Sensitivity Versus Accuracy in Multiclass Problems Using Memetic Pareto Evolutionary Neural Networks

Juan Carlos Fernández Caballero, *Member, IEEE*, Francisco José Martínez, *Member, IEEE*, César Hervás, *Member, IEEE*, and Pedro Antonio Gutiérrez, *Member, IEEE*

***Abstract*—This paper proposes a multiclassification algorithm using multilayer perceptron neural network models. It tries to boost two conflicting main objectives of multiclassifiers: a high correct classification rate level and a high classification rate for each class. This last objective is not usually optimized in classification, but is considered here given the need to obtain high precision in each class in real problems. To solve this machine learning problem, we use a Pareto-based multiobjective optimization methodology based on a memetic evolutionary algorithm. We consider a memetic Pareto evolutionary approach based on the NSGA2 evolutionary algorithm (MPENSGA2). Once the Pareto front is built, two strategies or automatic individual selection are used: the best model in accuracy and the best model in sensitivity (extremes in the Pareto front). These methodologies are applied to solve 17 classification benchmark problems obtained from the University of California at Irvine (UCI) repository and one complex real classification problem. The models obtained show high accuracy and a high classification rate for each class.**

***Index Terms*—Accuracy, local search, multiclassification, multiobjective evolutionary algorithms, neural networks, sensitivity.**

I. Introduction

CLASSIFICATION is one of the most frequently encountered decision making tasks in human activity. A classification problem occurs when an object needs to be assigned into a predefined group or class based on a number of observed attributes related to that object. Many problems in business, science, industry, and medicine can be treated as classification problems [10], [37], [50]. Some examples include, among others, bankruptcy prediction [27], credit scoring [65], medical diagnosis [42], quality control [46], handwritten character recognition [24], and speech recognition [60]. The extension from two-class to a multiclass pattern classification problem is nontrivial, and often leads to unexpected complexity or weaker performances [25].

Manuscript received July 22, 2009; revised October 23, 2009, January 04, 2010, and January 19, 2010; accepted January 19, 2010. Date of publication March 11, 2010; date of current version April 30, 2010. This work was supported in part by the Spanish Inter-Ministerial Commission of Science and Technology (MICYT) under Project TIN 2008-06681-C06-03, the FEDER funds, and the P08-TIC-3745 Project of the "Junta de Andalucía" (Spain).

J. C. Fernández Caballero, C. Hervás, and P. A. Gutiérrez are with the Department of Computer Science and Numerical Analysis, University of Córdoba, 14071 Córdoba, Spain (e-mail: jcfernandez@uco.es; chervas@uco.es; pagutierrez@uco.es; i02gupep@uco.es).

F. J. Martínez is with the Department of Management and Quantitative Methods, ETEA, 14005 Córdoba, Spain (e-mail: fjmestud@etea.com).

Digital Object Identifier 10.1109/TNN.2010.2041468

A classifier design method is usually an algorithm that develops a classifier to approximate an unknown input–output mapping function from finitely available data, i.e., training samples. Once this classifier has been designed, it can be used to predict the class labels that correspond to unseen samples. Hence, the objective of developing a good classifier is to ensure high prediction accuracy for unseen future data, i.e., testing capability. Many techniques have been proposed to improve the overall testing capability of the classifier designed, but very few methods maintain this capability in all classes. This is very important in some data sets (such as medicine, remote sensing, economy, etc.) to ensure the benefits of one classifier over another.

Artificial neural networks (ANNs) [9] have been an object of renewed interest among researchers, both in statistics and computer science, owing to the significant results obtained in a wide range of classification and pattern recognition problems. Many different types of neural network architectures have been used, but the most popular one is multilayer perceptron (MLP). Recent vast research activities in neural classification have established that neural networks are a promising alternative to various conventional classification methods [75].

Learning algorithms for ANN models can be divided into two basic groups: local search algorithms and global search algorithms. The backpropagation algorithm (BP) belongs to the first group. Some methods require a topology to be preset by an expert and only modify the value of the coefficients while the topology changes through iterative processes of growth (constructive) or pruning (destructive). Evolutionary algorithms (EAs), which can evolve network models optimizing the architecture and weights individually or simultaneously [71], belong to the second group.

Often a great number of objectives must often be processed to obtain a viable solution to a problem, usually without any *a priori* knowledge of how the objectives interact with each other [40]. In this optimization process, within the field of ANNs, the objective of optimizing both the network model and the accuracy of the model (measured by the percentage of correctly classified patterns CCR or C) is usually to minimize the number of coefficients in the model to thus encourage more simple and more interpretable models which generally yield better results in accuracy for the testing data set.

There are different methods for optimizing two or more objectives. The most popular are the methods that use a regularization term or aggregating function (scalarized multiobjective learning), which aggregates a scalar objective function, and also the methods based on Pareto dominance [14], [16]. However,

the first case is not an efficient way to solve the problem. First, the scalarized multiobjective learning method can only generate one Pareto solution at a time and it assumes the convexity of the Pareto frontier. Second, choosing a suitable regularization term is occasionally a tedious problem of trial and error. This is particularly important if multiple objectives conflict with each other, and consequently, no single optimal solution is able to optimize all the objectives simultaneously.

An additional potential advantage of the Pareto-based learning approach is that using multiobjective techniques may help the learning algorithm to get out of local optima, thus improving the accuracy of the learning model. Some empirical evidence has been reported in [4] and [6].

The training of ANNs by evolutionary Pareto-based algorithms [40] known as multiobjective evolutionary artificial neural networks (MOEANNs), has been in use in the last few years to solve classification tasks [50], having some of its main exponents in [1] and [40]. The objective pursued when using multiobjective evolutionary algorithms (MOEAs) with ANNs is mainly the design of models with the greatest possible accuracy in a classification problem and with little structural complexity (this last objective defined as the squared sum of all the weights in the network, squared sum of the absolute value of the weights, the number of hidden neurons in the net or the number of links or connections in the network) [4], [11], [41].

The idea of designing neural networks within a multiobjective setup was first considered by Abbass [4]. In his work, the multiobjective problem formulation essentially involved the setting up of two objectives: the complexity of the network and the training error (quadratic error/mean square error). For it, an algorithm called MPANN which uses Pareto differential evolution [5] was proposed, showing improvements over many other MOEAs. MPANN was considered later in [3] for learning and to form ANN ensembles, albeit, with a different multiobjective formulation.

On the other hand, it is theoretically and empirically verified that the combination (ensemble) of the results obtained by different classifiers may improve the results that each provides individually [66]. The success of an ensemble depends basically on two factors related to the classifiers in it: accuracy and diversity. Both the quality of each individual classifier and the differences between them are necessary conditions for the effective functioning of the ensemble in which the classifiers are integrated. A recent algorithm called diverse and accurate ensemble learning algorithm (DIVACE) [12] makes use of good ideas found in negative correlation learning (NCL) [44] and MPANN [3]. Also DIVACE formulates the ensemble learning problem as a multiobjective problem explicitly within an evolutionary setup, aimed at finding a good tradeoff between diversity and accuracy. Other good studies about ensembles can be found in [38] and [39], where nondominated ANNs on the accuracy–complexity tradeoff surface were used to construct an ensemble classifier from the Pareto front obtained by MOEAS.

This type of model (ensembles) presents, in general, a good result on the testing set, because the usual effect of ensemble averaging is to reduce the variance of the error of a set of classifiers. They are accuracy oriented: their weighting strategy may bias towards the prevalent class since that contributes more to overall classification accuracy [64].

The objective of this paper is to propose a simple neural network model for classification based on a 2-D performance measure associated with multiclass problems. This is done because a simple model makes it easier to understand the cause–effect relationship between the input variables and the output classification target, e.g., by applying a rule-based system to the best neural network model [61]. Our proposal tries, unlike most works which seek high accuracy (the measure for the correct classification rate C), to achieve a high classification rate level in the testing data set with a good classification level for each class. Concretely, we consider the minimum of the sensitivities of all classes S, that is, the lowest percentage of examples correctly predicted as belonging to each class with respect to the total number of examples in the corresponding class and the traditionally used accuracy C. The sensitivity versus accuracy pair (S, C) expresses two features associated with a classifier: global performance C and the rate of the worst classified class S. These two objectives, after certain levels, are usually in conflict in the optimization process. This can be seen in Section V.

In our opinion, the results given in [48] show that sensitivity as an objective provides interesting ideas on creating models for pattern recognition. In this previous work, an approach is proposed to deal with multiclass classification problems using an EA for designing ANNs. From this perspective, we try to optimize accuracy and sensitivity with different strategies observing the behavior of different fitness functions such as accuracy, entropy, sensitivity, and area (a combination of accuracy and sensitivity) in an evolutionary neural network scheme. From this analysis, we presented a two-stage EA with entropy (first stage) and area (second stage) as fitness functions. The two-stage algorithm obtained promising results, achieving a high classification rate level in the global data set with an acceptable level of accuracy for each class. We also considered a mono-objective algorithm with only the sensitivity fitness function. The results obtained in the work just mentioned showed that the sensitivity fitness function S generally obtains classifiers with a better sensitivity level than the accuracy fitness function, although at a lower accuracy level. This behavior is especially significant when unbalanced data sets or multiclass problems with a high number of classes are considered. In summary, this previous work shows the importance of optimizing precision with sensitivity but does not really obtain a satisfactory result for both measures. In the present work, we show that a multiobjective approach is a much better option to optimize both accuracy and sensitivity. We build a much more robust algorithm that achieves an interesting result in this new area of dealing with multiclass problems.

So, to obtain classifiers with high accuracy C and good levels of sensitivity S, for all classes, this work presents an MOEA for classification with ANNs, concretely, a memetic version of NSGA2 introducing the *iRprop+* [34] algorithm as the local optimizer. The algorithm designs the ANN architecture, finding an adequate number of neurons in the hidden layer and number of connections along with the weights of the net. A network with a low number of neurons cannot generalize correctly due to a

learning disability, while a network with a large number of neurons increases the complexity of the model and overtrains the patterns.

This methodology is applied to solve 17n benchmark classification problems obtained from the University of California at Irvine (UCI) repository [8] and one real complex qualitative chemometric problem. The algorithm has been empirically analyzed, justifying the strategy with an MOEA based on Pareto dominance. Our approach is compared to two well-known Pareto-based multiobjective algorithms.

The rest of the paper is organized as follows. In Section II, accuracy and sensitivity measures are proposed and their properties are discussed. Section III presents an overview of multiobjective evolutionary neural networks. Section IV describes the MPENSGA2 algorithm and our problem is shown as a multiobjective Pareto-based optimization problem. Section V shows the experimental design, and finally, the conclusions are drawn in Section VI.

II. Accuracy and Sensitivity in Classification Problems

A. Accuracy

Statistics and machine learning communities have traditionally used overall correct classification or accuracy C to measure the performance of a classifier, generally avoiding the classification level of each class in the results. However, the pitfalls of using accuracy have been pointed out by several authors [56], [55]. Actually, it is enough to realize that accuracy cannot capture all the different behavioral aspects found in two different classifiers.

Assuming that all misclassifications are equally costly and that there is no profit gained by a correct classification, we assume that a good classifier should obtain both a high accuracy level and an acceptable level for each class. In real problems these objectives are usually in competition. Achieving a high overall classification accuracy level usually means sacrificing classification accuracy in one or more classes. This problem is especially significant when we deal with classification problems that differ in their prior class probabilities (class imbalances) or where there are a great number of classes [30]. It is clear that in many medical problems and imbalanced problems there is a cost associated with the classifier, so receiver operating characteristic (ROC) curves [20] are often used to check if one classifier is better than another in terms of the minority class.

When there are two classes, ROC curves are an alternative to accuracy. However, the standard ROC perspective is limited to classification problems with two classes. Extension of the standard two class ROC for multiclass problems (Q-classes) is attractive, since it would confer the benefits of ROC analysis to more problems in pattern recognition. Recently, a number of studies in this area have been performed. In [19], the approach considers a multiobjective optimization problem where the objective is to simultaneously minimize the $Q(Q-1)$ misclassification rates given by the off-diagonal elements of the confusion matrix. The main shortcoming of this approach is that unfortunately the dimension of the Pareto optimal fronts grows at the rate of the square of the number of classes. In [43], the authors simplify the multiclass ROC by means of an algorithm that analyzes the ROC dimensions that are independent of each other, identifying independent classes and groups of interacting classes, allowing the ROC to decompose. The resulting decomposed ROC hypersurface can be studied in the same way as the ideal case. In [49], the ROC concepts are extended to diagnostic enterprises with three possible outcomes, and a ROC surface on 3-D coordinates is plotted for a trichotomous decision task using the volume under the ROC surface (VUS). Thus, the VUS summarizes global diagnostic accuracy for trichotomous tests, just as the area under the ROC curve does for a two-alternative diagnostic task. Information gain at points on the surface can be calculated in the same way as for 2-D ROC curves, and researchers can thus compare three-way ROCs by comparing the maximum information gain on each ROC surface. On the other hand, Everson and Fieldsend [19] extend multiclass ROC analysis based on tradeoffs between the misclassification rates from one class to each of the other classes; the multiclass ROC surface is considered to be the solution to the multiobjective optimization problem when the objectives are these misclassification rates. Thus, the computational cost grows as a function of the number of classes.

Even though the ROC analysis can be theoretically extended to cover a multiclass case, in real cases, this extension presents difficulties. Unfortunately, the dimension of the Pareto optimal fronts grows at the rate of the square of the number of classes. Moreover, it is very difficult to obtain a good estimate of the Pareto front in the objectives or fitness function space when there is an increase in the number of objectives to be optimized. The Pareto fronts obtained will have a lot of individuals which increases the computational cost for finding each Pareto front. Finally, working with more than two objectives has the disadvantage of visualization in more than three dimensions.

B. Sensitivity–Accuracy Approach

Let us define a problem of classification with Q classes and N training or testing patterns. We define the performance of a classifier g by means of the corresponding $Q \times Q$ contingency or confusion matrix $M(g)$

$$M(g) = \left\{ n_{ij}; \sum_{i,j=1}^{Q} n_{ij} = N \right\}$$

where n_{ij} represents the number of times that the patterns are predicted by classifier g to be in class j when they really belong to class i. The diagonal corresponds to correctly classified patterns and the off-diagonal corresponds to mistakes in the classification task.

