IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 17, 2017, accepted February 6, 2018, date of publication March 2, 2018, date of current version March 16, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2807124

Hierarchical Flexible Beta Fuzzy Design by a Multi-Objective Evolutionary Hybrid Approach

YOSRA JARRAYA, SOUHIR BOUAZIZ, AND ADEL M. ALIMI, (Senior Member, IEEE)
Research Groups in Intelligent Machines, National Engineering School of Sfax, University of Sfax, Sfax 3038, Tunisia

Corresponding author: Yosra Jarraya (yosra.jarraya@ieee.org).

This work was supported by the Ministry of Higher Education and Scientific Research of Tunisia under Grant LR11ES48.

ABSTRACT This paper introduces a novel evolutionary approach for the automated modeling of efficient hierarchical or multilevel fuzzy systems. The proposed approach is formed by two principal learning stages: the first stage consists of using a proposed multi-objective algorithm called the multi-objective extended immune programming algorithm in order to evolve the architecture of the hierarchical fuzzy system. The use of such a step aims to optimize the structure of the hierarchical fuzzy system and to generate in the same time an accurate and an interpretable fuzzy system with a few number of fuzzy rules. In the second stage, the parameters of beta membership function parameters and the consequent parts of rules are tuned by applying the hybrid artificial bee colony algorithm. The proposed hybrid approach interleaves these two learning phases for the architecture learning and the parameter tuning until a near optimum hierarchical fuzzy system is generated. The efficiency of the methodology is evaluated through some well-known benchmark time-series problems, a nonlinear plant identification problem, and some high-dimensional classification data sets. Compared with other existing works, the proposed system proves its superiority in terms of reaching high accuracy, smaller rule-base, and good convergence speed.

INDEX TERMS Beta basis function, hierarchical fuzzy system, hierarchical structure optimization, multi-objective optimization, parameter tuning.

I. INTRODUCTION

Actually, the modelling of fuzzy models is considered as one of the most interesting research subjects allowing the imitation of human reasoning. In fact, fuzzy systems have been applied to different fields, such as classification [1], [2], pattern recognition [3], data mining problems [4], control problems [5], [6] and time series prediction [7], [8], due to their aptitude to handle imprecision and uncertainty.

However, when the dimensionality and the complexity of the systems increase, the total number of fuzzy rules in a standard fuzzy system increases exponentially with the growth of the number of input variables. This problem is better known as ''the curse of dimensionality'' problem, and it has bad effects on the interpretability of the resulted rule base. As an alternative to solve this problem, hierarchical or multilevel fuzzy design was suggested by Raju and Zhou [9] to decrease the rules number from an exponential function of system variables to a linear one. In this case, instead of the use of high-dimensional standard fuzzy system (flat), a number of lower-dimensional sub-fuzzy models are hierarchically linked. This method of hierarchical modeling allows the construction of fuzzy systems which are more interpretable as well as accurate and with good approximation abilities.

Recently, hierarchical fuzzy modeling has been considered as a search problem and an optimization task. Several research works have been presented in the literature to construct or to refine these systems. Joo and Sudkamp [10] proposed an approach that converts a multidimensional fuzzy system to a two-layer multilevel fuzzy system to improve the run-time and to reduce the rules number. In [11], a method was proposed to construct a hierarchical rule base by applying two structure evolutionary techniques which are the Bacterial and the Genetic Programming Algorithms. In [12], a Hierarchical Collaborative Structure (HCS) was proposed by the authors where all sub-fuzzy systems improved the overall accuracy gradually by adding their own contributions. For this approach, three phases of learning were used: the building of the structure, the identification of parameters and the data division among the different levels of the hierarchy. Lin and Lee [13] also suggested an approach based on a genetic algorithm to tune the hierarchical architecture and the parameters of five inputs multilevel fuzzy controller for the

control low-speed problem. Zeng and Keane [14] examined the representation capacity of hierarchical fuzzy systems and proved the universal approximation ability of these systems. In addition, Chung and Duan [15] focused on the incremental and the aggregated type of hierarchical systems. They proposed a ranking method of the input variables according to their importance, and then the most important inputs will be attributed to the first layer of the hierarchy, and the less important inputs will be attributed to the second level, and so on.

It should be noted that, most of these hierarchical fuzzy systems proposals are focused on improving the accuracy using different optimization techniques and machine learning. However, little works have been done to exploit multi-objective evolutionary process and hybrid learning approaches for evolving hierarchical fuzzy systems. In fact, recently, Multi-Objective Evolutionary Algorithms (MOEAs) have become one of the hottest topics in EA research. Moreover, applying hybrid approaches which combine different machine learning algorithms has also become a promising and powerful method for fuzzy system learning.

In the same context, we propose in this work a novel hybrid methodology for the automated design of high-performance Hierarchical Flexible Beta Fuzzy Systems (HFBFSs) in a multi-objective context. For that, a tree-based encoding scheme is used to illustrate the fuzzy system in a hierarchical representation, and the Beta function proposed by Alimi [16]–[18] is used for the modeling of the membership functions (MFs). For the evolutionary process, a Multi-Objective Extended Immune Programming (MOEIP) algorithm is applied as a first phase to optimize the structure of the proposed HFBFS. The aim of this phase is to evolve the HFBFS structure with the purpose of reaching in the same time a good balance of accuracy/interpretability. Next, the Hybrid Artificial Bee Colony (HABC) algorithm [19] is applied as a second phase for the tuning of the free parameters encoded in the best architecture. These parameters are the Beta MF parameters and the consequent parts of fuzzy rules. We interleave the two phases of architecture and parameter tuning until an adequate HFBFS is generated.

The rest of this article is planned as follows: In section II, the proposed Hierarchical Flexible Beta Fuzzy System is detailed. The initialization step using a clustering technique is then introduced in section III. Next, the multi-objective hierarchical structure optimization and the parameter optimization processes are introduced in sections IV and V respectively. And, the global hybrid evolving algorithm is presented in section VI. Next, some experimental results are given in section VII. And finally, in section VIII, we conclude by some conclusion remarks.

II. THE HIERARCHICAL FLEXIBLE BETA FUZZY SYSTEM (HFBFS)

In general, the modeling of hierarchical fuzzy systems is defined by determining the number of hierarchical levels, the sub-fuzzy models number in each level, defining the different rule bases of the hierarchy, the way of arrangement of original inputs in the different levels, and so on.

In this work, the modeling of an accurate and simple multilevel fuzzy system is considered as a search task in architecture and parameter spaces. For the encoding scheme, a tree-based encoding representation is employed to illustrate the proposed hierarchical fuzzy system. The reason for choosing this encoding method is that the tree has a natural flexible hierarchical representation, and by consequence, it can provide more adjustable and modifiable structures when the search progresses. In addition, the choice of the membership function type plays an important role in the improvement of the system's performance. That's why, in this work, we choose to adopt the Beta function [16]–[18] instead of using Triangular or Gaussian membership functions which are usually employed in the specialized literature. The initiative of adopting Beta function for the modeling of fuzzy systems was presented by Alimi and in this case the system is called Beta Fuzzy System (BFS). The use of the Beta function proves more efficiency than the other functions due to its high flexibility and its universal approximation capacity [16]–[18]. Therefore, in this study, the resulted suggested model is named the Hierarchical Flexible Beta Fuzzy System (HFBFS) and it is characterized by this node set N:

$$N = B \cup T = \{BFS_{kj}^{l}/k \in \{2, \ldots, K\}, j \in \{1, \ldots, J\}, l \in 1, \ldots, (L-1)\} \cup \{x_1, \ldots, x_M\} \quad (1)$$

where:

- $BFS_{kj}^{l}(k \in \{2, \ldots, K\}, j \in \{1, \ldots, J\}, l \in \{1, \ldots, (L-1)\})$ defines the non-terminal node set B. It represents a Beta sub-Fuzzy System (BFS) of type Sugeno formed by k number of inputs and one estimated output. K denotes the tree's maximal degree. j defines the index of the corresponding BFS formed by k input variables, J represents the number of times in which the corresponding BFS occurs with k number of inputs. l denotes the index of level, and L defines the levels number (tree's depth);
- $x_1, x_2, \ldots, x_M$ present the inputs forming the terminal node set T;

For the evaluation of the proposed HFBFS, outputs of some BFS_{kj}^{l} are regarded as inputs for other BFS_{kj}^{l}, and the evaluation is done recursively by depth-first method. For simplicity, we illustrate a simple example through Figure 1 to show how the HFBFS can be evaluated. Figure 1 (left) shows a possible tree structural representation (with two levels and four inputs). The corresponding HFBFS is illustrated in Figure 1 (right). For this example, the output of the sub-fuzzy model BFS_{21}^{2} of the second level ($l = 2$) is firstly evaluated. The rule base of this sub-model has the following form:

$$R_i^{l=2}: \text{ If } x_1 \text{ is } A_{1i}^{2} \text{ and } x_2 \text{ is } A_{2i}^{2} \text{ then } y^2 = a_{0i}^{2} + a_{1i}^{2} x_1 + a_{2i}^{2} x_2$$

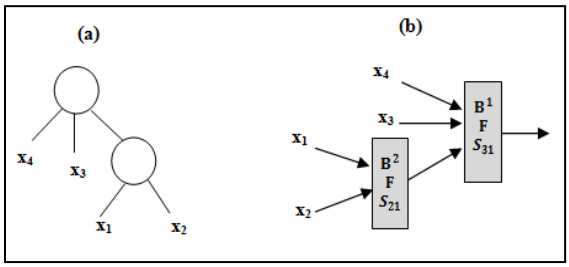

FIGURE 1. (a) A simple example of a tree structural representation, (b) The corresponding HFBFS: $B = \{BFS^1_{31}, BFS^2_{21}\}$, $T = \{x_1, x_2, x_3, x_4\}$.

where x_1 and x_2 are the input variables and y^2 is the output of the sub-fuzzy model BFS^2_{21}. A^2_{1i} and A^2_{2i} are the Beta fuzzy sets of variables x_1 and x_2 respectively. a^2_{0i}, a^2_{1i} and a^2_{2i} are the crisp consequent part parameters.

