the Firefly algorithm as a binary linear classifier. The FA optimizes the parameter values for hyper-planes in the feature space. Experimental results show that the Firefly classifier achieves good accuracy when compared with other algorithms. The experimental results and the accuracy values are compared with those of the classifiers CPSO, LDW PSO, GA, J48, C4.5, Credal-C4.5, MID3 and CCDT.

The results prove that the firefly classifier is a competitive classifier. Therefore, the FA approach can be used for other more complex classification problems. Through conducting experiments, we have come to some conclusions regarding the application of the FA. First, increasing the population size improves the accuracy of the firefly classifier, but the algorithm performance is degraded when we reach a population size of more than 100. Second, increasing the number of iterations over 100 in the two experiments has not increased the algorithm accuracy. This proves that firefly classifier achieves very good results with fast convergence. Lastly, the results show that the Firefly classifier is a reasonably good classifier when it is compared with other state of the art classifiers [12]-[13].

As a future work, we suggest analyzing the algorithm when it is applied to a multiclass dataset. Another future work can be studying the effect of tunning different parameters of the FA.

REFERENCES

[1] C. Aggarwal and C. Zhai, "A Survey of Text Classification Algorithms in Mining Text Data," Springer, pp. 163-222, 2012.

[2] W. Clancey, "Heuristic Classification," Artificial Intell. Journal, vol. 27, no. 3, pp. 289-350, 1985.

[3] J. Quinlan, "Induction of Decision Trees," Machine Learning, vol. 1, no. 1, pp. 81-106, 1986.

[4] G. Pappa and A. Freitas, Automating the Design of Data Mining Algorithms, Springer Verlag, Berlin Heidelberg, 2010.

[5] M. Mavroforakis and S. Theodoridis, "A Geometric Approach to Support Vector Machine (SVM) Classification," IEEE Transactions on Neural Networks, vol. 17, no. 3, pp. 671-682, 2006.

[6] A. Ng and M. Jordan, "On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes," Advances in Neural Information Processing Systems, vol. 2, pp. 841-848, 2002.

[7] P. Langley and K. Thompson, "An Analysis of Bayesian Classifiers, " Aaai, 1992.

[8] Jr . Fister, X. Yang, I. Fister, J. Brest and D. Fister, "A Brief Review of Nature-inspired Algorithms for Optimization," arXiv preprint arXiv:1307.4186 , 2013.

[9] X. Yang, "Firefly Algorithms for Multimodal Optimization," International Symposium on Stochastic Algorithms, Stochastic Algorithms: Foundations and Applications (SAGA 2009), Springer, pp. 169-178, 2009.

[10] I. Fister, X. Yang and J. Brest, "A Comprehensive Review of Firefly Algorithms," Swarm and Evolutionary Computation, vol. 13, pp. 34-46, 2013.

[11] X. Yang, "Firefly Algorithm, Stochastic Test Functions and Design Optimization," International Journal of Bio-Inspired Computation,vol. 2, no. 2, pp. 78-84, 2010.

[12] T. Sousa, A. Silva and A. Neves, "Particle Swarm-based Data Mining Algorithms for Classification Task," Parallel Computing, vol. 30, no. 5, pp. 767-783, 2004.

[13] C. Mantas and J. Abellán, "Credal-C4. 5: Decision Tree Based on Imprecise Probabilities to Classify Noisy Data," Expert Systems with Applications, vol. 41, no. 10, pp. 4625-4637, 2014.

[14] N. Bhargava, G. Sharma, R. Bhargava and M. Mathuria,"Decision Tree Analysis on j48 Algorithm for Data Mining," Proceedings of International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, no. 6, pp. 1114-1119, 2013.

[15] F. Partovi and M. Anandarajan, "Classifying Inventory Using an Artificial Neural Network Approach," Computers & Industrial Engineering, vol. 41, no. 4, pp. 389-404, 2002.

[16] S. Zahiri and S. Seyedin, "Intelligent Particle Swarm Classifiers," Iranian Journal of Electrical and Computer Engineering, vol. 4, no. 1, pp. 63-70, 2005.

[17] D. Martens, M. De Backer, R. Haesen, J. Vanthienen, M. Snoeck and B. Baesens, "Classification with Ant Colony Optimization," IEEE Transactions on Evolutionary Computation, vol. 11, no. 5, pp. 651-665, 2007.

[18] Z. Assarzadeh, S. Monadjemi, P. Moallem and S. Hashemi, "Harmony Search-based Classifier," Int. J. Acad. Res. Appl. Sci., vol. 5, no. 1, pp. 108-119, 2015.

