Memetic Evolutionary Multi-Objective Neural Network Classifier to Predict Graft Survival in Liver Transplant Patients

M. Cruz-Ramírez*
Dept. Computer Science and Numerical Analysis
U. of Córdoba, Spain
mcruz@uco.es

J. C. Fernández
Dept. Computer Science and Numerical Analysis
U. of Córdoba, Spain
jfcaballero@uco.es

F. Fernández-Navarro
Dept. Computer Science and Numerical Analysis
U. of Córdoba, Spain
i22fenaf@uco.es

J. Briceño
Liver Transplantation Unit
Hospital Reina Sofía
Córdoba, Spain. CIBERehd
javibriceno@hotmail.com

M. de la Mata
Liver Transplantation Unit
Hospital Reina Sofía
Córdoba, Spain. CIBERehd
mdelamatagarcia@gmail.com

C. Hervás-Martínez
Dept. Computer Science and Numerical Analysis
U. of Córdoba, Spain
chervas@uco.es

ABSTRACT

In liver transplantation, matching donor and recipient is a problem that can be solved using machine learning techniques. In this paper we consider a liver transplant dataset obtained from eleven Spanish hospitals, including the patient survival or the rejection in liver transplantation one year after the surgery. To tackle this problem, we use a multi-objective evolutionary algorithm for training generalized radial basis functions neural networks. The obtained models provided medical experts with a mathematical value to predict survival rates allowing them to come up with a right decision according to the principles of justice, efficiency and equity.

Keywords

Artificial neural networks, Generalized radial basis functions, Liver transplantation, Multi-objective evolutionary algorithm

1. INTRODUCTION

Nowadays, a liver transplantation or hepatic transplantation is a well accepted treatment for patients with terminal liver disease. Predicting the first one-year graft survival post transplantation has the potential to play a critical role in understanding and improving the matching procedure between the recipient and donor. Although a high number of data related to the transplantation procedures has been collected over the last years, only a small subset of the predictive factors has been used in modeling liver transplantation outcomes in order to predict survival rates.

Organ transplantation procedures involve a large number of variables that may have significant impact on the survival of the graft and/or the patient. The omission of the vast majority of the variables may hinder the discovery of underlying relationships between survival and the related factors. In such approaches the complete information underlying the transplantation datasets cannot be revealed effectively. This may cause non-optimal policy adoptions. The further steps (e.g., donor-recipient match) would also be ineffective since they build on the first step, namely, determination of significant variables, which would indicate to which patient an organ should be allocated based on what criteria.

New trends in biomedicine consider Artificial Neural Networks (ANNs) as a classification method, where a decision is based on the recognition of complex patterns within the data. ANNs have been studied in a wide variety of medical applications, such as cancer diagnosis [2], myocardial infarction [20], diagnosis of thyroid function [22] or predictive blood glucose levels [21].

In the field of organ transplantation, ANNs have been designed, for example, to diagnose cytomegalovirus disease [23]. In addition, the use of ANNs was investigated in the prediction of graft failure [18], in the prediction of liver transplantation outcome [11], in the selection of patients for liver transplantation [19] and in the prediction of tacrolimus blood levels [8]. However, these studies used static models to predict long-term outcomes based on data collected before and immediately after transplantation.

In this paper, we study the generalization improvement of classifiers designed using a Multi-Objective Evolutionary Algorithm (MOEA) [9] for the determination of receiver organ suitability. These classifiers evaluate the one-year survival

*Corresponding author at: Department of Computer Science and Numerical Analysis, University of Córdoba, Rabanales Campus, Albert Einstein Building 3rd Floor, 14071 Córdoba, Spain. Tel.: +34 957 218 349; Fax: +34 957 218 630

after liver transplantation in eleven hospitals and they are aimed to achieve a high classification level for each class. The method is based on two measures: the correct classification rate (C) and the minimum sensitivity (MS), as the minimum of the sensitivities of all classes.

We use a generalized version of the standard Radial Basis Function Neural Network (RBFNN), based on the probability density function of the Generalized Gaussian Distribution (GGD). This basis functions are called Generalized gaussian Radial Basis Functions (GRBF). The GGD may represent different forms of distribution function by changing a real parameter τ, such as the impulsive, Laplacian, Gaussian and uniform distributions. Based on this probability distribution, the GRBF is defined by removing the constraints of a probability function.