Let us denote the number of patterns associated with class i by

$$f_i = \sum_{j=1}^{Q} n_{ij}, \qquad i = 1, \ldots, Q.$$

Two scalar measures can be defined in order to take the elements of the confusion matrix into consideration from different points of view. Let $S_i = n_{ii}/f_i$ be the number of patterns correctly predicted to be in class i with respect to the total number

of patterns in class i (sensitivity for class i). Therefore, the sensitivity for class i estimates the probability of correctly predicting a class i example.

From the above quantities, we define the sensitivity S of the classifier as the minimum value of the sensitivities for each class $S = \min(S_i; i = 1, \ldots, Q)$.

The correct classification rate or accuracy C is defined as

$$C = (1/N) \sum_{i=1}^{Q} n_{ii}$$

that is, the rate of all the correct predictions.

Specifically, it is the 2-D measure (S, C) associated with a given classifier g that is considered in this work. This measure tries to evaluate two features of the classifier: the global performance in the whole data set and the performance in each class.

The selection of S as a complementary measure of C can be justified upon considering that

$$C = \sum_{i=1}^{Q} \frac{f_i}{N} S_i$$

is the weighted average of the sensitivities in which the weights depend on the data set, thus providing both intuitive and computational support of the sensitivities of each one of the Q classes. C is estimated based on fixed weight f_i, and it is the choice of t that minimizes

$$\sum_{i=1}^{Q} f_i (S_i - t)^2.$$

From a statistical point of view, since it is a weighted average, C will be a good and representative W-estimator of location of the set of sensitivities if they are homogeneous [31]. On the other hand, C will not be a representative measure if the sensitivities are very scattered. Keeping this fact in mind, the complementation of the information contributed by C to a classifier could be obtained by considering some measure that minimizes dispersion. The range $R = \max\{S_i\} - \min\{S_i\}$ is a possible choice. Its minimization may be reached either minimizing $\max\{S_i\}$ or maximizing $\min\{S_i\}$. Since the first option should be ruled out considering that C increases if the sensitivities increase, the second will be the appropriate choice. Therefore, $S = \min\{S_i\}$ can be considered a complementary measure of C whose value must be maximized. It will improve C as a weighted average of the correct classification rates of the Q classes. This perspective involves two objectives that have not been used previously in the design of ANN models although they are equivalent (subsets) to those points on the $Q(Q-1)$ surface which trade off the smallest total error, sum of the $Q(Q-1)$ elements, against the worst misclassification rate for any class (i.e., the largest sums in the rows of the $Q(Q-1)$ off-diagonal elements in the confusion matrix) [19]. Then, in a Pareto sense, therefore, a 2-D line is traced out on the $Q(Q-1)$ surface.

It is clear that two quantities cannot collect all the information given by the $Q(Q-1)$ misclassification rates included in the confusion matrix. However, the (S, C) pair tries to avoid the disadvantages of the approaches based on $Q(Q-1)$ misclassification rates (i.e., the difficulties for a graphic representation

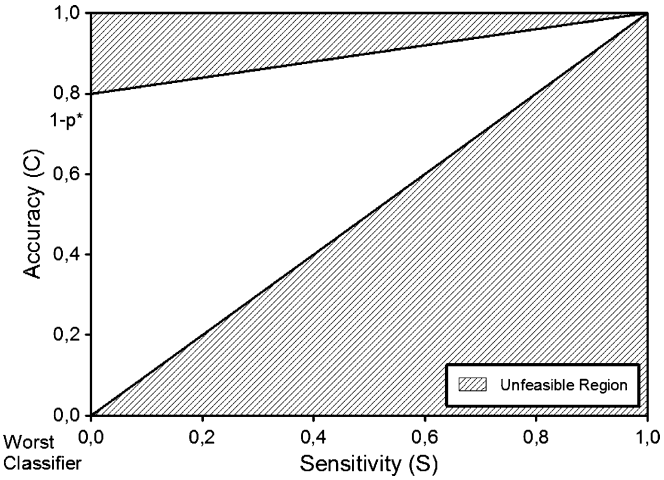

Fig. 1. Unfeasible region in the 2-D (S, C) space for a given classification problem.

that would allow us to visualize the performance of the classifiers; the increase in the dimension of the Pareto front with respect to the number of objectives, in our case, the $Q(Q-1)$ misclassification rates, and the computational problem associated with a multiobjective optimization problem which presents a lot of objectives). In this way, the (S, C) pair tries to find a point between the scalar accuracy measure and the multidimensional ones based on misclassification rates. These two measures verify that

$$S \leq C \leq 1 - (1 - S)p^* \tag{1}$$

being p^* the minimum of the estimated *prior* probabilities, value that has an important role in the relationship between the two measures.

Sensitivity S is represented on the horizontal axis and accuracy C on the vertical axis. One point in (S, C) space dominates another if it has more accuracy and equal or greater sensitivity, or if it has greater sensitivity and equal or better accuracy. Therefore, ideally, from the previous inequality of (1), each classifier will be represented as a point in the white region in Fig. 1, hence the area outside the triangle is marked as unfeasible. The area inside the triangle may be feasible (attainable), or may not be, depending upon the classifier and the difficulty of the problem. Observe that the optimum classifier $(C = 1, S = 1)$ is not feasible for all problems/classifiers, especially for problems with stochastic elements. For this reason, it is better to say that a classifier cannot be located in the unfeasible region. Furthermore, the points on the vertical axis correspond to classifiers that are not able to correctly predict any pattern of a given class. Note that it is possible to find among them classifiers with a high level of accuracy, particularly in problems with low p^* (unbalanced problems).

The following comments can be made from the concrete shape of the region. First, observe that an increase in accuracy C does not imply an increase in sensitivity S. Reciprocally, an increase in sensitivity S does not mean an increase in accuracy C. On the other hand, it should be noted that for a fixed value of accuracy C, a classifier will be better when it corresponds to a point nearer to the diagonal of the square.

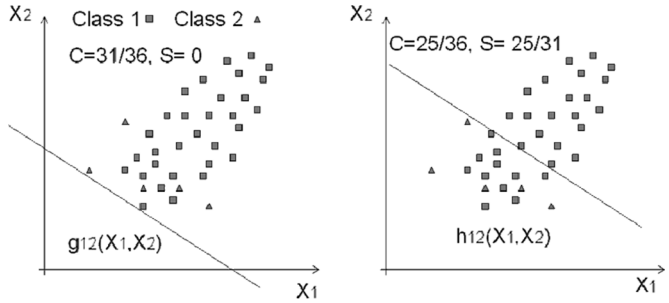

Fig. 2. Accuracy and sensitivity as conflicting objectives—an illustration.

It is important to note that sensitivity S and accuracy C are not cooperative in general, especially at certain levels. At the beginning of a learning process, accuracy and sensitivity could be cooperative, but after a certain level, objectives become competitive and an improvement in one objective tends to involve a decrease in the other one. The example, Fig. 2, shows that accuracy and sensitivity are conflicting objectives in general. For that unbalanced two-class example, in the left graph, the linear classifier obtains $C = 31/36$, and $S = 0$. If we want to improve the sensitivity, the decision boundary should be moved to separate the triangle class from the square one but, and in general because of that, it is necessary to reduce accuracy.

When the number of classes increases or when the problem is highly unbalanced, the quantity $p^* = f_j/N \leq 1/Q$ decreases in both cases, and the straight line which defines the upper border of the feasible set in (S, C) space tends to be horizontal because (1) tends to $S \leq C \leq 1$, thus the range of S values will be large even for high values of C. In these conditions, the use of C as the only measure is inadequate as it hides many different possibilities for S. These comments show that the sensitivity measurement versus that of accuracy can be especially useful in problems concerning unbalanced classes or when there are a great number of classes, and confirm, again, the inadequacy of accuracy in these situations.

III. Multiobjective Evolutionary Neural Networks

Evolutionary artificial neural networks (EANNs) have been a key research area for the last few years [72]. Methods and techniques have been developed to find better approaches for evolving ANNs, trying to design networks with good generalization capability. On the other hand, finding a good ANN architecture has also been a debatable issue in the field of ANNs. Methods for network growing, denominated constructive algorithms [52], start with a small network, usually a single neuron. This network is trained until it is unable to continue learning, and then new components are added to the network. This process is repeated until a satisfactory solution is found. Destructive methods, also known as pruning algorithms [57], start training with a larger network than necessary, and then remove unnecessary parts. The reason is that the large-sized network can learn reasonably quickly with less sensitivity to initial conditions while the reduced complexity of trimmed networks shows improved testing performance. However, all these methods still suffer from their slow convergence and long training time. In addition, they are based on gradient-descent techniques and, therefore, can easily get stuck at a local minimum, as the BP algorithm and its variants do.

The uses of evolutionary approaches in ANN training (with a single error function) have a number of intuitive advantages over gradient-descent training in this domain. A major advantage of the evolutionary approach over traditional learning algorithms, like BP, is its ability to escape a local optimum, its robustness, and its ability to adapt itself to a changing environment. Training ANNs with EAs is a powerful approach to address the exploitation/exploration dilemma. Selecting the size of the architecture of a neural network for a particular application is a difficult task. The architecture of the neural network directly affects two of the most important factors of neural network training: testing and training efficiency and efficacy.

EANNs can automatically find good architectures (a number of hidden neurons and weight values that do not produce overtraining or lack of learning in testing) for a neural network through evolutionary process and through crossover and mutation operators designed for this purpose. The evaluation of each individual in the population of each generation simulates the process of natural selection. There have been many applications for parametric learning (evolving the weights of the network) [58] and for both parametric and structural learning (evolving the weights and the number of hidden nodes and connections simultaneously) [7], [73]. On the other hand, Palmes *et al.* [51] classified two major types of EANNs: noninvasive and invasive. The first one refers to approaches where evolutionary selection is used but some gradient training is required for fitness evaluation; while an invasive approach tries to optimize weights and architecture in the evolution process. The authors point out that the invasive approach is better for the generation of efficient networks, avoiding gradient-based fitness evaluation, and resulting in more robust search coverage. This could indicate the appropriate selection of that methodology for training neural networks. Yao [70] presents a thorough review of the field of EANNs. Differential evolution (DE) is a branch of EAs developed by Storn and Price [63] for optimization problems over continuous domains. DE has been used in the last few years to design ANNs. For example, Ilonen *et al.* [36] analyzed DE as a candidate global optimization method for feedforward neural networks as compared to gradient-based methods, and designed ANNs using mean square error as the objective function.

On the other hand, Pareto-based techniques, specifically MOEAs, should provide a well distributed nondominated front and provide diversity (in the objective space) to explore the fitness landscape, although it is difficult to define appropriate quality in a Pareto front [14]. Also, these techniques present an uncountable set of solutions that, when evaluated, produce vectors whose components represent a tradeoff in an objective space. A decision maker then implicitly chooses an acceptable solution (or solutions) by selecting one or more of these vectors.

There are a limited number of studies using an MOEA to train a population of multiobjective ANNs which is usually applied to minimize the error in the training set and the complexity of the network. For example, Abbass has presented different studies on the design and training of ANNs using accuracy

and complexity as objectives [4], [5]. Along with Abbass, other authors use DE as the basis of their MOEAs [54], [74], but the most current MOEAs are not based on DE and are being used to design ANN models, with multiple applications and uses for them. The reader may refer to [21], [22], [28], and [59] for more details and examples about these studies based on accuracy complexity.

An improvement of the EA, both mono-objective and multiobjective based or not on Pareto dominance, is the incorporation of local search procedures throughout the evolution. Some studies carried out on the process of the convergence of an EA in a concrete optimization problem show how the genetic algorithm quickly finds acceptable solutions for the problem. However, it needs many generations to reach a good solution and they are too poor to find the best solution when they are in a region where the algorithm converges toward a global optimum [33]. On the other hand, it is well known that certain local procedures are able to find the global optimum when the search is carried out in a small region of the space. Therefore, in the combination of EAs and local procedures, EAs would carry out a global search inside the space of solutions, locating the ANNs near the global optimum, so the local procedure could arrive at the best solution quickly and efficiently. This type of algorithm receives the name of memetic or hybrid algorithm [62], [69].

Recently, new methods have been developed in order to improve the lack of precision of the EAs using local optimization algorithms [15]. Gradient-descent techniques are the most widely used class of local search algorithms for supervised learning in neural networks. The resilient backpropagation algorithm (*Rprop*) [34] is the best of these techniques known to the authors in terms of convergence speed, accuracy and robustness with respect to its parameters, although classic algorithms like BP are also frequently used. Recently, the improved resilient backpropagation (*iRprop*) algorithm has been proposed, which applies a backtracking strategy (i.e., it decides whether to take a step back along a weight direction, by means of a heuristic). It has been shown in several benchmark problems [35] that the improved *Rprop* consistently exhibits better performance than the original *Rprop* algorithm, and that is why we use a variant of the same algorithm in this work.

There are several studies that make use of MOEAs along with local optimizers, used after the mutation phase, to fine tune the weights. After learning, the fitness of each individual is updated with respect to the approximation error. This is called "lifetime learning." In addition, the modified weights are encoded back to the chromosome during lifetime learning. This is known as the Lamarckian type of inheritance. Jin *et al.* presented [41] a comparison of the results obtained with NSGA2 algorithm along with *Rprop* in several works [38], and mono-objective algorithms with a weighted sum of objectives, trying to minimize the mean square error (MSE) and the complexity of the network in terms of the number of links and hidden layer neurons. In [2], Abbass obtains neural network models for the diagnosis of cancer and other data sets from the UCI, using the BP algorithm as the local procedure and using, again, accuracy–complexity as objectives. In [26], MOEAs are used to classify human faces and cars by using the NSGA2 algorithm with the introduction of a local search in each generation. The MSE and the number of neurons in the hidden layer are minimized to obtain the classifiers. This is carried out with the improved *Rprop* algorithm although it involves a high computational cost. We wanted to avoid a high computational cost in this paper so the local search was introduced in a very small number of generations. As can be seen, MOEAs generally used for designing ANN models for classification try to minimize the error and minimize the complexity, but in no case is sensitivity used to improve a classifier when there are more than two classes (multiclassification problems). In this paper, we propose a memetic MOEA to design ANN models for multiclass problems [50], specifically the NSGA2 used in combination with the *iRprop* algorithm. This local search will improve the Pareto front obtained in only one objective, specifically in the direction of the objective that tries to minimize the classification error, which in this case is the cross-entropy error function (presented in the next section).