Next, the overall output of the HFBFS is computed by evaluating the output of the first level sub-fuzzy model BFS^1_{31}. It has three inputs: x_4, x_3 and y^2 (the output of BFS^2_{21}). Fuzzy rules of this sub-model have the following form:

$$R^{l=1}_j: \ If\ x_4\ is\ B^1_{1j}\ and\ x_3\ is\ B^1_{2j}\ and\ y^2\ is\ B^2_{3j}\ then$$
$$y^1 = b^1_{0j} + b^1_{1j}x_4 + b^1_{2j}x_3 + b^1_{3j}y^2$$

where B^1_{1j}, B^1_{2j} and B^2_{3j} are respectively the Beta fuzzy sets of variables x_4, x_3 and y^2. b^1_{0j}, b^1_{1j}, b^1_{2j} and b^1_{3j} are the consequent part parameters.

It should be noted that the Beta MF parameters and the rules consequent parameters of each BFS^l_{kj} will be tuned in the parameter optimization stage.

III. INITIALIZATION STEP USING A CLUSTERING TECHNIQUE

As a first stage of optimization, we focus on the search for the best HFBFS in terms of structure or architecture. But, it is important to know that optimizing the architecture of a given HFBFS using totally random parameters and rules will not give so good results and will slow down the optimization process. That's why, as a first initialization step, the subtractive clustering (SC) algorithm is applied to extract initial membership functions (type Beta) and initial fuzzy rules from the given data. The idea here consists of using a robust clustering method as the basis of a powerful and fast technique in order to initialize more lower-dimensional and accurate sub-fuzzy models from the beginning. And later, these parameters will be further tuned in the process of parameter optimization.

In general, the SC algorithm proposed by Chiu [20] is exploited to select natural groups of data taken from a given dataset. It divides a given data into meaningful and useful subgroups called clusters. Consequently, the generated centers of clusters will represent the membership functions centers of fuzzy sets, and each center of a cluster will be converted into a fuzzy rule. Based on this concept, a mechanism of initialization using this clustering method is applied in this work and is illustrated by the following steps:

After generating an initial random population of trees having random architectures, the SC algorithm is employed in the different levels of these trees. Each tree is examined separately, and for each level, terminal nodes formed by inputs will be clustered to generate the different sub-models BFS^l_{kj}. This initialization phase will ensure an automatic creation of fuzzy rules and of Beta MF parameters for the generated sub-systems. Consequently, it can be said that the resultant extracted fuzzy rules distributed in the different sub-fuzzy models are more adjusted to the input data than they are in sub-fuzzy models created without using clustering. Hence, starting with sub-fuzzy systems having not totally random parameters will facilitate and speed up the whole optimization process. More details about the SC algorithm are presented in [20].

IV. MULTI-OBJECTIVE HIERARCHICAL STRUCTURE OPTIMIZATION USING MOEIP ALGORITHM

A. BASIC CONCEPTS OF EIP FOR SINGLE OPTIMIZATION PROBLEM

Based on some results presented by [21], it has been proven that Immune Programming IP is more efficient than Genetic Programming GP. Indeed, successful solutions are obtained using a smaller generation number with a smaller size of population. Moreover, IP has a great ability to evolve architectures of trees or programs and shows a good flexibility to create more adaptable structures. For all these reasons, in this study, we choose to apply an adjustable version of the IP in the phase of architecture optimization of a HFBFS. The used algorithm is named the Extended Immune Programming (EIP) algorithm and it is detailed by the following steps:

a) Initialization: An initial random population of HFBFSs (antibodies) is created using random architectures (random number of levels and random nodes for each level). The non-terminal node parameters of each HFBFS (Beta membership functions parameters) and the rule bases are initialized by the subtractive clustering algorithm.

b) Evaluation: An antigen describing the problem to be resolved is introduced. The antigen is compared to all the antibodies (NA antibodies), and their fitness Fit(i) (affinity) with respect to the antigen is evaluated.

c) Cloning: Ab_i is an antibody selected from the actual population to be examined; In this step, a random number presenting the probability of cloning P_c is generated. If the affinity of Ab_i is heigher than P_c, so this antibody is cloned and introduced in the next generated population.

d) Mutation: if the previous chosen high-affinity Ab_i has not been cloned because of the stochastic character of this step, so it is presented to hypermutation (Figure 2). In this study, four operators of mutation were applied:

- Modifying one terminal node: choose randomly a terminal node and replace it by another terminal node;

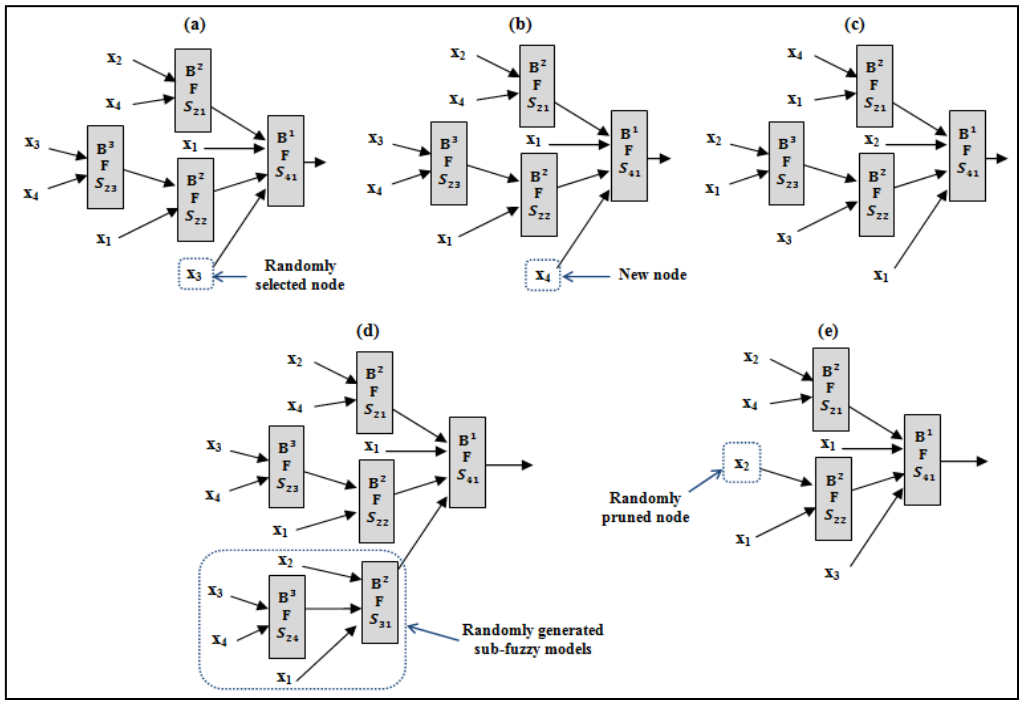

FIGURE 2. (a) Illustration of the EIP mutation operators: (a) Original HFBFS. (b) Modifying one terminal node. (c) Modifying all terminal nodes. (d) Growing. (e) Pruning.

- Modifying all terminal nodes: all terminal nodes of the antibody are selected and replaced by other terminal nodes at random;
- Growing: choose randomly a terminal node and replace it by a randomly generated sub-fuzzy system.
- Pruning: select randomly a sub-fuzzy system (non terminal node) and replace it by another terminal node at random.

To use those mutation operators, the method of Chellapilla [22] was applied by the following steps: (i) Generate a number N (presents a sample from a Poisson variable generated at random), (ii) Choose N operators at random from the previously defined mutation operators, (iii) Execute the N operators consecutively on the parents and generate the offspring.

In fact, the mutation phase is exploited in various manners so that a population with a high genetic diversity is obtained. This diversity will have good effects to overcome local optima.

e) Replacement: if Ab_i is not chosen for mutation or cloning, a novel antibody is generated and introduced in the next population (using a probability of replacement P_r). As a result, low affinity individuals are replaced implicitly.

f) Iteration-population: The EIP operators (steps c–e) are repeated until a full novel population is generated.

g) Iteration-algorithm: after the construction of the new population, the generation counter (EIP_Itr = 0 in the initialization step) is incremented, EIP_Itr = EIP_Itr + 1. So, the EIP is iteratively executed (steps b-f) until a stopping criterion is reached.

In this section, the EIP is presented as a single optimization algorithm and the used fitness function reflects only the accuracy of the system. The next sections will show how this algorithm is extended to improve not only the accuracy but also the interpretability of the generated hierarchical fuzzy system. In this case, the problem is called multi-objective.

B. MULTI-OBJECTIVE OPTIMIZATION PROBLEM

A minimization multi-objective problem is formulated as:

$$Min\, f\,(x) = [f_1\,(x), \ldots, f_k\,(x)] \tag{2}$$

$$\text{subject to: } g_j\,(x) \geq 0 \quad j = 1, \ldots n \tag{3}$$

$$h_j\,(x) = 0 \quad j = 1, \ldots m \tag{4}$$

where $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ is a vector of solutions defined on the space of decision variables. $g_j\,(x)$ and $h_j\,(x)$ are the functions that represent the problem constraints. k defines the number of fitness functions, n and m are the number of equality and inequality constraints respectively.