The paper is organized as follows: Section 2 shows a description of the method used; Section 3 describes the GRBF; Section 4 describes the dataset used and explains the experimental design; Section 5 shows the results obtained, while conclusions and future research are outlined in Section 6.

2. METHOD

The MOEA used in this paper is called Memetic Pareto Differential Evolutionary Neural Network (MPDENN). This section briefly explains the schema of this algorithm. For more details, see [10].

The MPDENN algorithm was developed by R. Storn and K. Price in [24], modified by H. Abbass to train ANNs [1] and adapted by our research group for C and MS measures [12]. The fundamental bases of this algorithm are Differential Evolution (DE) and the concept of Pareto dominance. In this case, the MPDENN algorithm is used for training ANNs with GRBFs (see Section 3) in order to solve a real-world complex problem of donor-recipient allocation in liver transplant.

Figure 1 shows the framework of the MPDENN algorithm. This algorithm starts generating a random population of size N. Each individual in the population is a neural network, represented as a vector, that is, each neural network is a vector of layers, each layer is a vector of neurons and each neuron is a vector of links. The population is sorted according to the non-domination concept and dominated individuals are removed from the population. Then the population is adjusted until its size is between 3 and half the maximum size by adding dominated individuals or deleting individuals according to their respective distance from their nearest neighbor. After that, the population is completed with new offspring generated from three randomly selected individuals in the population. The child is generated applying the crossover operator to the three parents (α_1, α_2 and α_3). The resultant child is a perturbation of the main parent (α_1). This perturbation occurs with a probability p_c for each neuron. It may be: structural, according to expression (1), where neurons are removed or added to the hidden layer; or parametric, according to expression (2) for the hidden layer, or (3) for the output layer, where the weight of the main parent (α_1) is modified by the difference between the weights of the secondary parents (α_2 and α_3) multiplied by a random variable with normal distribution, $N(0,1)$.

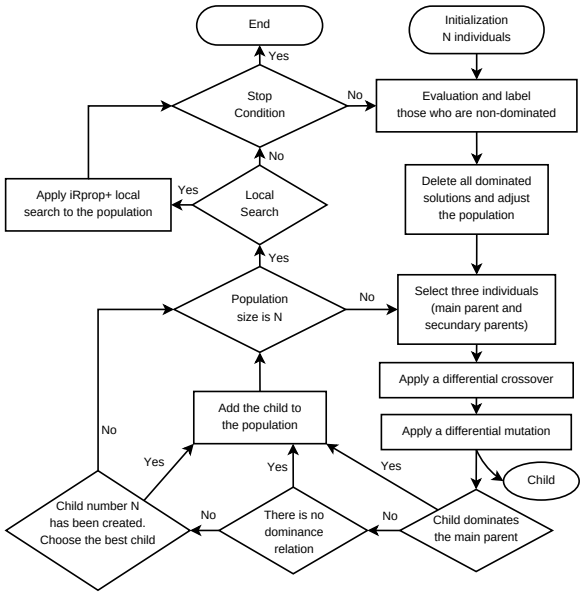

Figure 1: Framework for MPDENN.

$$\rho_h^{child} \leftarrow \begin{cases} 1 & if\,(\rho_h^{\alpha_1}+)\,N(0,1)\,(\rho_h^{\alpha_2}-\rho_h^{\alpha_3}) \geq 0.5 \\ 0 & otherwise \end{cases}, \quad (1)$$

$$w_{ih}^{child} \leftarrow w_{ih}^{\alpha_1} + N(0,1)\,(w_{ih}^{\alpha_2} - w_{ih}^{\alpha_3}), \quad (2)$$

$$w_{ho}^{child} \leftarrow w_{ho}^{\alpha_1} + N(0,1)\,(w_{ho}^{\alpha_2} - w_{ho}^{\alpha_3}), \quad (3)$$

where $\rho_h^{\alpha_1}$, $\rho_h^{\alpha_2}$ and $\rho_h^{\alpha_3}$ are binary values representing whether or not the hidden neuron h is in the parents α_1, α_2, and α_3, respectively; ρ_h^{child} indicates whether or not the hidden neuron h is in the child; $w_{ih}^{\alpha_1}$ is the weight between the input neuron i and hidden neuron h in the parent α_p and $w_{ho}^{\alpha_p}$ is the weight between the hidden neuron h and output neuron o in the parent α_p; w_{ih}^{child} and w_{ho}^{child} are the weights in the child.