[19] A. Gandomi, X. Yang and A. Alavi, "Mixed Variable Structural Optimization Using Firefly Algorithm," Computers & Structures, vol. 89, no. 23, pp. 2325-2336, 2011.

[20] K. Durkota, Implementation of a Discrete Firefly Algorithm for the QAP Problem within the Seage Framework, BSc Tthesis, Faculty of Electrical Engineering, Czech Technical University, 2011.

[21] M. Sayadi, R. Ramezanian and N. Ghaffari-Nasab, "A Discrete Firefly Meta-heuristic with Local

Search for Makespan Minimization in Permutation Flow Shop Scheduling Problems," International Journal of Industrial Engineering Computations, vol. 1, no. 1, pp. 1-10, 2010.

[22] R. Sawalha and I. Abu Doush, "Face Recognition Using Harmony Search-based Selected Features," International Journal of Hybrid Information Technology, vol. 5, no. 2, pp. 1-16, 2012.

[23] G. Jati, "Evolutionary Discrete Firefly Algorithm for Travelling Salesman Problem," Springer, pp. 393-403, 2011.

[24] M. Alweshah, "Firefly Algorithm with Artificial Neural Network for Time Series Problems," Research Journal of Applied Sciences, Engineering and Technology, vol. 7, no. 19, pp. 3978-3982, 2014.

[25] M. Alweshah and S. Abdullah, "Hybridizing Firefly Algorithms with a Probabilistic Neural Network for Solving Classification Problems," Applied Soft Computing, vol. 35, pp. 513-524, 2015.

[26] M. Alweshah, A. Hammouri and S. Tedmori, "Biogeography-based Optimization for Data Classification Problems," International Journal of Data Mining, Modelling and Management, vol. 9, no. 2, pp.142-162, 2017.

[27] H. Faris, I. Aljarah, N. Al-Madi and S. Mirjalili, "Optimizing the Learning Process of Feed Forward Neural Networks Using Lightning Search Algorithm," International Journal on Artificial Intelligence Tools, vol. 25, no. 6, 2016.

[28] I. Aljarah, H. Faris and S. Mirjalili, "Optimizing Connection Weights in Neural Networks Using the Whale Optimization Algorithm," Soft Computing, pp.1-15, 2016.

[29] P. Cheewaprakobkit, "Study of Factors Analysis Affecting Academic Achievement of Undergraduate Students in International Program," Proceedings of the International Multi-conference of Engineers and Computer Scientists, 2013.

[30] N. Diamantidis, D. Karlis and E. Giakoumakis, "Unsupervised Stratification of Cross-validation for Accuracy Estimation," Artificial Intelligence Journal, vol. 116, no. 2, pp. 1-16, 2000.

[31] O. Inan, M. Uzer and N. Yılmaz, "A New Hybrid Feature Selection Method Based on Association Rules and PCA for Detection of Breast Cancer," International Journal of Innovative Computing, Information and Control, vol. 9, no. 2, pp. 727-729, 2013.

[32] S. ElMustafa, A. Jaradat, I. Abu Doush and N. Mansour, "Community Detection Using Intelligent Water Drops Optimization Algorithm," International Journal of Reasoning-based Intelligent Systems, vol. 9, no. 1, pp. 52-65, 2017.

ملخص البحث:

توظف هذه الدراسة خوارزمية "يَراعاتٍ" من أجل إيجاد المستوى الفائق للقرار الأفضل في فضاء الخصائص. يستخدم المصنف المقترح تقنية تثبُّتٍ متقاطعٍ مبنية على تقسيم ذي عشرة أضعاف لكل من طوري التدريب و الاختبار المستخدمَين في عملية التصنيف.

وقد جرى استخدام خمس مسائل مرجعية ثنائية لتمييز الأنماط بأبعاد مختلفة لمتجهات الخصائص، وذلك من أجل عرض فاعلية المصنف المقترح. كذلك، تمت مقارنة نتائج مصنِّف خوارزمية "اليراعات" المقترح مع نتائج طرق أخرى، وذلك عبر التجربة. فقد تم إجراء تجربتين لهذا الغرض، وأثبتت نتائجهما أن مصنف خوارزمية "اليراعات" منافس جيد لتقنيات التصنيف الاخرى. وتجدر الإشارة أن المصنف المقترح حقق نتائج أفضل في ثلاثٍ من مجموعات البيانات الأربع المختبرة في التجربة الثانية.