IV. The MPENSGA2 Memetic Multiobjective Evolutionary Algorithm

A. Base Classifier Framework

We consider that the most widely used models of ANNs are the multilayer perceptron (MLP) neural networks. The functional model considered is the following:

$$f_l(\mathbf{x}, \boldsymbol{\theta}_l) = \beta_0^l + \sum_{j=1}^{M} \beta_j^l \sigma_j(\mathbf{x}, \mathbf{w_j}), \qquad l = 1, 2, \ldots, Q$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_J)^T$ is the transpose matrix containing all the neural net weights, $\boldsymbol{\theta}_l = (\beta_0^l, \beta_1^l, \ldots, \beta_M^l, \mathbf{w}_1, \ldots, \mathbf{w}_M)$ is the vector of weights of the l output node, $\mathbf{w}_j = (w_{1j}, \ldots, w_{Kj})$ is the vector of weights of the connections between input layer and the jth hidden node, Q is the number of classes of the problem, M is the number of sigmoidal units in the hidden layer, K is the number of features in each pattern to be classified, $\mathbf{x}$ is the input pattern, and $\sigma_j(\mathbf{x}, \mathbf{w}_j)$ has the following expression:

$$\sigma_j(\mathbf{x}, \mathbf{w}_j) = \frac{1}{1 + \exp\left(-\left(w_{0j} + \sum_{i=1}^{K} w_{ij} x_i\right)\right)}.$$

The outputs of the neural network can be seen from a probabilistic point of view, which considers the softmax activation function given by the following expression:

$$g_l(\mathbf{x}, \boldsymbol{\theta}_l) = \frac{\exp f_l(\mathbf{x}, \boldsymbol{\theta}_l)}{\sum_{j=1}^{Q} \exp f_j(\mathbf{x}, \boldsymbol{\theta}_j)}, \qquad l = 1, 2, \ldots, Q \quad (2)$$

where $g_l(\mathbf{x}, \boldsymbol{\theta}_l)$ is the probability pattern $\mathbf{x}$ has of belonging to class l. We adopt the common technique of representing the class levels using a "1-of-Q" encoding vector $\mathbf{y} = (y^{(1)}, y^{(2)} \ldots, y^{(Q)})$, such as $y^{(l)} = 1$ if $\mathbf{x}$ corresponds to an example belonging to class l, and $y^{(l)} = 0$ otherwise. Taking this consideration into account, it can be seen that the class predicted by the classifier corresponds to the neuron on the

1. t=0 and generate a random population P(t) of size N_p, where each individual presents a sigmoidal basis function structure and where "t" is the number of the actual generation.
2. Evaluate the individuals P(t) on basis of Entropy and Sensitivity.
3. Use the fast-non-dominated-sort for obtaining a list F with the fronts of the population P(t).
4. Assign to each individual a rank value equal to his dominance level and a crowding distance value for the case of tie in the following selection process.
5. Use binary tournament for selecting N_p individuals of P, according to their rank and crowding distance value
6. Do mutation (one of the five mutations randomly) on each of the individuals selected to generate a new offspring population Q(t) of size N_p.
7. Evaluate the individuals Q(t) on basis of Entropy and Sensitivity.
8. While stopping criterion is not met do
   a) f=1
   b) R(t) = P(t) U Q(t).
   c) F= fast-non-dominated-sort(R(t)).
   d) If (number of generation t is equal to 2/7 or 4/7 or 6/7 of the total number of generations)
      e) Apply iRprop+ to the individuals of the first Pareto front F^l of F and evaluate the individuals F^l on basis of Entropy and Sensitivity, R'(t) being the R(t) population with the first front modified.
      f) F= fast-non-dominated-sort(R'(t))
   g) end if
   h) While size of population P(t+1) is < N_p to do
      I. Calculates crowding-distance for front F^f
      II. P(t+1) = P(t+1) U F^f
      III. f=f+1 //The number (rank from the fast-non-dominated-sort) of Pareto front is incremented
   i) end while
   j) Sort population P(t+1) according to their rank and crowding value and select the fisrt N_p individuals. The new population P(t+1) of size N_p is now completed.
   k) Use binary tournament and according to their rank and crowding value obtain N_p individuals from P(t+1).
   l) Do mutation (one of the five mutations randomly) on each of the individuals selected by binary tournament to generate a new population Q(t+1) of size N_p.
   m) Evaluate the individuals Q(t+1) on basis of Entropy and Sensitivity.
   n) t=t+1
9. end while

Fig. 3. MPENSGA2 algorithm pseudocode. Local search in italic face.

output layer whose output value is the greatest. The optimum rule $C(\mathbf{x})$ is the following:

$$C(\mathbf{x}) = \hat{l}, \qquad \text{where} \quad \hat{l} = \arg\max_l g_l(\mathbf{x}, \boldsymbol{\theta}_l), \quad \text{for } l = 1, 2, \ldots, Q$$

Therefore, it is probable that one of the classes does not need to be estimated because of the normalization condition. With no loss in generality we can assume $f_Q(\mathbf{x}, \boldsymbol{\theta}_Q) = 0$, which implies a reduction of one node in the output layer of the neural network equal to $Q - 1$, where Q is the number of classes.

B. Fitness Functions

In order to establish a measure for determining the goodness of the MLP models, two functions have been considered: accuracy and the cross-entropy error function. If we use the training data set $D = \{(\mathbf{x}_n, \mathbf{y}_n); 1 \leq n \leq N\}$ then accuracy C is given by

$$C = \frac{1}{N} \sum_{n=1}^{N} I\left(C(\mathbf{x}_n) = \mathbf{y}_n\right)$$

where $I(.)$ is the zero-one loss function and N is the number of patterns in the data set. A good classifier tries to achieve the highest possible C in a given problem for the testing set. The other function used in this research to evaluate a classification model is the cross-entropy error E [9] and is given by the following expression for Q classes:

$$E(g, \boldsymbol{\theta}) = -\frac{1}{N} \sum_{n=1}^{N} \sum_{l=1}^{Q} y_n^{(l)} \log g_l(\mathbf{x}_n, \boldsymbol{\theta}_l). \tag{3}$$

The advantage of using the error function $E(g, \boldsymbol{\theta})$ instead of $(1 - C)$ is that this is a continuous function, which makes the convergence more robust. The first fitness measure to maximize is a strictly decreasing transformation of the entropy error $E(g, \boldsymbol{\theta})$ given by

$$A_1(g, \boldsymbol{\theta}) = \frac{1}{1 + E(g, \boldsymbol{\theta})}$$

where g is the function

$$g(\mathbf{x}, \boldsymbol{\theta}) = \left(g_1(\mathbf{x}, \boldsymbol{\theta}_1), \ldots, g_Q(\mathbf{x}, \boldsymbol{\theta}_Q)\right).$$

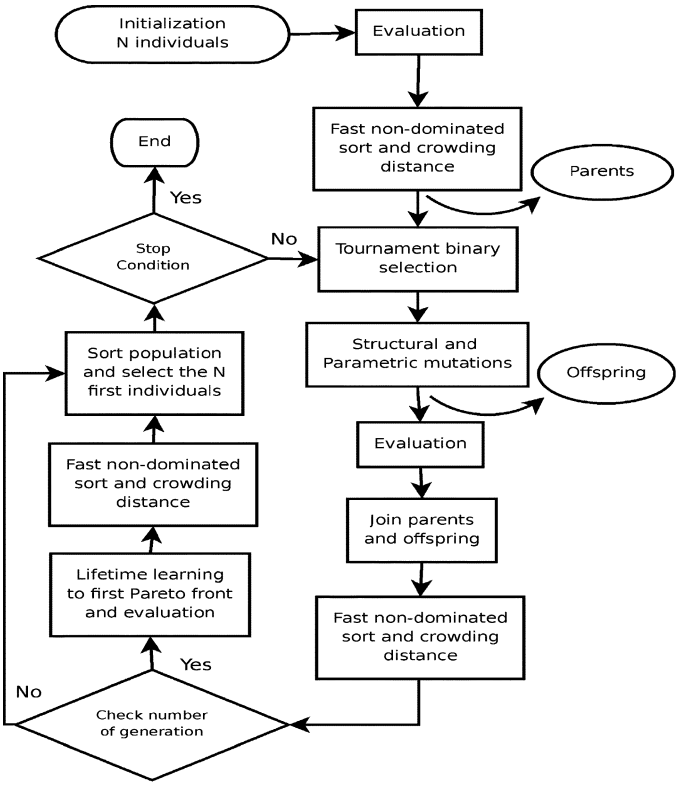

Fig. 4. General framework for MPENSGA2.

The second fitness measure to maximize is the minimum sensitivity S of the classifier

$$A_2(g) = S(g).$$

This objective, *a priori*, could be considered to be in no conflict with C, and to share a relationship of mutual dependency with it. However, it has been experimentally verified that they are in conflict depending on the data sets as the levels of accuracy achieved by the classifiers. The tradeoff situation between them can, therefore, be represented as a Pareto front [14], [16].

C. Mutations

We consider two types of mutation operators (structural and parametric) to obtain new individuals in the evolutionary process. These mutations are similar to those defined in the GNRL model [7]. The mutation operators inject new genetic material into a population of individuals, thereby ensuring that a larger part of the search space is covered.

There are five mutations, four structural mutations, and one parametric mutation, and the probability of choosing a type of mutator and applying it to an individual is equal to 1/5.

Parametric mutation [45] is done for each weight w_{ji} of the neural network with Gaussian noise $w_{ji}(t+1) = w_{ji}(t) + \xi(t)$ where $\xi(t) \in N(0, T(t))$, represents a 1-D normally distributed random variable with mean 0 and variance $T(t)$, and $T(t)$ represents a temperature in descent throughout the evolutionary process (descent value is configurable in the algorithm), making abrupt changes at the beginning (exploration) and smaller changes at the end (exploitation).

Structural mutation introduces diversity in the population that leads to different locations in the search space. Specifically, the operators used are add/delete neurons and add/delete connections. With regard to the mutations add or delete links, the number of links to add or delete are applied between the input layer and the hidden layer and between the hidden layer and the output layer. Specifically, we randomly add or delete 30% of the total number of links in the input-hidden layers, and 5% of the total in the hidden-output layers. Weights are assigned using uniform distribution defined throughout two intervals, $[-5, 5]$ for connections between the input layer and hidden layer and $[-10, 10]$ for connections between the hidden layer and the output layer. These values have been obtained experimentally and are sufficiently robust.

If the structural mutation that has been assigned randomly cannot be applied to a network, a new mutation is chosen. This can occur in cases in which we try to add or delete a neuron in the hidden layer when the maximum or minimum set value is reached (these values are dependent on the problem and are obtained experimentally by trial and error). This can also occur when trying to delete a link when it is the only one between two neurons, or if the elimination of a link in a hidden layer leaves a neuron unlinked to the output layer.

For further details about these mutations and about the generation of the neural networks in the beginning and during the evolutionary process, the reader can consult previous works [47].

D. iRprop+ Local Optimizer

The *Rprop* algorithm is believed to be a fast and robust learning algorithm. It employs a sign-based scheme to update the weights in order to eliminate harmful influences of the derivatives' magnitude on the weight updates, i.e., the magnitude of the update along each weight direction only depends on the sign of the corresponding derivative. The size of the update step along a weight direction is exclusively determined by a weight-specific "update-value" $\Delta_{ij}^{(t)}$. The *improved Rprop* algorithm (denoted by *iRprop+*) applies a backtracking strategy (i.e., it decides whether to take a step back along a weight direction or not by means of a heuristic). The idea of this modification is that weight updates causing changes in the signs of corresponding partial derivatives should be reverted only in case of error increase. Thus, this training scheme combines the local information (i.e., the sign of the partial derivative of the error with respect to a weight like *Rprop*) with more global information (i.e., the error value at each iteration) in order to decide whether to revert an update step for each weight individually. It has been shown in several benchmark problems in [35] that the *iRprop+* exhibits consistently better performance than the *Rprop* algorithm.

We have carried out the adaptation of the *iRprop+* local optimizer to the softmax activation function (2) and the cross-entropy error function (3). In this case, the gradient vector is given by

$$\nabla E(\boldsymbol{\beta}^l, \mathbf{w}) = \left(\frac{\partial E}{\partial \boldsymbol{\beta}^l}, \frac{\partial E}{\partial \mathbf{w}} \right).$$

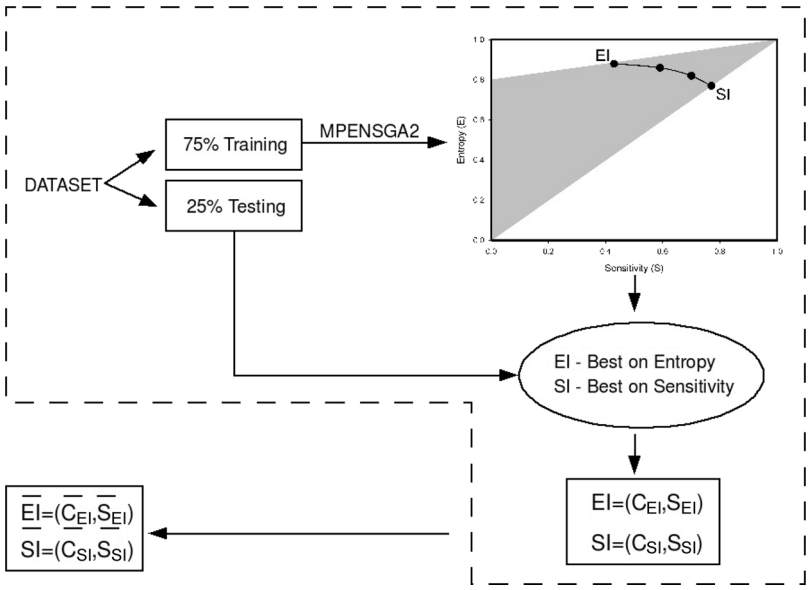

Fig. 5. Achieving statistical results from MPENSGA2.