Afterwards, the mutation operator is applied to the child. The mutation operator consists in adding or deleting neurons in the hidden layer depending on a p_m probability for each of them. If the neuron exists, it is deleted, but if it does not exist, then it is created and the weights are established randomly, according to expression (4).

$$\rho_h^{child} \leftarrow \begin{cases} 1 & if\,\rho_h^{child} = 0 \\ 0 & otherwise \end{cases}. \quad (4)$$

Finally, the child is added to the population according to dominance relationships with the main parent, that is, the child is added if it dominates the main parent α_1, if there is not dominance relationship with him or if it is the best

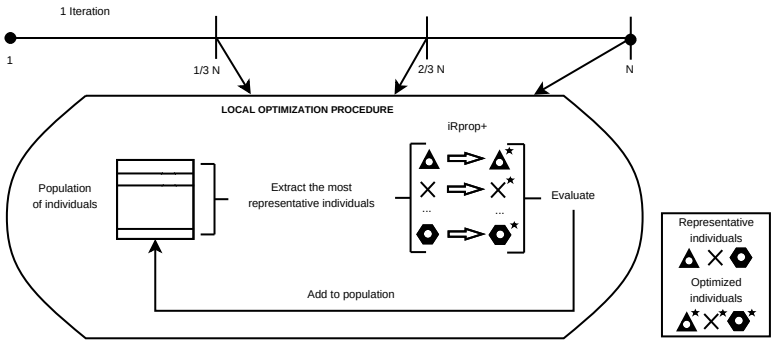

Figure 2: Local search procedure Scheme.

child of the N rejected children (where N is the population size).

This process of generation of children is repeated until the population is complete.

In three concrete generations of the evolution (the first initially, the second in the middle and the third at the end), a local search algorithm is applied once the population is completed. This procedure can be seen in Figure 2. Local search does not apply to all individuals, only to the most representative. The process for selecting these individuals is as follows: if the number of individuals in the first Pareto front is lower than or equal to the desired number of representative individuals (num), a local search is carried out on all individuals in the first front without needing to apply K-means [17]. But, if the number of individuals in the first front is greater than num, a K-means is applied to the first front to get the most representative num individuals, who will then be the object of a local search.

The algorithm is finished when the maximum number of generations is reached.

3. GENERALIZED RADIAL BASIS FUNCTION

The original GRBFs were proposed by Francois [15]. Recently different approaches have been introduced to estimate the parameters of this novel model [6, 14]. The GRBF is defined by replacing the square power in the exponent of the standard Radial Basis Function by the τ parameter:

$$B_j(\mathbf{x}, \mathbf{w}_j) = \exp\left(-\left(\frac{\|\mathbf{x} - \mathbf{c}_j\|}{r_j}\right)^{\tau_j}\right),$$

where $\mathbf{w}_j = (\mathbf{c}_j, r_j, \tau_j)$, $\mathbf{c}_j = (c_{j1}, c_{j2}, \ldots, c_{jk})$ is the center of the cluster represented by j-th GRBF transformation, r_j is the corresponding radii or standard deviation, τ_j is the exponent of the basis function, and $c_{ji}, r_j, \tau_j \in \mathbb{R}$. Figure 3 presents the activation for the GRBF with different values of τ. The incorporation of the τ parameter causes the contraction-relaxation of the GRBF curvature. Thanks to the additional τ parameter, the GRBF can define more accurately the membership of the patterns that are located near the decision boundary between clusters.

One problem in high dimensional spaces is that of distances *concentrate*: the range of possible distances is not fully spanned anymore, and most of the patterns are very far from one another [4]. Therefore, the probability of finding patterns near the center, when the dimension is high, is almost zero. In particular GRBFs have been analyzed in high-dimensional problems in [6, 14, 15]. The problem we address in this paper presents sixty-six input variables (categorical variables have been transformed into binary variables, one for each possible category). This problem could be regarded as a high dimensionality classification problem. Therefore, the use of GRBF is totally justified in this problem.