Unlike single objective optimization problems where one optimum solution is generated, multi-objective optimization methods create a number of optimum solutions named non-dominated or Pareto optimal solutions. The dominance concept is presented as follows: A solution x dominates y (expressed by $x \prec y$) if and only if x is greater than y in at least one fitness function and x is not worse than y in any fitness function. x is named Pareto optimal if x is not dominated by

any other solution of the present population. In the objective space, the set of all Pareto optimal solutions is called the Pareto optimal front.

C. MULTIPLE OBJECTIVE FUNCTIONS

A multi-objective optimization problem requires the consideration of more than one objective function in the optimization process. In this work, both the accuracy and the interpretability are considered in the architecture optimization stage of the hierarchical fuzzy system. The accuracy measure is defined based on the nature of the treated problem. In the case of nonlinear prediction and identification systems, the Root Mean Squared Error (RMSE) is used as an objective function:

$$Objective\ 1:\ RMSE = \sqrt{\frac{1}{K}\sum_{k=1}^{K}\left(y_k - \hat{y}_k\right)^2} \quad (5)$$

where K defines the samples number, y_k and $\hat{y}_k$ are the actual output and the calculated output respectively. For classification problems, we consider another objective function reflecting the error in classification. This function combines the mean square error (MSE) and the classification error as follows:

$$Objective\ 1:\ Error = \frac{1}{K}\left(\sum_{k=1}^{K}\left(y_k - \hat{y}_k\right)^2 + \sum_{k=1}^{K}\left(c_k \neq \hat{c}_k\right)\right) \quad (6)$$

where c and $\hat{c}$ are respecively the class and the predicted class. Regarding the second objective function, the system's interpretability is designated by the number of fuzzy rules (for all the treated problems):

$$Objective\ 2:\ Interpretability = NR \quad (7)$$

where NR defines the total number of fuzzy rules.

D. MULTI-OBJECTIVE EXTENDED IMMUNE PROGRAMMING ALGORITHM (MOEIP)

In this section, An Elitist Strategy based on the Extended Immune Programming operators and a dominance concept is proposed. This Strategy makes use of an archive A (secondary population), in which a number of non-dominated individuals of antibodies are stored. In fact, each solution of the population represents a HFBFS (tree), and as the search progresses, the dominance criterion is applied to evolve this population towards an optimal Pareto Front.

In addition, in the case of single optimization, a child is usually selected over its parent (in the evolutionary process) if it has a better value of fitness function. In our proposed multi-objective algorithm, the superiority is measured as a dominance relationship, and a child is selected over its parent only if this latter dominates its parent. As a result, as the search progresses, the different solutions move closer more to the Pareto front. To additionally ensure a good diversity of the obtained Pareto front, a pruning method using the crowding distance value is also used (applied in Non-dominated Sorting Genetic Algorithm II: NSGA II [23]). The crowding distance of each solution of the archive presents the distance between this solution and its neighbors in the space of fitness function. In each generation, the solutions of the archive A are sorted according to their values of crowding distance. And next, solutions having the best diversity (with the highest crowding distance values) are maintained and the worst are discarded from the archive.

Suppose that x_i is a solution of the Pareto front. The crowding distance $cd\ (x_i)$ of x_i is computed by the following steps:

i) The crowding distance of x_i is initialized: $cd\ (x_i) = 0$;
ii) For each objective function f_j do:
- Sort the different non-dominated solutions along the objective function f_j;
- $cd\ (x_i) = cd\ (x_i) + f_j$ (the solution which precedes x_i in the ordered sequence) - f_j (the solution which follows x_i in the ordered sequence);

To summarize, the basic ideas of the MOEIP are:
- Evolve the structure of the HFBFS and take into consideration the amelioration of the accuracy and the interpretability in the same time.
- Use of EIP operators combined with a dominance concept to guide the search through an optimal Pareto-front of non-dominated solutions.
- Keep a diverse set of non-dominated solutions using the crowding distance measure.
- In the mutation step, the replacement of antibodies members is done using dominating offspring.

A general pseudo code of the MOEIP is introduced by Algorithm 1. It should be noted that the MOEIP algorithm stops when the maximum iteration's number is reached. As a result to this optimization step, a Pareto front of optimal HFBFSs is obtained. In this case, the most suitable solution (which has a good tradeoff between the two objective functions) is chosen from the generated Pareto front to perform a second step of parameter tuning.

V. PARAMETER OPTIMIZATION USING THE HABC ALGORITHM

A. ARTIFICIAL BEE COLONY ALGORITHM (ABC)

Inspired by the intelligent foraging behavior of honey bee swarm, Karaboga proposed in 2005 the Artificial Bee Colony (ABC) algorithm [24] as a swarm intelligence-based algorithm. ABC is considered as one of the most important algorithms in the field of bee-inspired approaches. In general, three types of bees exist in ABC according to the searching manner for food: employed bees, scout bees and onlooker bees. Employed bees have the function of exploiting the sources of nectar. They share the knowledge about the amount of nectar with onlooker bees that wait at the nest. Onlooker bees establish communication with the employed bees to locate sources of food. And Scout bees look for a new nectar source by searching the space randomly. In this algorithm, the food source position is considered as a solution of the problem, while the amount of food represents the fitness of

Algorithm 1: MOEIP Algorithm

Input: *NA* (size of the population)
Max_Itr: maximum iteration number
Max_Size: the archive maximum size
Output: *A* (archive containing non-dominated solutions)

Begin

1: Generation of a random population P_0 of initial antibodies (HFBFSs) having random architectures and creation of an initial external empty archive $A_0 = \emptyset$.
$MOEIP\_Itr = 0$;

2: Evaluation of P_0.

3: Applying dominance criterion on P_0 in order to store the initial Pareto optimal solutions in the archive A_0.

4: Pruning of A_0 based on the crowding distance value: If the archive size (solutions number) is greater than *Max_Size*, then the crowding distances of all individuals of the archive are calculated and sorted in a descending order. The first *Max_Size* antibodies are then chosen to update the archive.

5: **While** ($MOEIP\_Itr < Max\_Itr$) **do**

6: Update of the population P_t: Apply EIP operators on $P_t \cup A_t$ to generate the new population P_{t+1}:
- Cloning (P_c).
- Mutation: if an offspring dominates the parent,
then this latter is replaced by its offspring.
- Replacement (P_r).

7: Update of A_t: Applying dominance criterion on $A_t \cup P_{t+1}$ to create the external archive A_{t+1}.

8: Pruning of A_{t+1} based on the crowding distance measure (as in step 4).

9: $MOEIP\_Itr = MOEIP\_Itr + 1$.

End

10: Returning the final non-dominated solutions of A_t.

End

the solution. The employed bees number is equal to the food sources number.

When collecting honey, each employed bee searches for a source of nectar v_i that has more amount of food in the neighborhood of the present source. The employed bee memorizes the best found position of nectar x_i in the neighborhood of the current position based on the following formula:

$$v_i = x_i + \varphi 1(x_i - x_k) \tag{8}$$

where x_k is a randomly chosen solution (source of food), $\varphi 1$ is a random number within the range $[-1, 1]$. v_i and x_i are then compared and the employed bee memorizes the best nectar source position.

Next, the onlooker bees choose a food source based on the probability associated with that source. The probability value of being chosen as a food source is given by:

$$P_i = \frac{Fit_i}{\sum_{i=1}^{NP} Fit_i} \tag{9}$$

where Fit_i presents the i^{th} solution fitness value and NP presents the number of solutions (sources of food).

After a number of trials, if a source of food is not improved, then this solution will be abandoned. So, the corresponding employed bee becomes a scout bee looking for a new source of food at random according to:

$$x_i = a_j + rand_i(b_j - a_j) \tag{10}$$

where $\boldsymbol{b_j}$ and $\boldsymbol{a_j}$ are respectively the upper and the lower bounds for decision variable j.

B. HYBRID ARTIFICIAL BEE COLONY ALGORITHM (HABC)

Although its efficiency, ABC in its original version still has some weakness. In fact, we can remark that the global best solution does not be directly used in this algorithm. The scout bee can replace, at some limit, a best solution by a random one. At the same time, ABC still suffers from insufficiency in regard to its solution search equation, which has great exploration ability but is bad at exploitation. Such disadvantages may affect the performance of the algorithm and slow down its convergence speed.

Considering the good exploitation ability of the Opposite-based Particle Swarm Optimization (OPSO) algorithm [25], [26], Bouaziz *et al.* [19] proposed an ameliorated version of ABC based on OPSO search mechanism. The new algorithm is called the Hybrid Artificial Bee Colony algorithm (HABC). In this hybrid algorithm, employed bees, scout bees and onlooker bees use the OPSO mechanism to find new solutions. To realize this hybridization, a position and an opposite-position of a particle (in OPSO) are considered as food source positions (in HABC). Thus, the solution search equation of the original ABC (Equation (8)) is replaced by the following OPSO position equations:

$$x_i(t+1) = x_i(t) + (1 - \Psi(t))\, v_i(t+1) \tag{11}$$

$$\bar{x}_i(t+1) = \bar{x}_i(t) + (1 - \Psi(t))\, v_i(t+1) \tag{12}$$

where x_i and $\bar{x}_i$ are respectively the position and the opposite-position of particle i. Ψ is a positive constant presenting the inertia weight, and v_i is the velocity of the i^{th} particle defined as follows:

$$v_i(t+1) = \Psi(t)\, v_i(t) + c_1\varphi_1(p_i(t) - x_i(t)) + c_2\varphi_2\left(p_g(t) - x_i(t)\right) \tag{13}$$

where φ_1 and φ_2 are random numbers in [0,1], and c_1 and c_2 are constant factors. p_i and p_g are respectively the local best position of particle i, and the global best position (found by the swarm).