Let $\boldsymbol{\theta}_l$ be any of the parameters of $\boldsymbol{\beta}^1$ and $\mathbf{w}$, being therefore

$$\frac{\partial E}{\partial \boldsymbol{\theta}_l} = \frac{1}{N}\sum_{n=1}^{N}\sum_{l=1}^{Q} y_n^{(l)} \frac{1}{g_l(\mathbf{x_n}, \theta_l)} \frac{\partial g_l(\mathbf{x_n}, \boldsymbol{\theta}_l)}{\partial \boldsymbol{\theta}_l}$$

$$\frac{\partial g_l(\mathbf{x_n}, \boldsymbol{\theta}_l)}{\partial \boldsymbol{\theta}_l} = \frac{1}{\left(1+\sum_{l=1}^{Q-1} e^{f_l}\right)^2} \times \left\{ e^{f_l}\frac{\partial f_l}{\partial \boldsymbol{\theta}_l}\left(1+\sum_{l=1}^{Q-1} e^{f_l}\right) - e^{f_l}\sum_{r=1}^{Q-1} e^{f_l}\frac{\partial f_l}{\partial \boldsymbol{\theta}_l}\right\}$$

$$\frac{\partial g_l(\mathbf{x_n}, \boldsymbol{\theta}_l)}{\partial \boldsymbol{\theta}_l} = g_l \frac{\partial f_l}{\partial \boldsymbol{\theta}_l} - g_l^2 e^{-f_l}\sum_{l=1}^{Q-1} e^{f_l}\frac{\partial f_l}{\partial \boldsymbol{\theta}_l}.$$

Finally, we have the following expressions for the output layer:

$$\frac{\partial f_l}{\partial \beta_0^k} = \delta_{kl} = \begin{cases} 0, & k \neq l \\ 1, & k = l \end{cases}, \qquad \frac{\partial f_l}{\partial \beta_s^k} = \begin{cases} 0, & l \neq k \\ \sigma\left(\sum_{i=1}^{K} w_{is} x_i\right), & l \neq k \end{cases}$$

and for the hidden layer

$$\frac{\partial f_l}{\partial w_s^t} = \beta_s^l \sigma'\left(\sum_{i=1}^{k} w_{is} x_i\right) x_t,$$
$$\text{where} \quad s = 1, 2, \ldots, m, \qquad t = 1, 2, \ldots, k$$

E. MPENSGA2

This section describes an MOEA for designing neural network models applied to multiclassification problems, with a local search procedure, called memetic Pareto evolutionary NSGA2 (MPENSGA2). This algorithm obtains different nondominated sets with classifiers that present a good balance between accuracy and sensitivity. We select the first nondominated set (see process in NSGA2 [17] for obtaining Pareto fronts) that is obtained in the objectives space (fitness functions space).

Our approach evolves architectures and connection weights simultaneously, each individual being a fully specified ANN. The ANNs are represented using an object-oriented approach and the algorithm deals directly with the ANN phenotype. Each connection is specified by a binary value indicating if the connection exists and a real value representing its weight. As crossover is not considered due to its potential disadvantages in evolving artificial networks [7], [72], this object-oriented representation does not assume a fixed order between the different hidden nodes.

The MOEA starts by generating a random population $P(0)$ of size N_p. The population is sorted on nondomination, assigning to each solution a fitness (or rank) equal to its nondomination level (1 is the best level, 2 is the next-best level, and so on). Then, the usual binary tournament selection and mutation operators are used to create an offspring population $Q(0)$ of size N_p. Since elitism is introduced by comparing the current population with previously found best nondominated solutions, the procedure is different after the initial generation. In Fig. 3, we show the pseudocode of the MPENSGA2 algorithm. The reader can see [17] to compare our algorithm to the original NSGA2 proposed by Deb *et al.*.

A local search procedure is applied when we combine the parents and offspring populations in NSGA2. Then, only the individuals from the first Pareto front (obtained from the fast nondominated sort) of this combined population are optimized by *iRprop+* [34], considerably reducing the computational cost since the local procedure is not applied to the whole mutated offspring population. *iRprop+* can be seen as a kind of lifetime learning (only for the Entropy error function) within a generation. After learning, the fitness of each individual is updated with regard to the approximation error. In addition, the weights

TABLE I
CHARACTERISTICS OF UCI BENCHMARKS AND A QUALITATIVE ANALYTICAL CHEMISTRY DATA SET

Dataset	#Patterns	#Training patterns	#Test patterns	#Input variables	#Classes	#Patterns per class	p^*
TWO CLASSES							
AustralianC	690	517	173	51	2	307-383	0.4411
BreastC	286	215	71	15	2	201-85	0.2957
BreastC-W	699	524	175	9	2	458-241	0.3428
German	1000	750	250	61	2	700-300	0.3
HeartStatlog	270	202	68	13	2	150-120	0.4411
Ionosphere	351	263	88	34	2	126-225	0.3636
Pima	768	576	192	8	2	500-268	0.3489
Vote	435	326	109	16	2	267-168	0.3853
MULTICLASS							
Autos	205	152	53	72	6	67-3-22-54-32-27	0.0188
Balance	625	469	156	4	3	288-49-288	0.0641
BTX	63	42	21	3	7	9-9-9-9-9-9-9	0.1428
Gene	3175	2381	794	120	3	762-765-1648	0.2405
Iris	150	111	39	4	3	50-50-50	0.3333
Lymphography	148	111	37	38	4	2-81-61-4	0.0270
Newthyroid	215	161	54	5	3	150-35-30	0.1296
Post-operatory	90	67	23	20	3	2-24-64	0.0434
Vowel	990	737	253	10	11	90-90-90-90-90-90-90-90-90-90-90	0.0909
Yeast	1484	1112	372	8	10	30-20-429-244-163-51-44-35-5-463	0.0026

modified during lifetime learning are encoded back to the chromosome, which is known as the Lamarckian type of inheritance [68]. Lifetime learning is done at the beginning, in the middle of the evolution and at the end of the evolutionary process, that is, only three local search procedures are carried out throughout the evolutionary process. In Fig. 4, we can observe the framework used in this paper, and in [40], we can see the general framework of different Pareto-based multiobjective work with ANNs.

V. EXPERIMENTS

A. Data Sets

For the experimental design, we consider 17 data sets taken from the UCI repository [8] and a qualitative analytical chemistry data set outside the UCI repository which is not commonly used, called BTX.

The design was conducted using a stratified holdout procedure (see [53]) with 30 runs, where approximately 75% of the patterns were randomly selected for the training set and the remaining 25% for the test set.

Table I shows the features for each data set. We have divided the table into two sets. The first set consists of binary classification problems (two classes), while the second set consists of multiclass problems. This division has been made since many classification algorithms present a lower performance when they are applied to multiclass problems because, among other reasons, the higher the number of classes, the lower the p^* value is. The table shows the total number of instances in each data set, the number of instances in training and testing, the number of input variables, the number of classes (outputs), the total number of instances per class, and the p^* value.

The BTX data set must be specially mentioned due to current interest in applications to real problems. The BTX is a classification problem consisting of discriminating different types of drinking waters. The data set includes a set of 63 drinking water samples spiked with individual standards of benzene, toluene, or xylene as well as with binary or ternary mixtures of them at concentrations between 5 and 30 μg/L. This constitutes an overall data set composed of seven different classes of contaminated drinking water samples with the same number of patterns per class. The reader can find more information about this problem in [29].

The experiments were performed with the same data sets and training and testing partitions for all algorithms.

B. Experimental Setup

Once the Pareto front is built, we use two strategies or automatic selection methodologies of individuals: the best model in accuracy and the best model in sensitivity (extremes in the Pareto front). This allows us to compare the C and S values with other classification methods found in the literature. The process of obtaining these models is as follows (see Fig. 5): as our procedures are stochastic, the MPENSGA2 algorithm is run 30 times. In each execution, once the first Pareto front is calculated, we chose the extreme values in training, that is, the best individual in entropy and the best individual in sensitivity. These individuals are called EI (entropy individual) and SI (sensitivity individual). Once this is done, we get the values of accuracy C and sensitivity S in the testing of the best individuals EI and SI. Therefore, we will have an individual performance $EI_{\text{testing}} = (C_{EI\_\text{testing}}, S_{EI\_\text{testing}})$ and another individual performance $SI_{\text{testing}} = (C_{SI\_\text{testing}}, S_{SI\_\text{testing}})$ for one run. This is repeated 30 times and then the average and standard deviation obtained from the 30 individuals is calculated, obtaining $\overline{EI}_{\text{testing}} = (\overline{C}_{EI\_\text{testing}}, \overline{S}_{EI\_\text{testing}})$ and $\overline{SI}_{\text{testing}} = (\overline{C}_{SI\_\text{testing}}, \overline{S}_{SI\_\text{testing}})$. Therefore, the first expression $\overline{EI}_{\text{testing}}$ represents the average performance obtained taking entropy into account as the primary objective when we choose an individual from the first Pareto front, and the second $\overline{SI}_{\text{testing}}$ taking into account sensitivity. So, we are taking the

TABLE II
STATISTICAL RESULTS FOR MPENSGA2E, MPENSGA2S, TRAINDIFFEVOL, MPANN-MSE, AND MPANN-HN FOR THE TESTING SETS OF THE TWO CLASSES AND MULTICLASS PROBLEMS CONSIDERED

Dataset	Algorithm	C(%)	S(%)	Dataset	Algorithm	C(%)	S(%)
			TWO CLASSES				
AustralianCard	MPENSGA2E	*88.07±1.57*	*86.13±2.73*	Breast Cancer	MPENSGA2E	**69.34±2.30**	*28.89±9.10*
	MPENSGA2S	**88.25±1.39**	**86.84±2.00**		MPENSGA2S	63.99±3.11	**53.09±6.58**
	TRAINDIFFEVOL	81.93±7.78	72.90±18.04		TRAINDIFFEVOL	*68.94±2.82*	26.35±11.17
	MPANN-MSE	87.59±1.18	85.95±1.98		MPANN-MSE	66.53±3.07	28.73±14.23
	MPANN-HN	87.78±2.49	85.83±4.43		MPANN-HN	66.53±3.07	28.41±14.34
Breast Cancer Wisconsin	MPENSGA2E	95.87±0.61	90.94±1.68	German	MPENSGA2E	**75.31±1.44**	*51.16±4.10*
	MPENSGA2S	95.60±0.85	90.72±1.84		MPENSGA2S	71.55±1.87	**68.80±3.11**
	TRAINDIFFEVOL	93.98±1.75	86.22±4.69		TRAINDIFFEVOL	71.73±2.11	28.36±19.90
	MPANN-MSE	*96.04±1.08*	*92.75±3.40*		MPANNMSE	73.61±1.80	48.89±5.33
	MPANN-HN	**96.27±1.00**	**93.30±3.36**		MPANN-HN	*73.76±1.77*	48.76±5.44
Heart Statlog	MPENSGA2E	**78.28±1.76**	61.89±2.09	Ionosphere	MPENSGA2E	**92.65±2.22**	**82.71±5.36**
	MPENSGA2S	*77.50±1.73*	*62.67±2.38*		MPENSGA2S	*92.08±2.30*	*82.40±4.14*
	TRAINDIFFEVOL	76.32±2.02	60.00±3.82		TRAINDIFFEVOL	85.23±4.68	65.31±9.11
	MPANN-MSE	76.91±1.10	**62.68±2.21**		MPANN-MSE	91.10±2.37	79.17±5.99
	MPANN-HN	76.91±1.10	**62.68±2.21**		MPANN-HN	91.10±2.37	79.17±5.99
Pima	MPENSGA2E	**78.99±1.81**	*60.45±2.59*	Vote	MPENSGA2E	**94.74±0.87**	**93.42±1.70**
	MPENSGA2S	76.96±2.09	**72.69±3.07**		MPENSGA2S	*94.68±0.91*	*93.38±1.61*
	TRAINDIFFEVOL	70.59±3.50	37.29±19.74		TRAINDIFFEVOL	93.39±1.69	91.99±1.69
	MPANN-MSE	*78.54±1.80*	59.65±3.71		MPANN-HN	94.19±1.28	92.53±2.34
	MPANN-HN	78.28±2.03	58.86±3.86		MPANN-HN	94.19±1.28	92.53±2.34
			MULTICLASS				
Autos	MPENSGA2E	**66.67±4.06**	*39.64±14.91*	Balance	MPENSGA2E	**94.02±1.53**	42.66±17.00
	MPENSGA2S	*66.04±4.78*	**42.28±10.97**		MPENSGA2S	92.48±2.16	**83.74±8.19**
	TRAINDIFFEVOL	26.79±7.49	0.00±0.00		TRAINDIFFEVOL	87.12±2.56	2.00±6.10
	MPANNMSE	48.42±3.71	0.00±0.00		MPANNMSE	*92.94±1.81*	*60.00±14.14*
	MPANN-HN	48.42±3.71	0.00±0.00		MPANN-HN	*92.94±1.81*	*60.00±14.14*
BTX	MPENSGA2E	**85.56±5.66**	**61.11±12.63**	Gene	MPENSGA2E	*86.42±2.04*	*80.85±3.45*
	MPENSGA2S	*85.40±5.99*	*60.00±13.56*		MPENSGA2S	**86.47±2.28**	**81.77±2.77**
	TRAINDIFFEVOL	71.11±3.94	1.11±6.09		TRAINDIFFEVOL	60.88±7.12	34.97±9.11
	MPANN-MSE	72.38±10.85	13.33±29.81		MPANN-MSE	75.11±4.98	36.20±3.87
	MPANN-HN	69.52±11.46	13.33±29.81		MPANN-HN	75.11±4.98	36.20±3.87
Iris	MPENSGA2E	**97.18±0.78**	**91.54±2.35**	Lymphography	MPENSGA2E	**85.05±4.24**	**5.17±19.67**
	MPENSGA2S	96.50±1.43	89.74±3.69		MPENSGA2S	**85.05±4.24**	**5.17±19.67**
	TRAINDIFFEVOL	*97.18±1.03*	*91.54±3.10*		TRAINDIFFEVOL	*81.98±4.62*	0.00±0.00
	MPANN-MSE	95.29±9.85	86.15±29.51		MPANN-MSE	80.45±5.96	0.00±0.00
	MPANN-HN	94.52±11.24	83.84±33.70		MPANN-HN	80.72±6.05	0.00±0.00
Newthyroid	MPENSGA2E	*95.12±2.31*	*74.81±10.08*	Post-op	MPENSGA2E	67.83±3.89	0.00±0.00
	MPENSGA2S	**95.56±2.15**	**75.08±10.67**		MPENSGA2S	38.12±16.6	**3.96±12.97**
	TRAINDIFFEVOL	91.11±4.77	59.47±22.74		TRAINDIFFEVOL	**69.57±1.14**	0.00±0.00
	MPANN-MSE	94.87±3.81	72.11±22.29		MPANN-MSE	*68.84±2.81*	0.00±0.00
	MPANN-HN	94.87±3.81	72.11±22.29		MPANN-HN	*68.84±2.81*	0.00±0.00
Vowel	MPENSGA2E	**79.51±3.19**	*57.39±9.84*	Yeast	MPENSGA2E	**59.91±0.98**	0.00±0.00
	MPENSGA2S	*77.36±3.90*	**57.97±6.07**		MPENSGA2S	*53.21±4.49*	**12.13±12.20**
	TRAINDIFFEVOL	40.83±2.80	0.00±0.00		TRAINDIFFEVOL	37.61±5.16	0.00±0.00
	MPANN-MSE	41.66±4.21	0.00±0.00		MPANN-MSE	48.87±3.53	0.00±0.00
	MPANN-HN	41.66±4.21	0.00±0.00		MPANN-HN	48.87±3.53	0.00±0.00

The best result is in bold face and the second best result in italic.

opposite ends of the Pareto front in each of the runs. Hence, the first procedure to obtain the average performance $\overline{EI}_{\text{testing}}$ is called MPENSGA2E and the second procedure to obtain the average performance $\overline{SI}_{\text{testing}}$ is called MPENSGA2S.