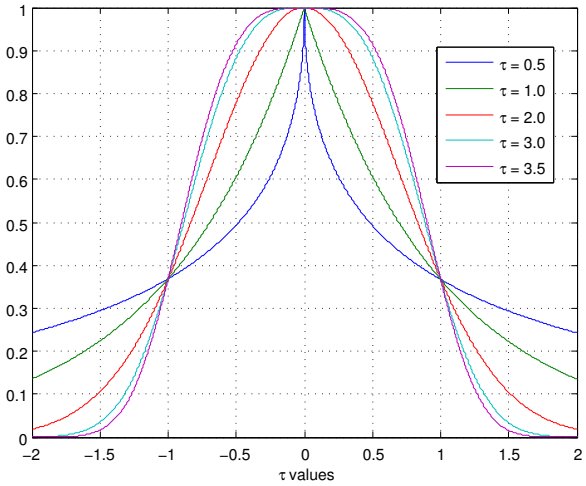

Figure 3: GRBF activation in one-dimensional space with $c = 0$ and $r = 1$ for different τ values.

However, the original formulation of the GRBF based on the τ parameter has some drawbacks in that the same variation in the τ value produces different effects on the GRBF curvature. Figure 3 shows that an increase of 0.5 in the τ value ($\Delta\tau = +0.5$) when $\tau = 1$ produces a significant variation in the GRBF curvature, although when $\tau = 3$, an increase of 0.5 barely modifies GRBF curvature. Furthermore, the τ parameter causes different curvatures for different GRBF radii values.

Because of the drawbacks of the original formulation of the GRBF, we reformulate the GRBF τ parameter as a function compounded by the radii and the α angle formed by the x-axis with the tangent to the GRBF at the point where $x = r$

(Figure 4).

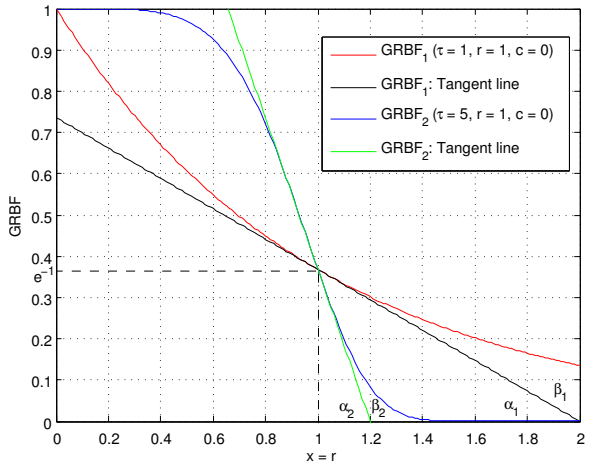

Figure 4: GRBF according to the new α parameter for a one dimensional input space.

The reformulation of the GRBF, from the α angle (Figure 4), is obtained as follows: firstly, we derive the GRBF with respect to the input variable ($c = 0$):

$$\frac{\partial B_j(x, \mathbf{w_j})}{\partial x} = -e^{-(\frac{x}{r})^\tau} \cdot \tau \cdot x^{\tau-1} \cdot r^{-\tau}.$$

Secondly, the derivative at the point $x = r$ is calculated:

$$\tan(\beta) = \left.\frac{\partial B_j(x, \mathbf{w_j})}{\partial x}\right|_{(x=r)} = -\frac{\tau}{e \cdot r}.$$

Finally, taking into account that $\tan(\alpha) = -\tan(\beta)$ (Figure 4) and that the derivative of the GRBF with respect to the input variable in the point $x = r$ is equal to $\tan(\beta)$, the α angle is determined as:

$$\alpha = \arctan\left(\frac{\tau}{e \cdot r}\right).$$

Therefore, the reformulated GRBF is expressed as:

$$B_j(\mathbf{x}, \mathbf{w}_j) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_j\|}{r_j}\right)^{e \cdot r_j \cdot \tan\alpha_j}.$$

4. EXPERIMENTAL STUDY

4.1 Dataset description

In order to solve the matching problem between the liver of the donor and the recipient of the waiting list, the dependent variable chosen is the *graft survival* at 12 months. This is a binary variable equal to 0 when representing *graft survival* (GS) class and 1 when representing *graft non-survival* (GNS) class. Thus, the problem refers to a classification problem. However, the relationship between the dependent variable and the independent/predictor variables is not known in advance. Therefore, as a first step of the method, various machine learning techniques (specifically binary classifiers here) which can conduct classification are implemented to predict the graft survival.