Readers can find more details concerning this algorithm in [19]. The pseudo code of HABC is given by Algorithm 2.

Algorithm 2: HABC Algorithm

Input: *NP* (size of the population)
Max_Itr: maximum iteration number
limit: a limit number
Output: best solution

Begin

1: Initialize the food sources and evaluate the population and the opposite population.
$trial_i = 0, (i = 1, \ldots, NP)$
$HABC\_Itr = 1$;

2: **While** ($HABC\_Itr < Max\_Itr$) **do**

3: Produce the solution x_i according to (11) and the opposite solution $\bar{x}_i$ according to (12) for employed bees and evaluate them.

4: Select the best solution between these two solutions using the fitness value.

5: If a solution does not improve, $trial_i = trial_i + 1$, otherwise $trial_i = 0$.

6: Compute the P_i probability according to (9) and use the roulette wheel selection method to choose a source of food for onlooker bees.

7: Produce the solution x_i according to (11) and the opposite solution $\bar{x}_i$ according to (12) for onlooker bees and evaluate them.

8: Select the best solution between these two solutions using the fitness value.

9: If solution does not improve, $trial_i = trial_i + 1$, otherwise $trail_i = 0$.

10: Update the local best position p_i and the global best position p_g.

11: If (Max($trial_i$) > *limit*)
Replace this source of food with a new randomly produced source of food by (10) and evaluate it.

12: Save the best solution achieved so far.

13: $HABC\_Itr = HABC\_Itr + 1$.

End

14: **Return** the best solution

End

In this study, the HABC is implemented as a parameter optimization phase for the optimization of the obtained HFBFS. Indeed, the parameters of the best HFBFS (created by the MOEIP structure optimization phase) constitute a food source position (solution) to be evolved via the HABC. So, each solution is represented by a $NParm \times Size$ matrix. *NParm* presents the parameters number while *Size* illustrates the number of the non terminal BFSs (sub-fuzzy models) of the best HFBFS found. The free parameters encoded in the matrix are the Beta parameters of MFs (center, spread, p and q) and the linear weights of the consequent parts of fuzzy rules. The employed objective function in this parameter optimization process is given by (5) or (6) (according to the nature of the treated problem).

VI. HYBRID ALGORITHM TO EVOLVE THE HFBFS

An HFBFS is considered to be optimal or near optimal solution if it has the best hierarchical structure with the optimal set of parameters. For that, the MOEIP and the HABC are combined and are alternately applied in order to find an optimum or a near-optimum HFBFS with good precision and a minimum number of rules. The flowchart of the proposed hybrid algorithm is given by Figure 3. *Global_Itr* and *Global_Gn* present respectively, the global iteration number and the global generation number. The main steps of the hybrid algorithm are described as follows:

1. Generate an initial population of HFBFSs with random structures and initialize their corresponding parameters (by the subtractive clustering algorithm).
   - $Global\_Gn = 0$;
   - $Global\_Itr = 0$;
2. Apply Multi-Objective Hierarchical Structure Optimization using the MOEIP. If the maximum number of MOEIP iterations is attained, then:
   - Generate the final Pareto front of non-dominated solutions. Each solution of the Pareto front corresponds to a HFBFS tree;
   - Choose one near-optimal HFBFS structure (in terms of accuracy and interpretability) to be the output of the multi-objective structure optimization phase;
   - $Global\_Itr = Global\_Itr + MOEIP\_Itr$;
   - Go to step (3);

   Otherwise return to step (2);
3. Apply Parameter Optimization using the HABC: The parameters of the best HFBFS (found by the multi-objective structure search) formulate a food source position. The rest of HABC population is generated at random in the allowable range of parameters.
4. If the maximum number of HABC iterations is reached, or no better parameter matrix is generated after a fixed time, then:
   - Encode the matrix of best parameters found on the fixed HFBFS structure and evaluate it;
   - $Global\_Itr = Global\_Itr + HABC\_Itr$;
   - Go to step (5);

   Otherwise return to step (3);
5. If a satisfactory HFBFS is generated or a maximum number of global iterations is reached, then stop; Otherwise, $Global\_Gn = Global\_Gn + 1$ and return to step (2);

VII. EXPERIMENTAL RESULTS

The proposed HFBFS is evaluated and is applied to predict and identify nonlinear systems, including two kinds of time series problems and one nonlinear plant identification problem. In addition, the classification performance of the system is also evaluated through testing two kinds of

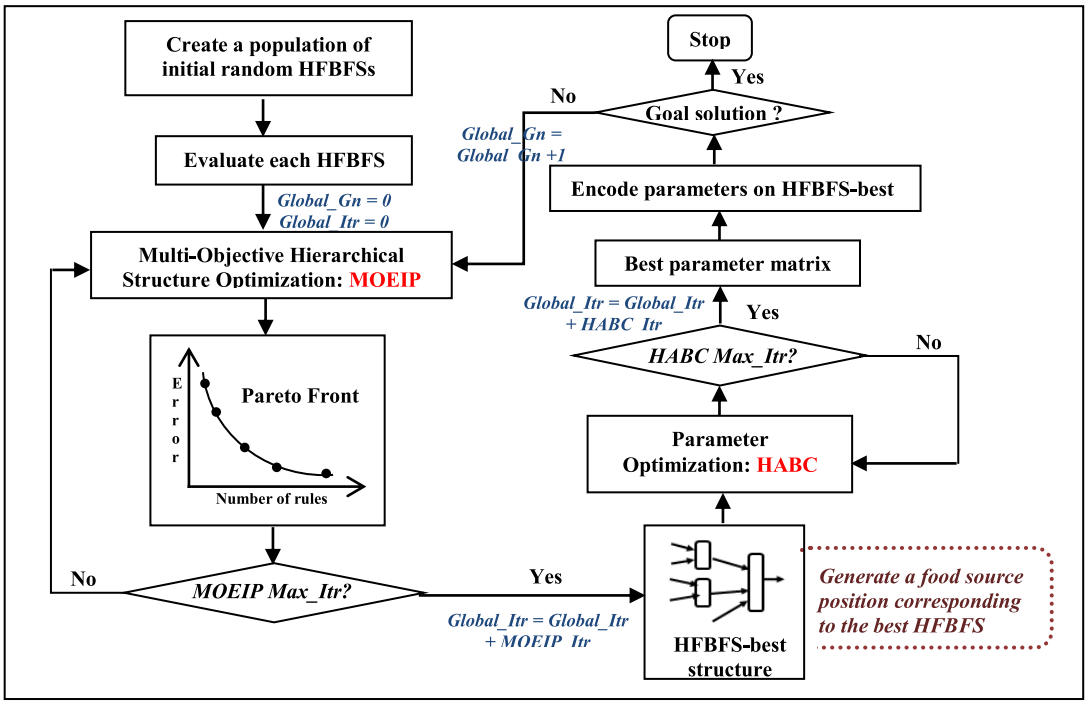

FIGURE 3. Flowchart of the hybrid algorithm.

TABLE 1. Initialization of parameters.

Algorithm	Parameter	Value
MOEIP	Size of population	20
	Probability of cloning (*Pc*)	0.7
	Probability of replacement (*Pr*)	0.5
HABC	Size of population	20
	LimitTrial	20
	c1	0.8
	c2	0.8

high-dimensional classification problems. The best-suited list of parameters employed in the simulations is shown in Table 1. Results are generated and compared with other neural/fuzzy learning techniques taken from the literature. The comparison is based on the following measures of performance:

- Accuracy or quality of solution: presented by the training and testing error values;
- Rule base complexity: presented by the number of used fuzzy rules (interpretability);
- Time complexity or convergence speed: presented by the number of global iterations (*Global_Itr*) and the Function Evaluations number (*FEs*);

A. HFBFS FOR NONLINEAR PREDICTION AND IDENTIFICATION SYSTEMS

To construct a meaningful comparison with other works from the literature, the used datasets are divided into the same number of training and testing samples. The experimentations are running 10 times and results are then averaged.

1) CHOATIC TIME-SERIES OF BOX AND JENKINS' GAS FURNACE

The Gas Furnace Problem is a time series problem related to a combustion procedure of a methane-air mixture. This example is usually exploited as a benchmark problem to test prediction approaches. The used dataset contains 296 observations derived from a laboratory furnace [27]: the initial 200 samples and the remaining 96 samples were employed respectively for the training and for the test. The gas flow into the furnace is considered as input $u(t)$, and the concentration of CO2 in outlet gas is considered as output $y(t)$. The goal here is to use $y(t-1)$ and $u(t-4)$ to predict $y(t)$.

After achieving an average number of 7 global iterations, 1 global generation and 291 number of Function Evaluations (*FEs*), the generated RMSE values for training and testing data are respectively 0.0049 and 0.0116. The number of fuzzy rules of the obtained optimum HFBFS is equal to 5. The evolved HFBFS, the desired output and the predicted output are illustrated through Figure 4. Results prove the high performance of the HFBFS which can attain low rates of training and testing errors after only few hundreds of *FEs* and using a few number of rules.