In all experiments, the population size for MPENSGA2 is established at $N_p = 100$. The mutation probability for each operator is equal to 1/5. For *iRprop+*, the adopted parameters are $\eta^- = 0.5$ (decreasing factor for stepsize), $\eta^+ = 1.2$ (increasing factor for stepsize), $\Delta_0 = 0.0125$ (the initial value of the stepsize for the weights Δ_{ij}), $\Delta_{\min} = 0$ (minimum stepsize for the weights), $\Delta_{\max} = 50$ (maximum stepsize for the weights), and Epochs $= 25$ (number of epochs for the local optimization).

C. Comparison Procedure

MPENSGA2 is compared to two popular ANN training algorithms:

- MPANN (memetic Pareto artificial neural networks) [4]. MPANN is an MOEA based on differential evolution [63] with two objectives; one is to minimize the error (MSE) and the other is to minimize the ANN complexity (number of hidden units). The BP algorithm is used in MPANN for local search. We have implemented a Java version using the pseudocode shown in [4] and the framework for evolutionary computation JCLEC [67], since the source code

TABLE III
MEAN TESTING ACCURACY ($\overline{C_G}(\%)$) AND SENSITIVITY ($\overline{S_G}(\%)$), MEAN ACCURACY RANKING ($\bar{R}_{C_G}$) AND MEAN SENSITIVITY RANKING ($\bar{R}_{S_G}$) FOR THE DIFFERENT METHODS EVALUATED AND THE EIGHTEEN DATA SETS

	Accuracy		Sensitivity	
Algorithm	$\overline{C_G}(\%)$	$\bar{R}_{C_G}$	$\overline{S_G}(\%)$	$\bar{R}_{S_G}$
MPENSGA2E	**82.81**	**1.56**	*55.70*	*2.17*
MPENSGA2S	*79.82*	*2.75*	**62.32**	**1.56**
TRAINDIFFEVOL	73.26	3.97	37.10	4.21
MPANNMSE	76.85	3.36	44.44	3.41
MPANN-HN	76.68	3.36	44.26	3.65

The best results is in bold face and the second best result in italic

TABLE IV
MEAN RANKING, CRITICAL DIFFERENCE VALUES, AND DIFFERENCES OF RANKINGS OF THE BONFERRONI–DUNN TESTS APPLIED FOR ACCURACY AND SENSITIVITY, USING MPENSGA2E AND MPENSGA2S AS THE CONTROL METHODS

ACCURACY		SENSITIVITY	
	Control Method		Control Method
$\bar{R}$	MPENSGA2E	$\bar{R}$	MPENSGA2S
$\bar{R}_{(1)}$=1.56	–	$\bar{R}_{(1)}$=2.17	$\lvert\bar{R}_{(2)}-\bar{R}_{(1)}\rvert=0.62$
$\bar{R}_{(2)}$=2.75	$\lvert\bar{R}_{(1)}-\bar{R}_{(2)}\rvert=1.19^{+}_{\circ}$	$\bar{R}_{(2)}$=1.56	–
$\bar{R}_{(3)}$=3.97	$\lvert\bar{R}_{(1)}-\bar{R}_{(3)}\rvert=2.42^{+}_{\bullet}$	$\bar{R}_{(3)}$=4.21	$\lvert\bar{R}_{(2)}-\bar{R}_{(3)}\rvert=2.65^{+}_{\bullet}$
$\bar{R}_{(4)}$=3.36	$\lvert\bar{R}_{(1)}-\bar{R}_{(4)}\rvert=1.81^{+}_{\bullet}$	$\bar{R}_{(4)}$=3.41	$\lvert\bar{R}_{(2)}-\bar{R}_{(4)}\rvert=1.85^{+}_{\bullet}$
$\bar{R}_{(5)}$=3.36	$\lvert\bar{R}_{(5)}-\bar{R}_{(3)}\rvert=1.81^{+}_{\bullet}$	$\bar{R}_{(5)}$=3.65	$\lvert\bar{R}_{(2)}-\bar{R}_{(5)}\rvert=2.09^{+}_{\bullet}$
$CD_{(\alpha=0.1)}$= 1.18; $CD_{(\alpha=0.05)}$= 1.32		$CD_{(\alpha=0.1)}$= 1.22; $CD_{(\alpha=0.05)}$= 1.32	

• : Statistically significant difference with $\alpha = 0.05$.
◦ : Statistically significant difference with $\alpha = 0.1$.
+ : The difference is in favour of the control method.
(1): MPENSGA2E; (2): MPENSGA2S; (3): TRAINDIFFEVOL; (4): MPANN-MSE; (5): MPANN-HN; CD: Critical Difference

TABLE V
BEST MODELS FOR PIMA IN TRAINING FOR THE EXTREMES OF THE PARETO FRONT

INDIVIDUAL	MATHEMATICAL MODEL
Best individual in (S, A_1) space considering A_1 (MPENSGA2E)	DF= 2.74 -4.74*(1/(1+exp{-(+1.07*(x1)+0.66*(x2)+0.43*(x3)-0.42*(x5)+5.89*(x6)-2.92*(x7)+0.06*(x8)+0.69)})) +4.81*(1/(1+exp{-(-0.17*(x1)-4.02*(x2)+1.07*(x3)+0.72*(x4)+0.14 * (x5)+1.22*(x6)-4.15*(x7)-2.96*(x8)-3.08)})) -1.09*(1/(1+exp{-(+0.62*(x1)+0.64*(x2)+0.24*(x3)-0.02*(x4)-0.02*(x5)+6.22*(x6)+1.06*(x7)+0.07*(x8)+1.95)})) +4.94*(1/(1+exp{-(+0.70*(x1)-0.57*(x2)-0.57*(x3)-0.52*(x4)-1.08*(x5)-1.87*(x6)+4.87*(x8)-2.84)})) $A_1(g)$ = 0.69; $A_2(g)$ =0.64; C_T=0.78; S_T=0.64; C_G =0.76; S_G=0.58; #neurons=4; #effective connections= 39
Best individual in (S, A_1) space considering S (MPENSGA2S)	DF = 2.77 -4.76*(1/(1+exp{-(+4.41*(x6)-2.98*(x7)+0.97*(1))})) +4.64*(1/(1+exp{-(+0.20*(x1)-4.73*(x2)+1.03*(x4)+0.10*(x5)-2.89*(x6) -4.55*(x7)-3.27*(x8)-3.01*(1))})) -1.09*(1/(1+exp{-(-0.07*(x2)+0.13*(x4)+0.77*(x6) +1.13*(x7)+0.07*(x8)+1.89*(1))})) +3.12*(1/(1+exp{-(+0.24*(x1)-0.26*(x7)+6.69*(x8)-2.66*(1))})) $A_1(g)$ =0.66; $A_2(g)$ =0.76; C_T=0.76; S_T=0.76; C_G =0.78; S_G=0.74; #neurons=4; #effective connections=26

is not publicly available. The strategy or automatic selection methodology of individuals is the same as that used with MPENSGA2 (see Section V-B). We also select the extremes in the Pareto front obtained by the algorithm. Thus, the methodology is named MPANN-MSE if the extreme chosen is the one that has better MSE; or MAPNN-HN if the extreme chosen is one with better complexity value. The local optimization algorithm incorporated in the algorithm is the *iRprop+* instead of the standard BP proposed in [4], since this algorithm has proven to render higher performance [35].

- TRAINDIFFEVOL (differential evolution training algorithm for feedforward neural networks) [36]. It is a mono-objective algorithm to train feedforward MLP neural networks, also based on the differential evolution [63]. This algorithm uses the mean squared error regularized by the mean squared weights and biases (MSEREG) for training the networks. We have implemented a modification of the source code provided by the authors[1] which obtains sensitivity for each class of the models trained.

[1]A Matlab implementation of TRAINDIFFEVOL is available at http://www.it.lut.fi/project/nngenetic/

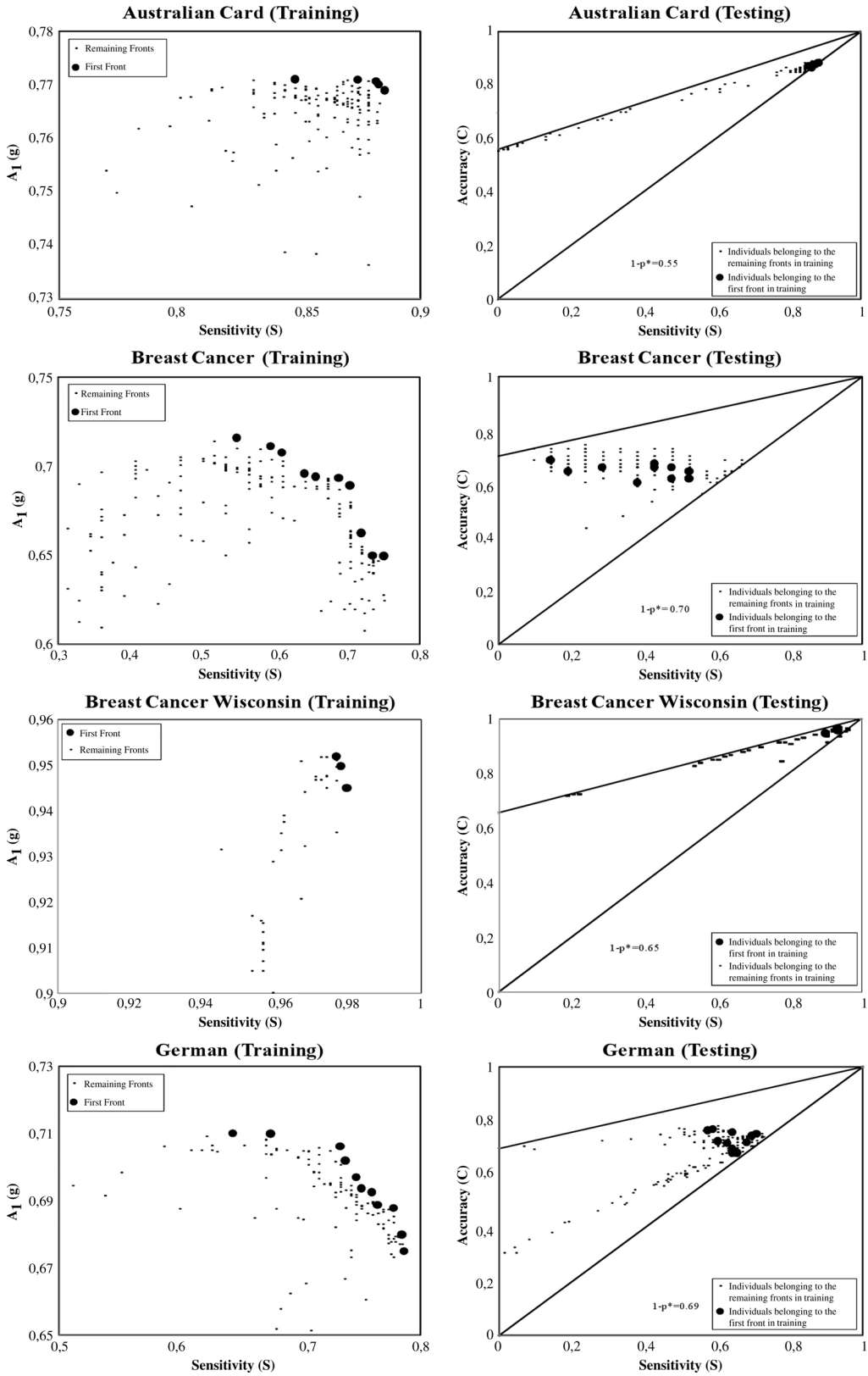

Fig. 6. Pareto front in training (S, A_1) and (S, C) associated values in testing for Australian Card, Breast Cancer, Breast Cancer Wisconsin, and German data sets in one specific run of the 30 runs carried out.

For MPANN and TRAINDIFFEVOL, we evaluate the testing accuracy and sensitivity of the best models obtained by different algorithms. From a statistical point of view, these comparisons are possible because we use the same partitions of the data sets. In other cases, it is difficult to justify the equity of the comparison procedure. Regarding the settings of each of the algorithms that MPENSGA2 has been compared to, we initially use the values advised by the authors in their respective paper.