A performance comparison with different learning methods from the literature is shown in Table 2. The proposed approach is principally compared with the Belief Rule Based (BRB) system [28], the fuzzy modeling approach using Memetic Algorithms (MAs) and Genetic Algorithms (GAs) [29], the Beta Basis Function Neural Network's system using Hierarchical Particle Swarm Optimization: HPSO-BBFNN [30], the Single Multiplicative Neuron model designed by a Particle Swarm Optimization

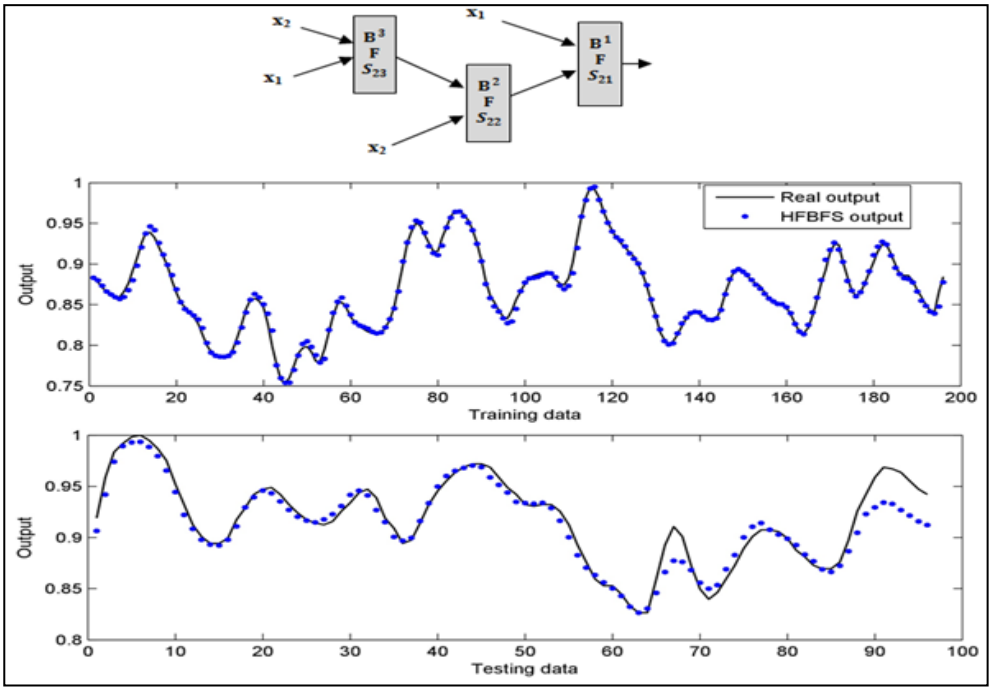

FIGURE 4. The evolved HFBFS, the real output and the computed output for the training and testing sets in the case of Jenkins–Box problem.

TABLE 2. Results comparison in the case of Jenkins–Box problem.

Approach	Rules	Training RMSE	Testing RMSE
FNT model [33]	-	0.0257	0.0264
ANFIS [31]	-	0.0374	0.0640
FBeM [32]	3	0.0421	-
COPSO-SMN [31]	-	0.2151	0.3416
BRB system [28]	25	0.2939	0.4616
HPSO-BBFNN [30]	-	0.2258	0.3876
Fuzzy_GA [29]	8	0.2779	-
Fuzzy_MA [29]	8	0.0871	-
Fuzzy_probability based clustering [34]	-	-	0.0190
HFBFS	**5**	**0.0049**	**0.0116**

algorithm with Co-Operative sub-swarms: COPSO-SMN [31], the Adaptive Neuro-Fuzzy Inference System ANFIS [31], the fuzzy set based granular evolving modeling approach: FBeM [32], the Flexible Neural Tree model: FNT [33] and the fuzzy model with a probability based clustering method [34]. As remarked, the developed HFBFS outperforms the competing methods and attains the lowest training and testing error using a few number of rules.

2) SUNSPOT NUMBER TIME SERIES PREDICTION

The sunspot number problem is considered as a difficult scientific benchmark. It constitutes a non stationary and high complex real world time series. This series presents the yearly average relative number of sunspot observed [35]. It is recorded for the years 1700-1979.

TABLE 3. Results comparison in the case of sunspot number problem.

Approach	Training RMSE	Testing 1 RMSE	Testing 2 RMSE
RFNN [37]	-	7.4e-02	2.1e-01
FWNN [36]	2.5e-01	3.3e-01	5.2e-01
ABC-BBFNN [38]	1.2e-03	1.8e-03	4.4e-03
FBBFNT [39]	1.9e-10	4.1e-10	7.2e-10
HFBFS	**4.8e-16**	**6.3e-16**	**8.5e-16**

To make a meaningful comparison with other existing models, we use the same partitioning of data. So, samples between 1700 and 1920 are exploited as training data. And for testing, two additional sets are employed: the first set is from 1921 to 1955 and the next set is from 1956 to 1979. Input variables are $y(t-4)$, $y(t-3)$, $y(t-2)$ and $y(t-1)$ and the predicted output is $y(t)$.

After accomplishing an average number of 15 global iterations, 1 global generation and 610 number of *FEs*, an optimized HFBFS was generated with an RMSE training value equal to 4.856e-016. For the testing data, the testing RMSE values for the first and the second sets are respectively 6.3350e-016 and 8.5660e-016. The number of the generated fuzzy rules is equal to 6. Figure 5 shows the evolved HFBFS, the desired and the predicted outputs for the training and testing sets.

Table 3 illustrates a comparison with other existing works such as Fuzzy Wavelet Neural Network models: FWNN [36] and Recurrent Fuzzy Neural Networks: RFNN [37]. The HFBFS is also compared with two other approaches which

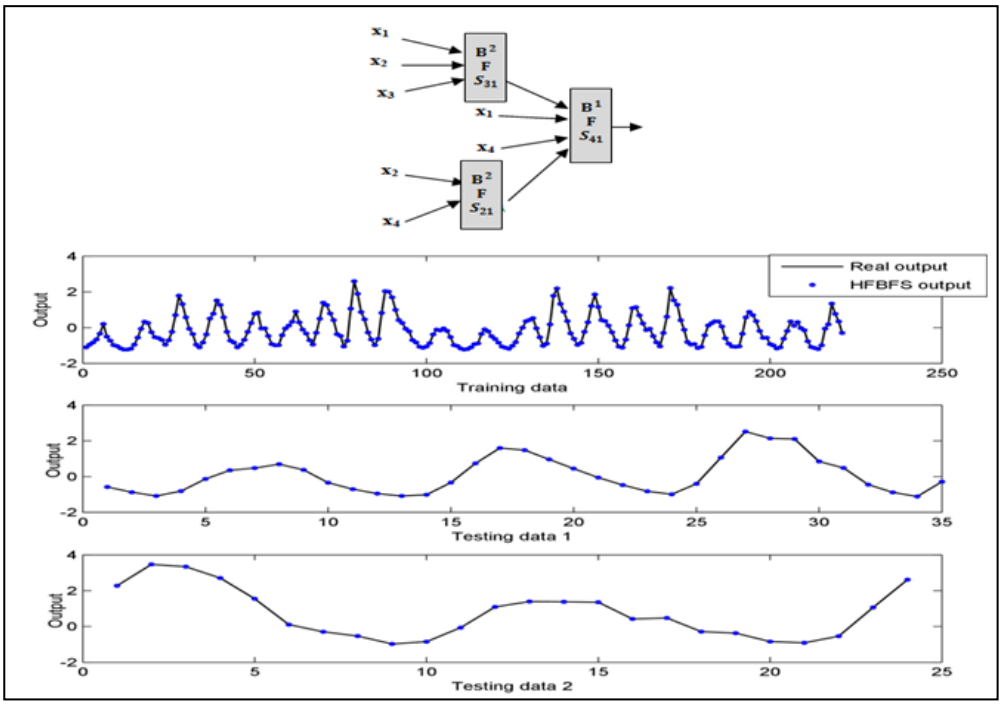

FIGURE 5. The evolved HFBFS, the real output and the calculated output for training and testing sets in the case of sunspot number dataset.

use the Beta basis function for designing Artificial Neural Networks. These two approaches are the Beta Basis Function Neural Networks trained by the Artificial Bee Colony algorithm: ABC-BBFNN [38], and the Flexible Beta Basis Function Neural Tree model trained by a hybrid evolutionary algorithm: FBBFNT [39]. As shown in Table 3, the HFBFS proves again its superiority for the sunspot number problem.

3) NONLINEAR PLANT IDENTIFICATION

The nonlinear system [27] to be identified is described by:

$$y_p(t+1) = \frac{y_p(t)\left[y_p(t-1)+2\right]\left[y_p(t)+2.5\right]}{8.5+\left[y_p(t)\right]^2+\left[y_p(t-1)\right]^2} + u(t) \tag{14}$$

where $y_p(t)$ defines the system output in the t^{th} time. The input of the plant is defined by $u(t)$ and is uniformly bounded in $[-2, 2]$. The system to be identified has the following form:

$$y_{pi}(t+1) = f\left[y_p(t), y_p(t-1)\right] + u(t) \tag{15}$$

The nonlinear function $f\left[y_p(t), y_p(t-1)\right]$ presents the input to the system, and $y_{pi}(t+1)$ presents the output. For the simulations, we use the first 500 data samples for training and the other 500 data samples for the test. $u(t)$ constituting the input signal is computed as follows:

$$u(t) = \begin{cases} 2\cos\left(\dfrac{2\pi t}{100}\right) & if\ t \le 200 \\ 1.2\sin\left(\dfrac{2\pi t}{100}\right) & if\ 200 \le t \le 500 \end{cases} \tag{16}$$

TABLE 4. Results comparison in the case of nonlinear identification system.

Method	Training RMSE	Testing RMSE	*Global_Itr*	*FEs*
HMDDE [41]	1.9e-02	1.1e-01	10000	-
ODE-NN [40]	1.9e-02	1.1e-01	1000	-
FBBFNT [19]	1.8e-10	2.1e-10	15067	270,311
HFBFS	**4.8e-16**	**4.7e-16**	**11**	**408**

After accomplishing 11 global iterations, 1 global generation and 408 number of *FEs*, an optimal HFBFS having 7 rules is generated with 4.8006e-16 value for training RMSE and 4.7210e-16 for testing RMSE. The evolved HFBFS, the actual output and the predicted output are illustrated through Figure 6.

Simulation results are illustrated in Table 4 and are compared with results of other studies such as the opposition based differential evolution neural network system ODE-NN [40], the hierarchical multi-dimensional differential evolution (HMDDE) algorithm for neural network optimization [41] and the flexible beta basis function neural tree (FBBFNT) [19]. It is remarkable from this table that the HFBFS gives the best results for the training and testing errors as well as for the number of iterations and *FEs*.

B. HFBFS FOR HIGH-DIMENSIONAL BIOMEDICAL CLASSIFICATION PROBLEMS

For this type of high dimensional problems, we apply a feature selection algorithm as a first filtering step to choose the most important inputs. Generally, the modeling of

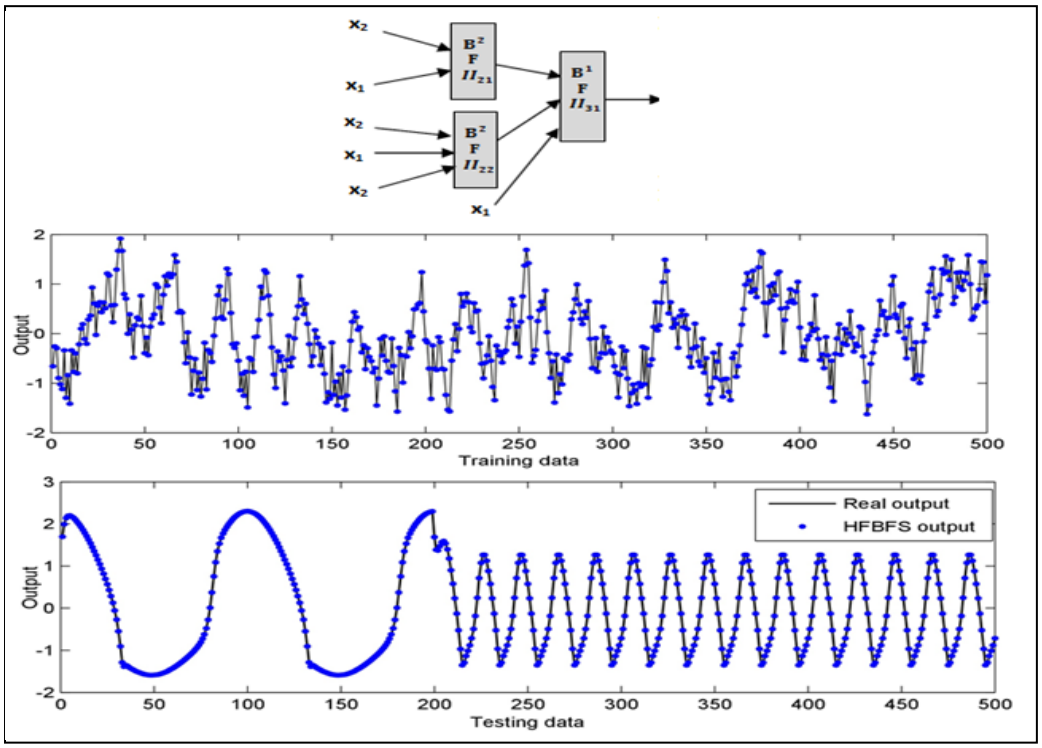

FIGURE 6. The evolved HFBFS, the actual output and the predicted output for training and testing sets in the case of the nonlinear plant identification problem.

high-dimensional problems is usually anteceded by a data preparation phase known as feature selection. The use of such step aims to reduce the feature space instead of exploiting all features of the original dataset, and consequently it allows enhancing the prediction performance of the system. In this study, a feature scoring technique named the Statistical Dependency (SD) algorithm [42] is used as a data preparation phase when the tested dataset is formed by a high input dimension.

Moreover, to well evaluate the efficiency of the approach, 5-fold cross validation method was performed on classification data sets as it gives an honest assessment of the true classification accuracy of the system. In general, cross-validation is based on a random split of the dataset into disjoint training and validation sets. This procedure is repeated and leads to more accurate results.

In addition to the used measures of performance, the effectiveness of the system is also evaluated using three other popular metrics generally employed in binary classification problems. These measures are the accuracy, the sensitivity (also called the true positive rate) and the specificity (also called the true negative rate):

$$Accuracy = (TP + TN)/(TP + TN + FN + FP) \quad (17)$$

$$Specificity = TN/(FP + TN) \quad (18)$$

$$Sensitivity = TP/(FN + TP) \quad (19)$$

where, TP (True Positive) is correctly identified, TN (True Negative) is correctly rejected, FP (False Positive) is incorrectly identified and FN (False Negative) is incorrectly rejected. In general, the specificity gives an idea about how well the system can predict one category and the sensitivity shows how well the system can predict the other category. Whereas, the accuracy indicates how well the system can predict both categories. In the context of medical tests, high rates of sensitivity indicate that few actual cases of disease are undetected. Whereas, high rates of specificity show that few healthy people are labeled as sick.

1) LUNG CANCER DATASET

This dataset is a classification dataset downloaded from the Kent Ridge Bio-medical Data Set Repository [43] and is employed for the classification of malignant pleural mesothelioma (MPM) and adenocarcinoma (ADCA) of the lung. It is formed by 181 samples: Among them 150 tissues are ADCA and the other 31 are MPM. Each sample is characterized by 12533 genes (features).

After executing the SD input feature selection algorithm, the features number is reduced from 12533 to 25 features. The results of classification accuracy in 5-fold cross validation for training and testing data are respectively 98.33% and 97.24%. The other measures of performance including specificity, sensitivity, the error, the number of rules and the *FEs* number are summarized in Table 5. These classification results are obtained after undergoing 12 global iterations, 1 global generation and 420 number of *FEs*. The rules number of the optimum HFBFS is equal to 9. Results of Table 5 prove the good performance of the suggested approach in terms of

TABLE 5. Classification results for the Lung cancer dataset.

Metrics	Training set	Testing set
Accuracy	98.33%	97.24%
Specificity	97.93%	97.43%
Sensitivity	96.53%	96.13%
Error	0.038	0.048
Rules	9	
FEs	420	
Selected features	25	

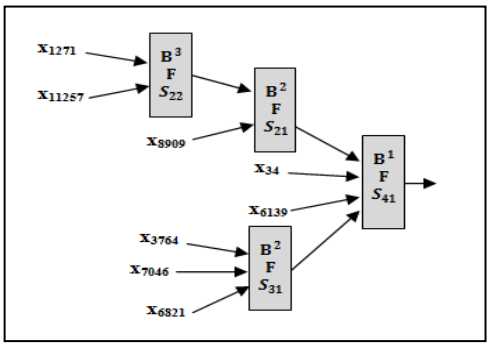

FIGURE 7. The obtained HFBFS for the Lung Cancer dataset.

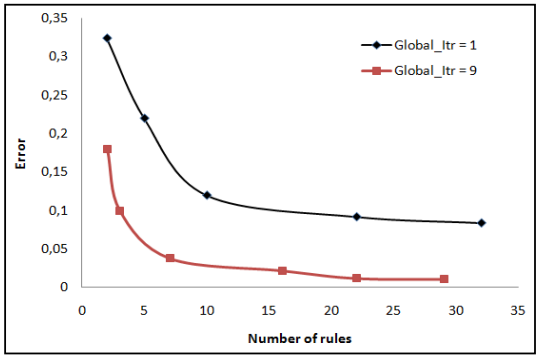

FIGURE 8. Pareto front Evolution for the Lung cancer dataset.

obtaining high rates of classification (high values for all the three classification factors) using few rules and in a reduced time (with a limited number of *FFs*). The evolved generated HFBFS is drawn in Figure 7.

To more evaluate the effectiveness of the evolutionary process, the evolution of the Pareto front during the learning task is also studied. Hence, we compare in Figure 8 the generated Pareto fronts over a period of time. The figure shows the evolution of the generated Pareto fronts during the optimization process (from $Global\_Itr = 1$ to $Global\_Itr = 9$). The first generated Pareto front after 1 global iteration ($Global\_Itr = 1$) contains the non-dominated solutions at the beginning of the optimization phase. We can clearly remark the significant improvement of the front's solutions after performing 9 global iterations. As shown in the figure, the front in red dominates the front in black showing a great amelioration of non-dominated solutions. This affirms the efficiency of the multi-objective training process and its capacity to ameliorate the solutions quality as the search progresses.

TABLE 6. Comparison of classification performance in the case of Lung Cancer dataset.