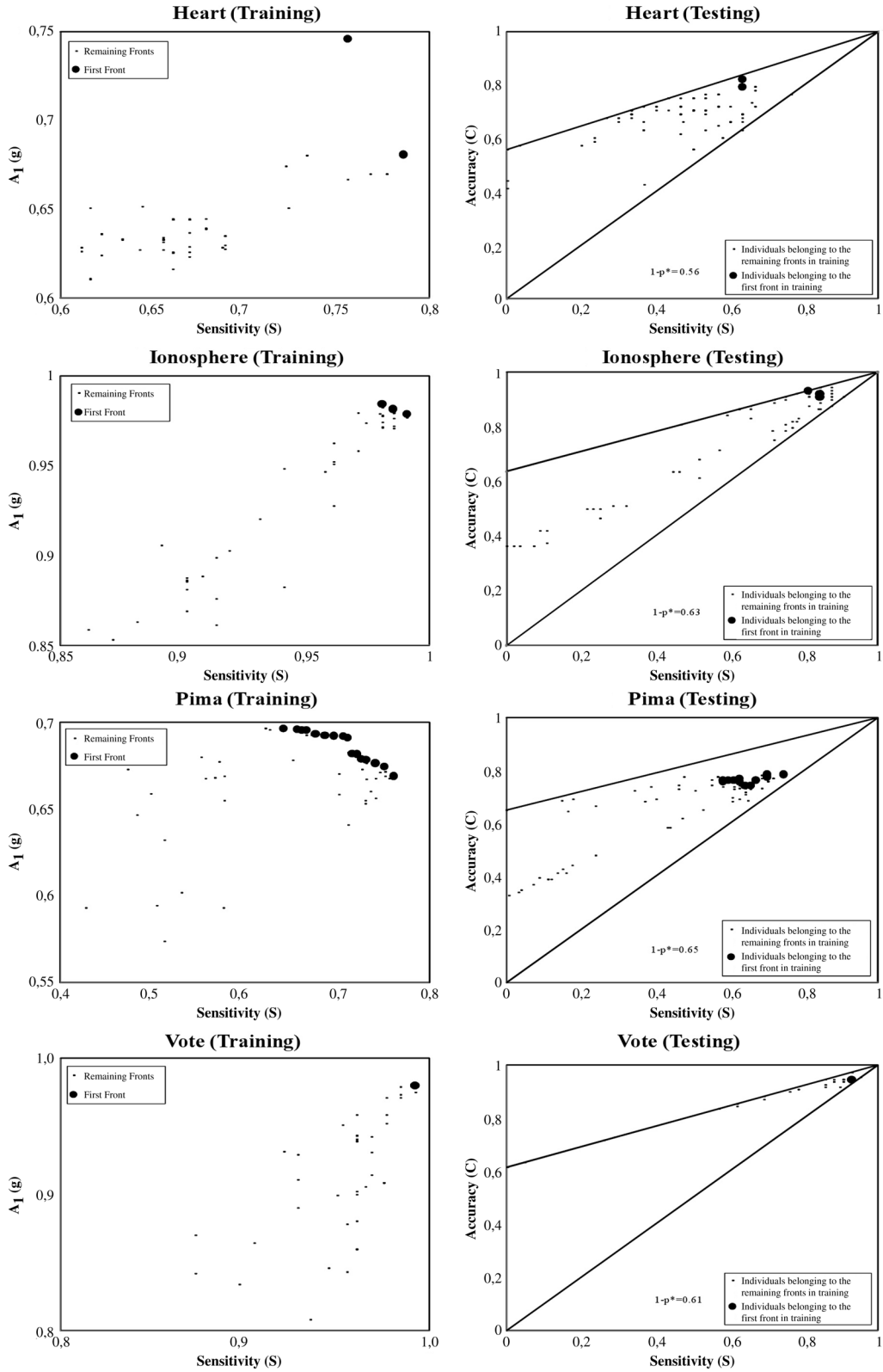

Fig. 7. Pareto front in training (S, A_1) and (S, C) associated values in testing for Heart, Ionosphere, Pima, and Vote data sets in one specific run of the 30 runs carried out.

However, having experimentally checked that those values resulted in a poor performance for some of the data sets considered (especially for multiclass problems), the most important parameters (number of epochs and number of individuals) have been individually adjusted by a trial and error process.

Table II presents the values of average and standard deviation for C and S for the testing set of each data set obtained for the best models in each run. This table is organized in the two different sets considered: two class problems and multiclass problems. In general, the best results for C or S are obtained through

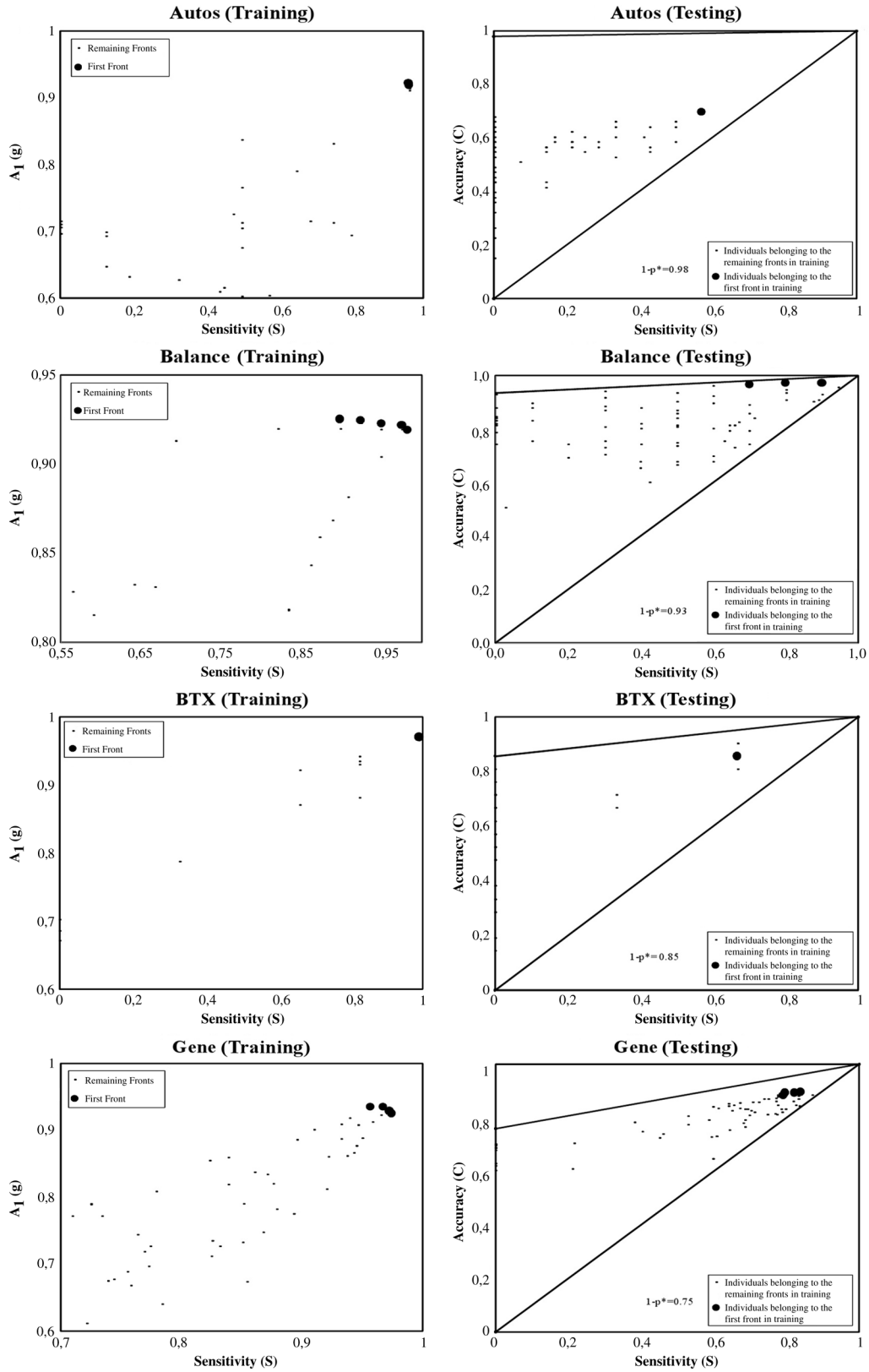

Fig. 8. Pareto front in training (S, A_1) and (S, C) associated values in testing for Autos, Balance, BTX, and Gene data sets in one specific run of the 30 runs carried out.

MPENSGA2E or MPENSGA2S approaches in all data sets except for Breast Cancer Wisconsin, Heart-Statlog, and Post-op.

Table III summarizes all these results, including the mean testing accuracy $\overline{C_G}(\%)$ and the mean sensitivity $\overline{S_G}(\%)$ for all the data sets and methods. The ranking of each method in each data set ($R = 1$ for the best performing method and $R = 5$ for the worst one) is obtained and the mean accuracy ranking throughout all the data sets $(\bar{R}_{C_G})$ and the mean sensitivity

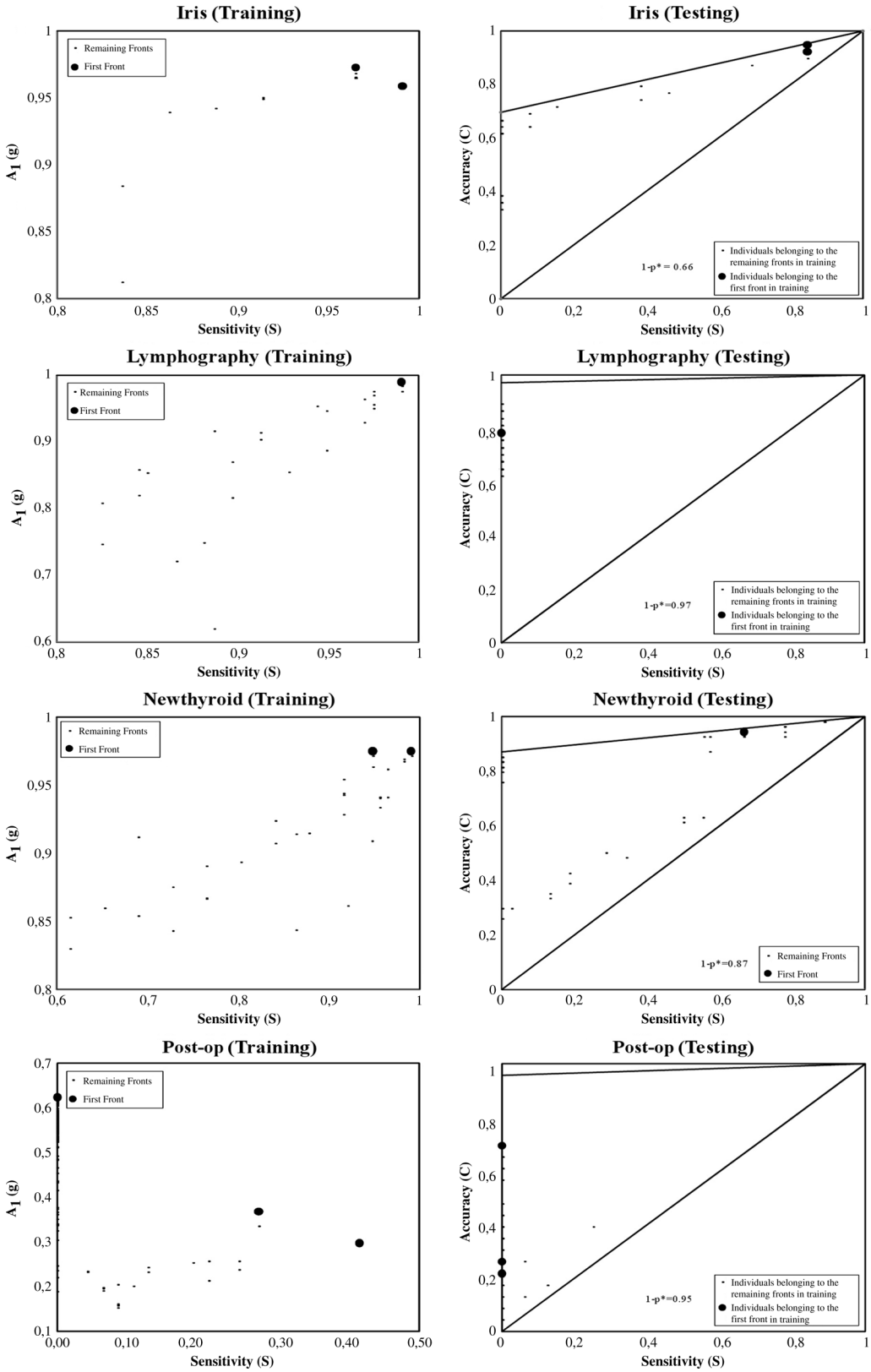

Fig. 9. Pareto front in training (S, A_1) and (S, C) associated values in testing for Iris, Lymphograpy, Newthyroid, and Post-op data sets in one specific run of the 30 runs carried out.

ranking $(\bar{R}_{S_G})$ are also included in Table III. By analyzing these two tables, the following specific comments can be made.

1) A descriptive analysis of the results leads to the following remarks: the MPENSGA2E methodology obtains the best results in C for 13 out of the 18 data sets, the second best results for three other data sets, the best mean accuracy $(\overline{C_G} = 82.81\%)$ and the best mean ranking $(\bar{R}_{C_G} = 1.56)$. In S, it obtains the best result for five data sets, the second

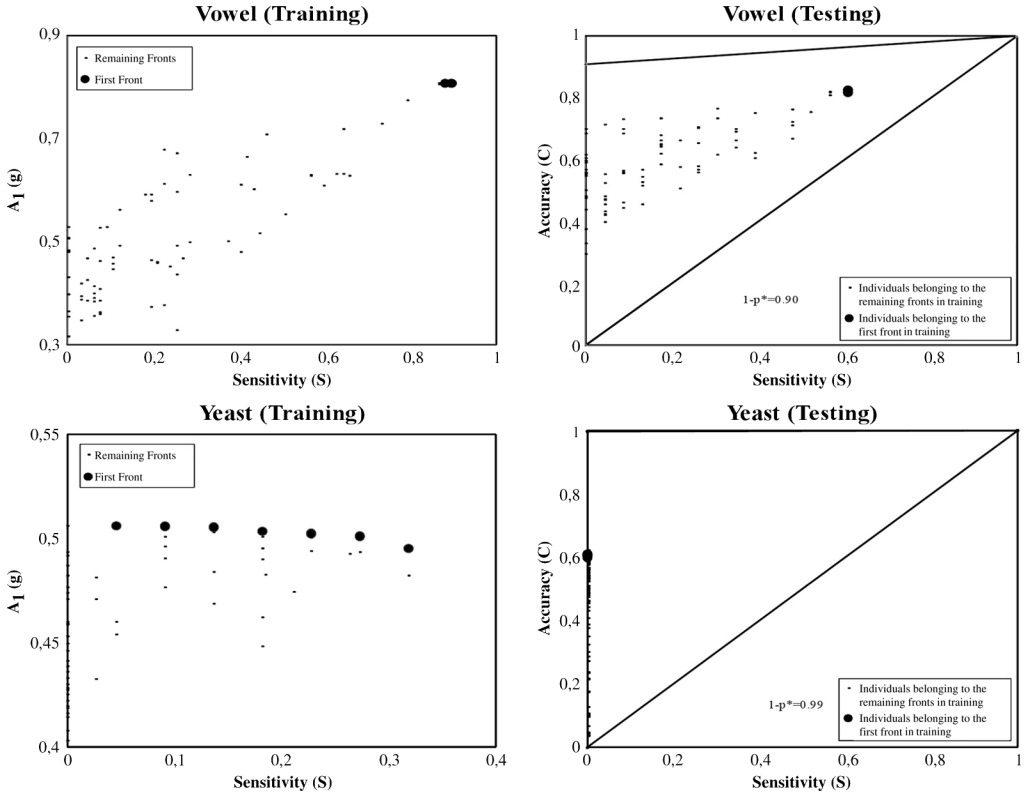

Fig. 10. Pareto front in training (S, A_1) and (S, C) associated values in testing for vowel and yeast data sets in one specific run of the 30 runs carried out.

best result for eight other data sets, the second best mean sensitivity ($\overline{S_G} = 55.70\%$), and the second best mean sensitivity ranking ($\bar{R}_{S_G} = 2.17$). Hence, we should choose this methodology (from a quantitative point of view) if our objective is to obtain classifiers with good results in C and acceptable values in S.