Approach	Rules	Classification accuracy
MOEAIF [45]	3	91.28%
Bootstrapping gene selection [44]	-	91.2%
Fuzy-PCA-ACO [47]	20	98.21%
LDA-GA [46]	-	97.7%~98.3%
HFBFS	**9**	**98.33%**

TABLE 7. Classification results for the Prostate cancer dataset.

Metrics	Training set	Testing set
Accuracy	96.42%	95.86%
Specificity	98.33%	97.12%
Sensitivity	94.57%	94.62%
Error	0.065	0.076
Rules	8	
FEs	514	
Selected features	20	

In order to make more meaningful analysis of results, a comparison with other existing methods is illustrated in Table 6. The HFBFS is mainly compared with a Bootstrapping gene selection method [44], a Multi-Objective Evolutionary Algorithm based Interpretable Fuzzy method: MOEAIF [45], a wrapper approach based on a Genetic Algorithm combined with Fisher's Linear Discriminant Analysis: LDA-GA [46] and a fuzzy modeling method using the Principal Components Analysis (PCA) as a data reduction method and the Ant Colony algorithm (ACO) for optimization [47]. As compared with those approaches, our system has the advantage of obtaining a good trade-off between the classification performance and the fuzzy rules number.

2) PROSTATE CANCER DATASET

The final used dataset in this experimentation is the prostate cancer data [43] for tumor versus normal classification. It is formed by 12600 features and 136 samples (77 prostate samples contain tumors and 59 do not contain tumors). The features present normalized gene expression values taken from the microarray image.

For this dataset, the features number is reduced by the SD algorithm from 12600 to 20. After accomplishing 14 global iterations, 1 global generation and 514 number of *FEs*, an optimal HFBFS was generated with 8 rules. Results of this experimentation are listed in Table 7. As seen in the table, the average of classification rates on the training sets is 96.42% and on the testing sets is 95.86%. In addition to the accuracy metric, the high average values of specificity

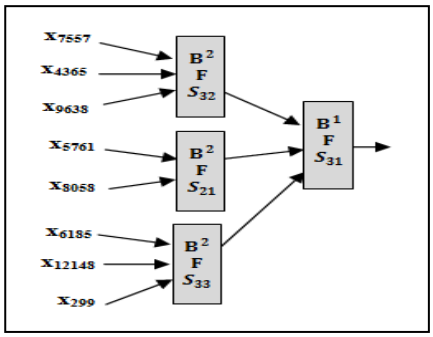

FIGURE 9. The obtained HFBFS for the Prostate Cancer dataset.

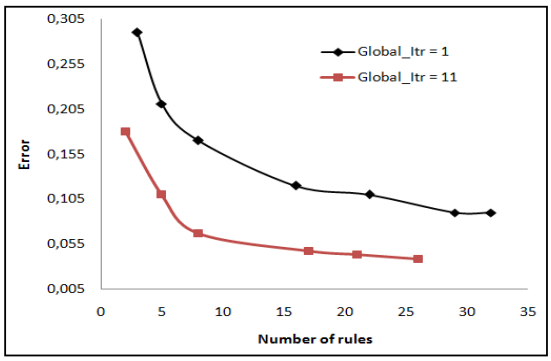

FIGURE 10. Pareto front Evolution for the Prostate cancer dataset.

TABLE 8. Comparison of classification accuracyr in the case of Prostate Cancer data set.

Approach	Classification accuracy
GSVM-RFE [49]	90.18%
TSVM [48]	92%
AR-FRS-GR [48]	94.3%
HFBFS	**96.42%**

and sensitivity prove that tests have high credibility and the proposed system is a well performing model. Regarding the rule base complexity, we can see that the system succeeds to reach a good balance between the classification rates and the number of used rules, which is the aim of this work. The best evolved HFBFS is shown in Figure 9.

The evolution of the Pareto front during the learning process is also shown by Figure 10. The figure aims to evaluate the effectiveness of the employed multi-objective immune process for classification problems. If we compare the generated Pareto fronts at the beginning of learning ($Global\_Itr = 1$) and after performing 11 global iterations of optimization ($Global\_Itr = 11$), we can observe the great amelioration of non dominated solutions constituting the Pareto fronts when the search progresses.

The proposed approach is essentially compared with a classification method using Association Rule and an attribute selection information Gain Ratio on Fuzzy Rough Set theory: AR-FRS-GR [48], a Transductive Support Vector Machine: TSVM [48] and a Granular Support Vector Machines-Recursive Feature Elimination: GSVM-RFE [49]. As shown in Table 8, our system gets again higher average classification accuracy than the other existing works.

VIII. CONCLUSION

This work presents a new hierarchical fuzzy learning methodology based on an automated arrangement of low-dimensional rule bases in a hierarchical architecture. The proposed hybrid approach evolves the hierarchical structure and the parameters of the HFBFS with the aim of attaining the desired balance between the accuracy and the system interpretability. To do this, two different machine learning techniques, MOEIP and HABC, are applied iteratively until a near optimal HFBFS is attained. In addition, a clustering technique and an input selection method (for classification problems) are used as two initial steps to reduce the computational effort and to obtain successful solutions in fewer iteration numbers. The experimentation results are very encouraging in comparison with other referenced works and prove the efficiency of this methodology for time series prediction problems, nonlinear plant identification and classifications problems. In fact, the obtained results are competitive not only in regards to reaching a good accuracy rate, but also in regards to reducing the number of rules, generating smaller size of HFBFSs and also attaining good convergence speed.

REFERENCES

[1] Y. Jarraya, S. Bouaziz, A. M. Alimi, and A. Abraham, "Evolutionary multi-objective optimization for evolving hierarchical fuzzy system," in *Proc. IEEE. Congr. Evol. Comput.*, Sendai, Japan, May 2015, pp. 3163–3170.
[2] M. Z. Jahromi and M. R. Moosavi, "Designing cost-sensitive fuzzy classification systems using rule-weight," in *Proc. 1st Int. Conf. Adv. Inf. Mining Manage. (IMMM)*, 2011, pp. 168–173.
[3] E. Boutleux and B. Dubuisson, "Fuzzy pattern recognition to characterize evolutionary complex systems. Application to the french telephone network," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, New Orleans, LA, USA, Sep. 1996, pp. 780–785.
[4] E. Hüllermeier, "Fuzzy methods in machine learning and data mining: Status and prospects," *Fuzzy Sets Syst.*, vol. 156, no. 3, pp. 387–406, Dec. 2005.
[5] K. Tanaka and M. Sano, "A robust stabilization problem of fuzzy control systems and its application to backing up control of a truck-trailer," *IEEE Trans. Fuzzy Syst.*, vol. 2, no. 2, pp. 119–134, May 1994.
[6] P. Singhala, D. N. Shah, and B. Patel, "Temperature control using fuzzy logic," *Int. J. Instrum. Control Syst.*, vol. 4, no. 1, pp. 1–10, Jan. 2014.
[7] Y. Jarraya, S. Bouaziz, A. M. Alimi, and A. Abraham, "Fuzzy modeling system based on hybrid evolutionary approach," in *Proc. 13th Int. Conf. Hybrid Intell. Syst. (HIS)*, Hammamet, Tunisia, Dec. 2013, pp. 72–77.
[8] Y. Jarraya, S. Bouaziz, A. M. Alimi, and A. Abraham, "Multi-agent evolutionary design of Beta fuzzy systems," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Beijing, China, Jul. 2014, pp. 1234–1241.
[9] G. V. S. Raju and J. Zhou, "Adaptive hierarchical fuzzy controller," *IEEE Trans. Syst., Man and*, vol. 23, no. 4, pp. 973–980, Jul. 1993.
[10] M. G. Joo and T. Sudkamp, "A method of converting a fuzzy system to a two-layered hierarchical fuzzy system and its run-time efficiency," *IEEE Trans. Fuzzy Syst.*, vol. 17, no. 1, pp. 93–103, Feb. 2009.
[11] K. Balázs, J. Botzheim, and L. T. Kóczy, "Hierarchical fuzzy system modeling by genetic and bacterial programming approaches," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Barcelona, Spain, Jul. 2010, pp. 1–6.
[12] P. Salgado, "Rule generation for hierarchical collaborative fuzzy system," *Appl. Math. Model.*, vol. 32, no. 7, pp. 1159–1178, Jul. 2008.
[13] L.-C. Lin and G.-Y. Lee, "Hierarchical fuzzy control for C-axis of CNC turning centers using genetic algorithms," *J. Intell. Robot. Syst.*, vol. 25, no. 3, pp. 255–275, 1999.

[14] X.-J. Zeng and J. A. Keane, "Approximation capabilities of hierarchical fuzzy systems," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 5, pp. 659–672, Oct. 2005.

[15] F.-L. Chung and J.-C. Duan, "On multistage fuzzy neural network modeling," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 2, pp. 125–142, Apr. 2000.

[16] A. M. Alimi, "The beta fuzzy system: Approximation of standard membership functions," in *Proc. 17th J. Tunisiennes d'Electrotechn. d'Autom. (JTEA)*, Nabeul, Tunisia, vol. 1. Nov. 1997, pp. 108–112.

[17] A. M. Alimi, "Beta fuzzy basis functions for the design of universal robust neuro-fuzzy controllers," in *Proc. Séminaire Commande Robuste Appl. (SCRA)*, Nabeul, Tunisia, Feb. 1997, pp. C1–C5.

[18] A. M. Alimi, R. Hassine, and M. Selmi, "Beta fuzzy logic systems: Approximation properties in the mimo case," *Int. J. Appl. Math. Comput. Sci.*, vol. 13, no. 2, pp. 225–238, 2003.