2) On the other hand, the results of the MPENSGA2S methodology show that, in C, it is the best method for four data sets and it is the second best for seven other data sets, resulting in the second best mean accuracy ($\overline{C_G} = 79.82\%$) and the second best mean accuracy ranking ($\bar{R}_{C_G} = 2.75$). In S, it obtains the best results for 12 data sets, the second best results for four other data sets, the best mean sensitivity ($\overline{S_G} = 62.32\%$), and the best mean sensitivity ranking ($\bar{R}_{S_G} = 1.56$). This methodology should be chosen if our objective is to obtain classifiers with good results in S and acceptable values in C.
3) A tendency to classify the examples in the majority class has been observed when analyzing the confusion matrices obtained by the models optimized by the TRAINDIFFEVOL algorithm. Consequently, this algorithm yields better results when applied to binary classification problems or well-balanced problems, but it yields a poor performance when applied to multiclass problems or very unbalanced problems.
4) The MPENSGA2E approach obtains the best accuracy and sensitivity values for Ionosphere, Vote, BTX, Iris, and Lymphography.
5) The MPENSGA2S approach obtains the best accuracy and sensitivity values for Australian Card, Gene, Lymphography, and Newthyroid.
6) The Autos, Lymphography, Post-op, and Yeast data sets deserve special mention. These problems are difficult classification problems for all methodologies because they are very unbalanced data sets (see Table II; all these data sets have a p^* value lower than 0.05). This makes the improvement in sensitivity very difficult. Although the accuracy rate is acceptable, the sensitivity level is very low due to the minority class with only 3, 2, 2, and 5 patterns, respectively. The MPENSGA2S procedure obtains for S: 42.28% on average in Autos, 5.17% in Lymphography, 3.96% in Post-op, and 12.13% in Yeast without dramatically reducing the average in C (see Table II), while the sensitivity with other methodologies maintains an average of 0.00%. When the problem is extremely unbalanced such as Autos and Lymphograpy, it is very difficult to improve sensitivity levels. These cases suggest, as future work, integrating resampling techniques [13] in our methodology. Then, we would see if these techniques obtain better results in C and S than our methodology.

To determine the statistical significance of the rank differences observed for each method in the different data sets, two nonparametric Friedman tests [23] have been carried out with the ranking of C_G and S_G of the best models as the test variable. The use of nonparametric tests is justified in this case, since a previous evaluation of the C_G and S_G values results in rejecting the normality and equality of the variance hypothesis (it is enough to observe the high variance values obtained especially for the sensitivity evaluation, and the differences existing between the variances in all methods). The test shows that the effect of the method used for classification is statistically significant for C_G values at a significance level of 5%, since the

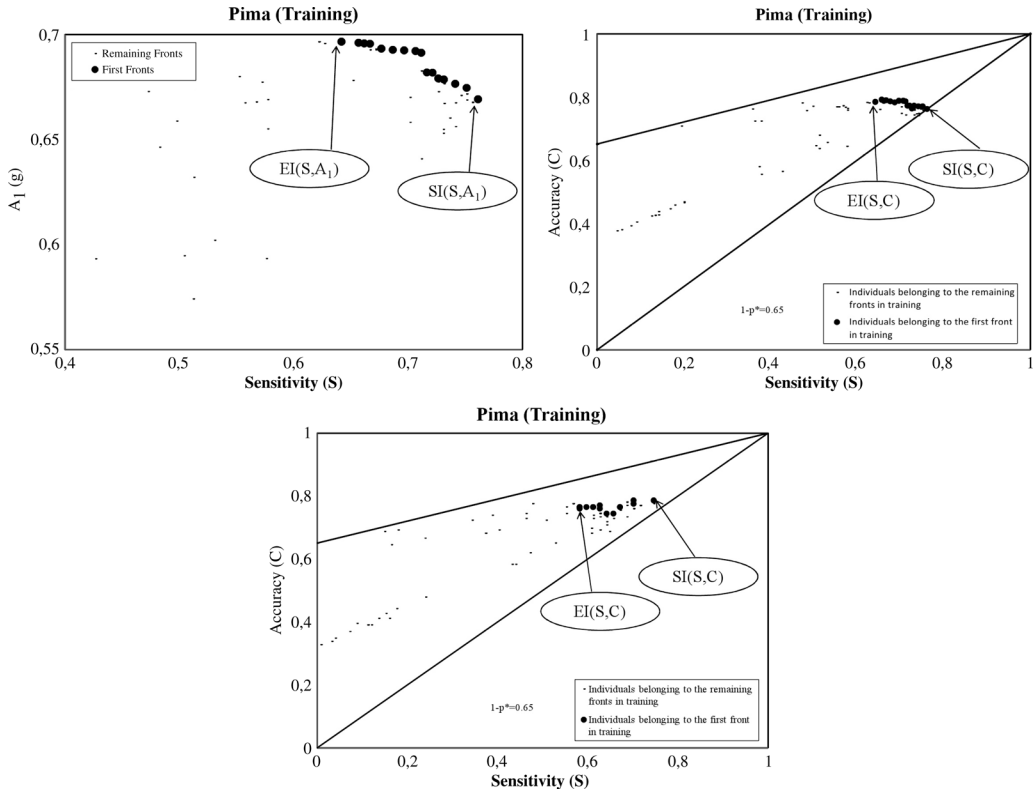

Fig. 11. Pareto front in training $(S, A1)$ and (S, C) associated values in training and testing for Pima data in one specific run.

confidence intervals are $C_0 = (0, F_{0.05} = 2.51)$ and the F-distribution statistical value is $F^* = 8.58 \notin F_0$. For S_G values, the test also shows the significance of the method applied, since the confidence intervals are $C_0 = (0, F_{0.05} = 2.52)$ and the F-distribution statistical value is $F^* = 14.75 \notin F_0$. Consequently, we reject the null hypothesis stating that all algorithms perform equally in mean accuracy and mean sensitivity rankings.

On the basis of these rejections, two post-hoc nonparametric Bonferroni–Dunn tests [18], [32] were applied, one for C_G values and other one for S_G values. The objective of these tests is to assess if the best performing algorithm for each evaluated measure (MPENSGA2E for C_G and MPENSGA2S for S_G) obtains significant differences in mean ranking when compared to other methods. In this way, MPENSGA2E will be the control method when comparing C_G mean ranking values and MPENSGA2S when comparing S_G mean ranking values. The results of the Bonferroni–Dunn test for $\alpha = 0.1$ and $\alpha = 0.05$ can be seen in Table IV, using the corresponding critical values for the two-tailed Bonferroni–Dunn test.

From the analysis of the results of these tests, we can conclude that the MPENSGA2E method obtains a statistically significant higher mean ranking of C_G when compared to all the methods, although the differences in the mean ranking of MPENSGA2S can only be assessed when we consider $\alpha = 0.1$. Regarding the MPENSGA2S method, it obtains a statistically significant higher mean ranking of S_G with $\alpha = 0.05$ when compared to all the methods except MPENSGA2E. It is important to note that both methodologies (MPENSGA2E and MPENSGA2S) result in a good tradeoff between the two objectives (accuracy and sensitivity) and this makes statistically significant differences between the two methods very difficult to assess. This is especially observed in those data sets where the individuals obtained are near optimum values in C or S.

Therefore, we can conclude that the methodologies used for obtaining models based on the Pareto front are well suited for improving one of the two objectives without, in general, causing the other to decrease.

These results are consistent with the Pareto fronts of entropy versus sensitivity for the training sets and the graphs of the correct classification rate versus sensitivity for the testing sets that are shown in Figs. 6– 10. In these figures, we can see the results obtained for each data set and the (S, C) space. The graphs are divided into (S, A_1) training graphs and (S, C) testing graphs. The procedure for obtaining these graphs is the following: in each of the 30 runs carried out, 30 Pareto fronts are obtained, then the front for one specific run is selected, the front selected being that which presents the best individual in Entropy in training at the end of the 30 runs of the hybrid EA. On the training graphs, we show the Pareto fronts obtained, A_1, and sensitivity being the objectives guiding the MPENSGA2. The testing graphs show the C and S values throughout the testing set for the individuals who are reflected in the training graphs. These graphs show the goodness of the classifiers obtained and their proximity to the $(1, 1)$ optimal point within the (S, C) space. Observe that the (S, C) values do not form Pareto fronts in the testing set, and the individuals that were in the first Pareto front in the training graphs can now be located within the (S, C) space in a region that is worse, since there is no exact mathematical relation between training entropy and testing accuracy, although an improvement in entropy produces an improvement in accuracy.

In the (S, C) graphs for the testing set, in general, and for sufficiently balanced data sets, these objectives are seen to be linearly related for low values of C and S, although they are in conflict for high values of C or S. For very unbalanced data sets, the increase in C does not imply an increase in S. We can also observe that in some data sets like Breast Cancer, German, and Pima, the size (cardinality) of the Pareto front is relatively large compared to others such as Autos, Lymphograpy, and Vote. There is a relationship between the number of points in the Pareto front, the size of each class of the problem, and the value of p^*.

Finally, so that the reader can observe the training classification error versus sensitivity for the MPENSGA2 algorithm, we have included a graph example in Fig. 11 for the Pima data set. In this figure, we can see the Pareto front in training in the $(S, A1)$ space and the associated values in the (S, C) space in training and testing for Pima data set in one specific run. Also we have shown the best models obtained in training (extreme of the Pareto front) together with the number of neurons, number of links, the $A_1(g)$, $A_2(g)$, C, and S values in training, and the C and S values in testing (see Table V). One important point to underline about these models is that the algorithm prunes its connections, selecting the most important variables (e.g., the MPENSGA2S does not include the $\times 3$ variable).

VI. Conclusion

In this paper, we have proposed a new approach for classification based on a 2-D performance measure associated with multiclass problems. Sensitivity S and accuracy C measures express two key features associated with a classifier: global performance C and the rate of the worst classified class S. Although sensitivity is not usually optimized in classification, it has been considered here given the need to obtain high precision in each class in real problems. The results show that optimizing these two measures, it is possible to obtain classifiers that combine a high classification level in the data set with a good classification rate for each class.

The methodology uses an MOEA which tries to boost these conflicting main objectives. Concretely, a memetic version of NSGA2, which introduces the *iRprop+* algorithm as a local optimizer adapted to the softmax activation function and the cross-entropy error function, designs the ANNs architecture finding an adequate number of neurons in the hidden layer and an adequate number of connections along with their corresponding weights. This automated optimization allows good values of accuracy and sensitivity to be obtained in the training and testing sets, and avoids overtraining.

The special features of the Pareto optimal front allowed us to consider two strategies or automatic selection methodologies of individuals: the best model in A_1 and the best model in sensitivity (extremes in the Pareto front). The approach is applied to solve 17 benchmark classification problems from UCI repository and a complex environment problem in qualitative analytical chemistry. The results confirm that the Memetic NSGA2 method in the two versions (MPENSGA2E and MPENSGA2S) obtains promising results with good levels of accuracy and sensitivity, improving the average values of sensitivity in the classifier in almost all the data sets analyzed, and obtaining values of accuracy similar to or higher than those obtained with other DE methodologies, except in very unbalanced data sets.

In our opinion, the (S, C) perspective and the memetic NSGA2 approach reveal a new point of view for dealing with multiclass classification problems, and provide the opportunity to improve the sensitivity and accuracy of a multiclassifier for a wide range of data sets.

Several future research directions are suggested by this study. First, other memetic MOEAS based on the (S, C) measures could be devised. Second, since the (S, C) measures are independent of the EA and the base classifier used, other types of base classifiers can be considered. Third, ensemble tools could be incorporated to deal with the Pareto-front obtained. Finally, a possible extension of the current work is to see whether this approach can be adapted to deal with very unbalanced problems, incorporating resampling techniques in the algorithm.

Acknowledgment

The authors would like to thank the anonymous reviewers who clearly dedicated considerable time and effort to the paper, and whose helpful comments led to considerable improvements.

References

[1] H. A. Abbass, "A memetic pareto evolutionary approach to artificial neural networks," in *Proc. 14th Australian Joint Conf. Artif. Intell.: Adv. Artif. Intell.*, London, U.K., Dec. 2001, pp. 1–12.

[2] H. A. Abbass, "An evolutionary artificial neural networks approach for breast cancer diagnosis," *Artif. Intell. Med.*, vol. 25, pp. 265–281, 2002.

[3] H. A. Abbass, "Pareto neuro-evolution: Constructing ensemble of neural networks using multiobjective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Canberra, Australia, Jun. 2003, pp. 2074–2080.

[4] H. A. Abbass, "Speeding up backpropagation using multiobjective evolutionary algorithms," *Neural Comput.*, vol. 15, pp. 2705–2726, 2003.

[5] H. A. Abbass, R. Sarker, and C. Newton, "PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems," in *Proc. Congr. Evol. Comput.*, Seoul, South Korea, May 2001, pp. 971–978.

[6] R. Albuquerque, A. Pádua, R. H. C. Takahashi, and R. R. Saldanha, "Improving generalization of MLPs with multi-objective optimization," *Neurocomputing*, vol. 35, pp. 189–194, 2000.

[7] P. J. Angeline, G. M. Saunders, and J. B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," *IEEE Trans. Neural Netw.*, vol. 5, no. 1, pp. 54–65, Jan. 1994.

[8] A. Asuncion and D. J. Newman, 2007, UCI Machine Learning Repository, [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[9] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer-Verlag, 2006.