[19] S. Bouaziz, H. Dhahri, A. M. Alimi, and A. Abraham, "Evolving flexible beta basis function neural tree using extended genetic programming & hybrid artificial bee colony," *Appl. Soft. Comput.*, vol. 47, pp. 653–668, Oct. 2016.

[20] S. Chiu, "Fuzzy model identification based on cluster estimation," *J. Intell. Fuzzy Syst.*, vol. 2, no. 3, pp. 267–278, Sep. 1994.

[21] P. Musilek, A. Lau, M. Reformat, and L. Wyard-Scot, "Immune programming," *Inf. Sci.*, vol. 176, no. 8, pp. 972–1002, Apr. 2006.

[22] K. Chellapilla, "Evolving computer programs without subtree crossover," *IEEE Trans. Evol. Comput.*, vol. 1, no. 3, pp. 209–216, Sep. 1997.

[23] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[24] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Dept. Comput. Eng., Erciyes Univ., Kayseri, Turkey, Tech. Rep. TR06, Oct. 2005.

[25] H. Dhahri and A. M. Alimi, "Opposition-based particle swarm optimization for the design of beta basis function neural network," in *Proc. Int. Joint Conf. Neural. Netw.*, Barcelona, Spain, Jul. 2010, pp. 18–23.

[26] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition versus randomness in soft computing techniques," *Appl. Soft. Comput.*, vol. 8, no. 2, pp. 906–918, 2008.

[27] G. E. P. Box and G. M. Jenkins, *Time Series Analysis Forecasting and Control*. San Francisco, CA, USA: Holden Day, 1976.

[28] Y.-W. Chen, J.-B. Yang, D.-L. Xu, and S.-L. Yang, "On the inference and approximation properties of belief rule based systems," *Inf. Sci.*, vol. 234, pp. 121–135, Jun. 2013.

[29] S. Wadhawan, G. Goel, and S. Kaushik, "Data driven fuzzy modelling for sugeno and mamdani type fuzzy model using memetic algorithm," *Int. J. Inf. Technol. Comput. Sci.*, vol. 5, no. 8, pp. 24–37, 2013.

[30] H. Dhahri, A. M. Alimi, and A. Abraham, "Hierarchical particle swarm optimization for the design of beta basis function neural network," in *Intelligent Informatics*, vol. 182. Berlin, Germany: Springer-Verlag, 2013, pp. 193–205.

[31] B. Samanta, "Prediction of chaotic time series using computational intelligence," *Expert Syst. Appl.*, vol. 38, no. 9, pp. 11406–11411, Sep. 2011.

[32] D. Leite, F. Gomide, R. Ballini, and P. Costa, "Fuzzy granular evolving modeling for time series prediction," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Taipei, Taiwan, Jun. 2011, pp. 2794–2801.

[33] Y. Chen, B. Yang, J. Dong, and A. Abraham, "Time-series forecasting using flexible neural tree model," *Inf. Sci.*, vol. 174, nos. 3–4, pp. 219–235, 2005.

[34] E. de la Rosa, W. Yu, and X. Li, "Probability based fuzzy modeling," in *Proc. IEEE Int. Conf. Syst. Man, Cybern.*, Banff, AB, Canada, Oct. 2017, pp. 1633–1638.

[35] A. J. Izeman, "J. R. wolf and the Zürich sunspot relative numbers," *Math. Intell.*, vol. 7, no. 1, pp. 27–33, 1985.

[36] S. Yilmaz and Y. Oysal, "Fuzzy wavelet neural network models for prediction and identification of dynamical systems," *IEEE Trans. Neural Netw.*, vol. 21, no. 10, pp. 1599–1609, Oct. 2010.

[37] R. A. Aliev, B. G. Guirimov, B. Fazlollahi, and R. R. Aliev, "Evolutionary algorithm-based learning of fuzzy neural networks. Part 2: Recurrent fuzzy neural networks," *Fuzzy Sets Syst.*, vol. 160, no. 17, pp. 2553–2566, Sep. 2009.

[38] H. Dhahri, A. M. Alimi, and A. Abraham, "Designing beta basis function neural network for optimization using artificial bee colony (ABC)," in *Proc. Int. Joint Conf. Neural. Netw.*, Brisbane, QLD, Australia, Jun. 2012, pp. 1–7.

[39] S. Bouaziz, A. M. Alimi, and A. Abraham, "Evolving flexible beta basis function neural tree for nonlinear systems," in *Proc. Int. Joint Conf. Neural. Netw.*, Dallas, TX, USA, Aug. 2013, pp. 1–8.

[40] B. Subudhi and D. Jena, "A differential evolution based neural network approach to nonlinear system identification," *Appl. Soft. Comput.*, vol. 11, no. 1, pp. 861–871, 2011.

[41] H. Dhahri, A. M. Alimi, and A. Abraham, "Hierarchical multi-dimensional differential evolution for the design of beta basis function neural network," *Neurocomputing*, vol. 97, pp. 131–140, Nov. 2012.

[42] J. Pohjalainen, O. Räsänen, and S. Kadioglu, "Feature selection methods and their combinations in high-dimensional classification of speaker likability, intelligibility and personality traits," *Comput. Speech Lang.*, vol. 29, no. 1, pp. 145–171, 2015.

[43] *Kent Ridge Bio-Medical Data Set Repository*. [Online]. Available: http://archive.is/FjJT9

[44] S. Pang, I. Havukkala, Y. Hu, and N. Kasabov, "Classification consistency analysis for bootstrapping gene selection," *Neural Comput. Appl.*, vol. 16, no. 6, pp. 527–539, Oct. 2007.

[45] Z. Wang and V. Palade, "Building interpretable fuzzy models for high dimensional data analysis in cancer diagnosis," *BMC Genomics*, vol. 12, no. 2, p. S5, 2011.

[46] E. B. Huerta, B. Duval, and J.-K. Hao, "Gene selection for microarray data by a LDA-based genetic algorithm," in *Pattern Recognition in Bioinformatics*. Melbourne, VIC, Australia: Springer, Oct. 2008.

[47] S. Ayat and M. Rahi, "Application of ant colony algorithm and principal components analysis in the diagnosis of lung cancer," *J. Math. Comput. Sci.*, vol. 13, no. 4, pp. 343–352, 2014.

[48] N. Innocent and M. Kurian, "Cancer prediction based on gene expression data through association rule based classification and fuzzy rough set attribute reduction on information gain ratio," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 2, no. 4, pp. 42–46, Apr. 2014.

[49] Y. Tang, Y.-Q. Zhang, Z. Huang, and X. Hu, "Granular SVM-RFE gene selection algorithm for reliable prostate cancer classification on microarray expression data," in *Proc. IEEE Symp. Bioinf. Bioeng. (BIBE)*, Minneapolis, MN, USA, Dec. 2005, pp. 290–293.

YOSRA JARRAYA was born in Sfax, Tunisia, in 1987. She received the B.E. degree in computer science and the master's degree in new technologies of dedicated computer systems from the National Engineering School of Sfax (ENIS), Sfax, in 2011 and 2012, respectively.

She is currently pursuing the Ph.D. degree with the University of Sfax. She is a Research Member with the Research Groups in Intelligent Machine, ENIS. Her main research interests include evolving fuzzy systems, type-2 fuzzy systems, evolutionary computation, and evolutionary multi-objective optimization.

SOUHIR BOUAZIZ was born in Sfax, Tunisia, in 1984. She received the degree in computer engineering in 2008, and the Ph.D. degree in computer engineering from the National Engineering School of Sfax in 2014.

She is currently a Teaching Assistant with the National Engineering School of Gabes and a member of the Research Groups in Intelligent Machines. Her research interests include computational intelligence: neural network, evolutionary computation, and swarm intelligence.

ADEL M. ALIMI (SM'00) was born in Sfax, Tunisia, in 1966. He received the degree in electrical engineering in 1990, and the Ph.D. and HDR degrees in electrical and computer engineering in 1995 and 2000, respectively.

He is currently a Professor in electrical and computer engineering with the University of Sfax. His research interests include the applications of intelligent methods (neural networks, fuzzy logic, and evolutionary algorithms) to pattern recognition, robotic systems, vision systems, industrial processes, intelligent pattern recognition, learning, analysis, and intelligent control of large-scale complex systems.

Dr. Alimi is a member of IAPR, INNS, and PRS. He was the General Chairman of the International Conference on Machine Intelligence ACIDCA-ICMI'2005 & 2000. He is the 2009–2010 IEEE Tunisia Section Treasurer, the 2009–2010 IEEE Computational Intelligence Society Tunisia Chapter Chair, the 2011 IEEE Sfax Subsection, the 2010–2011 IEEE Computer Society Tunisia Chair, the 2011 IEEE Systems, Man, and Cybernetics Tunisia Chapter, the SMCS Corresponding Member of the IEEE Committee on Earth Observation, and the IEEE Counselor of the ENIS Student Branch. He is an Associate Editor and a member of the editorial board of many international scientific journals, including *Pattern Recognition Letters*, *Neurocomputing*, *Neural Processing Letters*, *International Journal of Image and Graphics*, *Neural Computing and Applications*, *International Journal of Robotics and Automation*, and *International Journal of Systems Science*. He was a guest editor of several special issues of international journals, including *Fuzzy Sets and Systems*, *Soft Computing*, *Journal of Decision Systems*, *Integrated Computer Aided Engineering*, and *Systems Analysis Modelling and Simulations*.