[11] A. P. Braga, R. H. C. Takahashi, M. A. Costa, and R. A. Teixeira, "Multi-objective algorithms for neural networks learning," *Studies Comput. Intell.*, vol. 16, pp. 151–171, 2006.

[12] A. Chandra and X. Yao, "DIVACE: Diverse and accurate ensemble learning algorithm," in *Proc. 5th Int. Conf. Intell. Data Eng. Autom. Learn.*, Exeter, U.K., Aug. 2005, pp. 619–625.

[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[14] C. A. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*. New York: Springer-Verlag, 2007.

[15] C. Cotta and P. Moscato, "A gentle introduction to memetic algorithms," in *Handbook on Metaheuristics*, F. Glover and G. A. Kochenberger, Eds. Norwell, MA: Kluwer, 2001.

[16] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. New York: Wiley, 2004.

[17] K. Deb, A. Pratab, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA2," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[18] O. J. Dunn, "Multiple comparisons among means," *J. Amer. Stat. Assoc.*, vol. 56, pp. 52–56, 1961.
[19] R. M. Everson and J. E. Fieldsend, "Multi-class ROC analysis from a multi-objetive optimisation perspective," *Pattern Recognit. Lett.*, vol. 27, pp. 918–927, 2006.
[20] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, pp. 861–874, 2006.
[21] J. E. Fieldsend and S. Singh, "Pareto evolutionary neural networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 338–354, Mar. 2005.
[22] J. E. Fieldsend and S. Singh, "Pareto multi-objective non-linear regression modelling to aid CAPM analogous forecasting," in *Proc. Int. Joint Conf. Neural Netw.*, Honolulu, HI, May 2002, pp. 388–393.
[23] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *Ann. Math. Stat.*, vol. 11, no. 1, pp. 86–92, 1940.
[24] C. Fu, C. Chien-Hsing, L. Chin-Chin, and C. Chun-Jen, "A prototype classification method and its application to handwritten character recognition," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Mar. 2004, vol. 5, pp. 4738–4743.
[25] E. Gelenbe and K. F. Hussain, "Learning in the multiple class random neural network," *IEEE Trans. Neural Netw.*, vol. 13, no. 6, pp. 1257–1267, Nov. 2002.
[26] A. Gepperth and S. Roth, "Applications of multi-objective structure optimization," *Neurocomputing*, vol. 69, pp. 701–703, 2006.
[27] T. Van Gestel, B. Baesens, J. Suykens, M. Espinoza, D. E. Baestaens, J. Vanthienen, and B. De Moor, "Bankruptcy prediction with least squares support vector machine classifiers," in *Proc. IEEE Int. Conf. Comput. Intell. Financial Eng.*, Hong Kong, Mar. 2003, pp. 1–8.
[28] J. González, I. Rojas, J. Ortega, H. Pomares, F. J. Fernández, and A. F. Días, "Multiobjective evolutionary optimization of the size, shape, and position parameters of radial basis function networks for function approximation," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1478–1495, Nov. 2003.
[29] C. Hervás, M. Silva, P. A. Gutiérrez, and A. Serrano, "Multilogistic regression by evolutionary neural network as a classification tool to discriminate highly overlapping signals: Qualitative investigation of volatile organic compounds in polluted waters by using headspace-mass spectrometric analysis," *Chemometrics Intell. Lab. Syst.*, vol. 92, pp. 179–185, 2008.
[30] T. K. Ho and M. Basu, "Complexity measures of supervised classification problems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 289–300, Mar. 2002.
[31] D. C. Hoaglin, F. Mosteller, and J. W. Tukey*, Understanding Robust and Exploratory Data Analysis*. New York: Wiley, 1983.
[32] Y. Hochberg and A. Tamhane*, Multiple Comparison Procedures*. New York: Wiley, 1987.
[33] C. R. Houck, J. A. Joines, M. G. Kay, and J. R. Wilson, "Empirical investigation of the benefits of partial lamarckianism," *Evol. Comput.*, vol. 5, pp. 31–60, 1997.
[34] C. Igel and M. Hüsken, "Empirical evaluation of the improved Rprop learning algorithms," *Neurocomputing*, vol. 50, no. 6, pp. 105–123, 2003.
[35] C. Igel and M. Hüsken, "Improving the Rprop learning algorithm," in *Proc. 2nd Int. ICSC Symp. Neural Comput.*, Berlin, Germany, May 2000, pp. 115–121.
[36] J. Ilonen, J. K. Kamarainen, and J. Lampinen, "Differential evolution training algorithm for feed-forward neural networks," *Neural Process. Lett.*, vol. 17, pp. 93–105, 2003.
[37] A. K. Jain, R. P. W. Duin, and M. Jianchang, "Statistical pattern recognition: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 4–37, Jan. 2000.
[38] Y. Jin, T. Okabe, and B. Sendhoff, "Neural network regularization and ensembling using multi-objective evolutionary algorithms," in *Proc. Congr. Evol. Ensembling Using Multi-Objective Evol. Algorithms*, Portland, OR, Jan. 2004, pp. 1–8.
[39] Y. Jin, T. Okabe, and B. Sendhoff, "Evolutionary multi-objective approach to constructing neural network ensembles for regression," in *Applications of Evolutionary Multi-Objective Optimization*. Singapore: World Scientific, 2004, pp. 653–672.
[40] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.*, vol. 38, no. 3, pp. 397–415, Jun. 2008.
[41] Y. Jin, B. Sendhoff, and E. Körner, "Evolutionary multi-objective optimization for simultaneous generation of signal-type and simbol-type representations," in *Proc. 3rd Int. Conf. Evol. Multi-Criterion Optim.*, Guanajuato, Mexico, Mar. 2005, pp. 752–766.
[42] H. Kaizhu, Y. Haiqin, K. Irwin, and M. R. Lyu, "Maximizing sensitivity in medical diagnosis using biased minimax probability Machine," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 5, pp. 821–831, May 2006.
[43] T. C. W. Langrebe and R. P. W. Duin, "Efficient multiclass ROC approximation by decomposition via confusion matrix perturbation analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 810–822, May. 2008.
[44] Y. Liu and X. Yao, "Ensemble learning via negative correlation," *Neural Netw.*, vol. 12, no. 10, pp. 1399–1404, 1999.
[45] T. B. Ludermir, A. Yamazaki, and C. Zanchettin, "An optimization methodology for neural network weights and architectures," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1452–1459, Nov. 2006.
[46] E. Lughofer, J. E. Smith, M. A. Tahir, P. Caleb-Solly, C. Eitzinger, D. Sannen, and M. Nuttin, "Human-machine interaction issues in quality control based on online image classification," *IEEE Trans. Syst. Man Cybern. A, Syst. Humans*, vol. 39, no. 5, pp. 960–971, Sep. 2009.
[47] A. C. Martínez-Estudillo, C. Hervás-Martínez, F. J. Martínez-Estudillo, and N. García, "Hybridation of evolutionary algorithms and local search by means of a clustering method," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 36, no. 3, pp. 534–546, Jun. 2006.
[48] F. J. Martínez-Estudillo, P. A. Gutiérrez, C. Hervás, and J. C. Fernández, "Evolutionary learning by a sensitivity-accuracy approach for multi-class problems," in *Proc. IEEE World Congr. Comput. Intell.*, Hong-Kong, Jun. 2008, pp. 1581–1588.
[49] D. Mossman, "Three-way ROCs," *Med. Decision Making*, vol. 19, no. 1, pp. 78–89, 1999.
[50] G. Ou and Y. L. Murphey, "Multi-class pattern classification using neural networks," *Pattern Recognit.*, vol. 40, pp. 4–18, 2007.
[51] P. Palmes, T. Hayasaka, and S. Usui, "Mutation-based genetic neural network," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 587–600, May 2005.
[52] R. Parekh, J. Yang, and V. Honavar, "Constructive neural-network learning algorithms for pattern classification," *IEEE Trans. Neural Netw.*, vol. 11, no. 2, pp. 436–450, Mar. 2000.
[53] L. Prechelt, "A set of neural network benchmark problems and benchmarking rules," Fakultät für Informatik, Universität Karlsruhe, Karlsruhe, Germany, Tech. Rep. 21/94, 1994.
[54] K. V. Price, R. M. Storn, and J. A. Lampinen*, Differential Evolution. A Practical Approach to Global Optimization*. New York: Springer-Verlag, 2005.
[55] F. Provost and T. Fawcett, "Robust classification system for imprecise environments," in *Proc. 15th Nat. Conf. Artif. Intell.*, Madison, WI, Jul. 1998, pp. 706–713.
[56] F. Provost and T. Fawcett, "Analysis and visualization of the classifier performance: Comparison under imprecise class and cost distribution," in *Proc. 3rd Int. Conf. Knowl. Disc. Data Mining*, Aug. 1997, pp. 43–48.
[57] R. Reed, "Pruning algorithms. A survey," *IEEE Trans. Neural Netw.*, vol. 4, no. 5, pp. 740–747, Sep. 1993.
[58] A. J. F. van Rooij, L. C. Jain, and R. P. Johnson*, Neural Networks Training Using Genetic Algorithms*. Singapore: World Scientific, 1996.
[59] S. Roth, A. Gepperth, and C. Igel, "Multi-objective neural network optimization for visual object detection," *Multi-Objective Mach. Learn., Studies Comput. Intell.*, vol. 16, pp. 629–655, 2006.
[60] E. J. Scheme, B. Hudgins, and P. A. Parker, "Myoelectric signal classification for phoneme-based speech recognition," *IEEE Trans. Biomed. Eng.*, vol. 54, no. 4, pp. 694–699, Apr. 2007.
[61] R. Setiono, W. K. Leow, and J. M. Zurada, "Extraction of rules from artificial neural networks for nonlinear regression," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. 564–577, May 2002.
[62] J. E. Smith, "Coevolving memetic algorithms: A review and progress report," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 37, no. 1, pp. 6–17, Feb. 2007.
[63] R. Storn and K. Price, "Differential evolution. A fast and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, pp. 341–359, 1997.
[64] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognit.*, vol. 40, pp. 3358–3378, 2007.

[65] X. Tian and F. Deng, "An improved multi-class SVM algorithm and its application to the credit scoring model," in *Proc. 5th World Congr. Intell. Control Autom.*, Hangzhou, China, Jun. 2004, pp. 1940–1944.
[66] K. Tumer and J. Ghosh, "Analysis of decision boundaries in linearly combined neural classifiers," *Pattern Recognit.*, vol. 29, no. 2, pp. 341–348, 1996.
[67] S. Ventura, C. Romero, A. Zafra, J. A. Delgado, and C. Hervás, "JCLEC: A Java framework for evolutionary computation," *Soft Comput.—A Fusion Found. Methodol. Appl.*, vol. 12, pp. 381–392, 2008.
[68] D. L. Whitley, V. S. Gordon, and K. E. Mathias, "Lamarckian evolution, the baldwing effect and function optimization," in *Proc. Int. Conf. Evol. Comput./3rd Conf. Parallel Probl. Solving From Nature*, Jerusalem, Israel, Oct. 1994, pp. 6–15.
[69] W. Yan, Z. Zhu, and R. Hu, "Hybrid genetic/BP algorithm and its application for radar target classification," in *Proc. IEEE Nat. Aerosp. Electron. Conf.*, Dayton, OH, Jul. 1997, pp. 981–984.
[70] X. Yao, "Evolving artificial neural networks," *Proc. IEEE*, vol. 87, no. 9, pp. 1423–1447, Sep. 1999.
[71] X. Yao, "Evolutionary artificial neural networks," *Int. J. Intell. Syst.*, vol. 4, no. 3, pp. 203–222, 1993.
[72] X. Yao and Y. Liu, "A new evolutionary system for evolving artificial neural networks," *IEEE Trans. Neural Netw.*, vol. 8, no. 3, pp. 694–713, May 1997.
[73] X. Yao and Y. Liu, "Making use of population information in evolutionary artificial neural networks," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 28, no. 3, pp. 417–425, Jun. 1998.
[74] Y. J. Yau, J. Teo, and P. Anthony, "Pareto evolution and co-evolution in cognitive game AI synthesis," in *Proc. 4th Int. Conf. Evol. Multi-Criterion Optim.*, Matsushima, Japan, Mar. 2007, pp. 227–241.
[75] G. P. Zhang, "Neural networks for classification: A survey," *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.*, vol. 30, no. 4, pp. 451–462, Nov. 2000.

Juan Carlos Fernández Caballero (M'08) was born in Peñarroya-Pueblonuevo, Córdoba, Spain, in 1980. He received the B.S. degree in computer science from the University of Granada, Granada, Spain, in 2005. Currently, he is working towards the Ph.D. degree in computer science and artificial intelligence at the University of Córdoba, Córdoba, Spain.

He is an Assistant Professor at the Department of Computer Science and Numerical Analysis, University of Córdoba. His current areas of interest include neural networks and applications, evolutionary computation, and multiobjective optimization.

Francisco José Martínez (M'08) was born in Villacarrillo, Jaén, Spain. He received the B.S. and Ph.D. degrees in mathematics from the University of Granada, Granada, Spain, in 1987 and 1991, respectively.

From 1987 to 2002, he developed his research in non-Euclidean geometry, Lorentz spaces, and maximal surfaces. He is currently a Professor at the Department of Management and Quantitative Methods in ETEA, University of Córdoba, Córdoba, Spain. His current research interests include structure optimization of neural networks, evolutionary algorithms, and multiobjective optimization.

César Hervás (M'08) was born in Cuenca, Spain. He received the B.S. degree in statistics and operating research from the Universidad Complutense, Madrid, Spain, in 1978 and the Ph.D. degree in mathematics from the University of Seville, Seville, Spain, in 1986.

Currently, he is a Professor in the area of computer science and artificial intelligence at the Department of Computer Science and Numerical Analysis, University of Córdoba, Córdoba, Spain, and an Associate Professor at the Department of Quantitative Methods, School of Economics. His current research interests include neural networks, evolutionary computation, and modeling of natural systems.

Pedro Antonio Gutiérrez (M'08) was born in Córdoba, Spain, in 1982. He received the B.S. degree in computer science from the University of Sevilla, Seville, Spain, in 2006 and the Ph.D. degree in computer science and artificial intelligence from the University of Granada, Granada, Spain, in 2009.

Currently, he is an Assistant Professor at the Department of Computer Science and Numerical Analysis, University of Córdoba, Córdoba, Spain. His current areas of interest include neural networks and their applications, evolutionary computation, and hybrid algorithms.