To perform this binary classification problem we have been provided with 1001 patterns (Donor-Recipient (D-R) pairs) obtained along the years 2007 and 2008 in eleven hospitals located throughout the Spanish geography. 19 recipient characteristics, 20 donor characteristics and 3 operative factors were reported for each D-R pair. In this case, we have an unbalanced structure of the dataset since 840 pairs out of the 1001 patterns belong to the GS class and the rest to the GNS class.

To check the accuracy of the models obtained by the evolutionary algorithm, the neural network models are trained using a subset of the dataset (training set) and tested with the rest of it (generalization set). For this purpose, we built a training group of 75% randomly chosen D-R pairs and tested on the remaining group of 25%. During the creation of these two sets, this 75-25% proportion was kept for each of the participating liver transplantation units. That is, 75% of the patterns of each center is used for training and the remaining 25% for generalization. In addition, the proportion of 75-25% is maintained between the patterns of GS class and GNS class.

There is an additional difficulty in this dataset, a high number of patients lose the graft within the first two weeks, indicating that the D-R pairs are not very homogeneous in this class. So the standard classifiers, as discussed in the experimental section, generally classify all or almost all patterns as belonging to the GS class and none to the GNS class. These classifiers are called trivial classifiers. For a given (binary, ordinal, or other) classification problem, a trivial classifier Θ_k may be defined as a classifier that assigns all the pairs to the same class y_k. Accordingly, the trivial class, for a error measure E, (denoted $\tilde{y}$) may be defined as the class that minimizes the chosen error measure E on the training set D across all trivial classifiers, i.e., likewise, the Θ_k trivial-class classifier for E may be defined as that which assigns all D-R pairs to the trivial class for E. A measure such as standard error rate (namely, the fraction of D-R pairs that are incorrectly classified) is not robust to this imbalance, since the majority-class classifier (i.e., the trivial classifier that assigns all D-R pairs to the majority class, GS class, which is the trivial class for error rate) would be deemed extremely "error-free", probably more error-free than any genuinely designed classifier.

Assuming that all misclassifications are equally costly and there is no profit for a correct classification, we assume that a good classifier should obtain a high accuracy level as well as an acceptable level for each class (GS or GNS). In real problems these objectives are usually in competition. Achieving a high accuracy classification level usually means sacrificing the classification in some class.

To solve this difficulty, we consider in this paper a MOEA for neural networks. The MOEA used is the MPDENN defined in Section 2 and is guided by two performance measures of a binary classifier (these measures are described in the following Section).

4.2 Accuracy and Minimum Sensitivity

Considering $D = \{(\mathbf{x}_n, y_n); n = 1, 2, \ldots, N\}$ as a training dataset, where $\mathbf{x}_n = (x_{1n}, \ldots, x_{Kn})$ is the random vector of

measurements taking values in $\Omega \subset R^K$, and y_n is the class level of the n-th individual, we define C by:

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

where $I(\cdot)$ is the zero-one loss function, y_n is the desired output for pattern n and N is the total number of patterns in the dataset. A good classifier tries to achieve the highest possible C in a given problem. However, the C measure is a discontinuous function, which makes convergence very difficult in neural network optimization.

Thus, instead of accuracy, we consider the cross-entropy continuous function, E:

$$E(g, \mathbf{\Theta}) =$$
$$= -\frac{1}{N} \sum_{n=1}^{N} [y_n log(g(\mathbf{x}_n, \mathbf{\Theta})) + (1 - y_n) log(1 - g(\mathbf{x}_n, \mathbf{\Theta}))],$$

where g is neural network model and $g(\mathbf{x}_n, \mathbf{\Theta})$ is the probability that the $\mathbf{x}_n$ pattern belongs to the GS class, calculated using the next softmax activation function:

$$g(\mathbf{x}_n, \mathbf{\Theta}) = \frac{\exp f_l(\mathbf{x}_n, \mathbf{\Theta})}{\sum_{j=1}^{J} \exp f_j(\mathbf{x}_n, \mathbf{\Theta})}, \text{for } l = 1, ..., J,$$

where J is the number of classes in the problem and $f_l(\mathbf{x}_n, \mathbf{\Theta})$ is the output of the l-th output neuron for pattern $\mathbf{x}_n$, defined by:

$$f_l(\mathbf{x}_n, \mathbf{\Theta}) = \beta_0^l + \sum_{j=1}^{M} \beta_j^l B_j(\mathbf{x}_n, \mathbf{w}_j), \text{for } l = 1, ..., J,$$

where $\mathbf{\Theta} = (\beta_0^l, ..., \beta_M^l, \mathbf{w}_1, ..., \mathbf{w}_M)$ is the vector of weights of the output node, M is the number of hidden nodes, $\mathbf{w}_j = \{w_0^j, ..., w_K^j\}$, for $j = 1, ..., M$, is the vector of input weights of the hidden node j and $B_j(\cdot)$ is the GRBF activation function.

Then, we propose a strictly decreasing transformation of the entropy error $E(g, \Theta)$ as the first fitness measure to maximize:

$$A(g) = \frac{1}{1 + E(g, \mathbf{\Theta})}.$$

The second objective to maximize is the minimum sensitivity of the classifier, MS, defined as the minimum value of the sensitivities for each class, $MS = \min\{S_i;\ i = 0, 1\}$. That is, maximizing the lowest percentage of examples correctly predicted as belonging to each class with respect to the total number of examples in the corresponding class.

The minimum sensitivity versus accuracy pair (MS, C) expresses two features associated with a classifier: global performance, C, and the rate of the worst classified class, MS. The selection of MS as a complementary measure of C can be justified upon considering that:

$$C = \frac{f_0}{N} S_0 + \frac{f_1}{N} S_1,$$

is the weighted average of the sensitivities of each of the two classes, and f_i is the size of the C_i class. From a statistical point of view, since C is a weighted average, it will be a good and representative measurement of the set of sensitivities if they are homogeneous enough.

The following inequality is always fulfilled for a classifier g, $MS \leq C \leq 1 - (1 - MS)p*$, where $p*$ is the minimum of the estimated prior probabilities. Therefore, each classifier will be represented as a point in the (MS, E) training plane (see Figure 5). The area outside the triangle is an unfeasible region because of the inequality. Note that it is possible to find among them classifiers with a high level of accuracy, particularly in problems with low $p*$ (unbalanced problems).

A priori, we can think that MS and C objectives can be positively correlated, but while this may be true for small values of MS and C, it is not for values close to 1 on both MS and C. In this way, the objectives are competitive at the top right corner of the triangle. For a more detailed information, see [13].

4.3 Experimental Design

Once the Pareto front is built, two automatic methods are considered in order to construct a neural network model with the information of the models on it. These methods provide us with single models that can be compared with other classification methods existing in the literature. The process followed in these methods is the next one: once the first Pareto front is calculated using the patterns of the training set, the best individual belonging to the Pareto front on Entropy (Ei) and the best individual in terms of Minimum Sensitivity (MSi) are selected. Then the values of C and MS are calculated on the generalization set for the Ei and MSi individuals. Therefore we will have an individual $Ei_G = (C_{EiG}, MS_{EiG})$ and an individual $MSi_G = (C_{MSiG}, MS_{MSiG})$. This is repeated 30 times and then the average and standard deviation obtained from the individuals is estimated, $\overline{Ei}_G = (\overline{C}_{EiG}, \overline{MS}_{EiG})$, $\overline{MSi}_G = (\overline{C}_{MSiG}, \overline{MS}_{MSiG})$. The first expression is the average obtained taking E into account as the primary objective, and the second taking MS into account as the primary objective. So, the opposite extremes of the Pareto front are taken in each of the executions, and the first automatic procedure is called MPDENN-C (training with E and generalization with C) and the second MPDENN-MS (Minimum Sensitivity). In Figure 5, the process is shown graphically.

4.4 Statistical Analysis

We have compared the MPDENN algorithm with 7 state-of-the-art methods well known in the literature. 6 of this algorithms have been configured and runned in WEKA[1] [25], and the *LibSVM*[2] method is available in a website as a continuously updated software library for Support Vector Machines [7]. The methods used for comparison are:

MLP: A neural network classifier that uses backpropagation to adjust the weights. The nodes in this network are all sigmoid. This method is stochastic, so that its execution has been repeated 30 times with different seeds. Weights are updated in 0.3 and momentum applied to the weights is 0.2.

[1] http://www.cs.waikato.ac.nz/ml/weka/
[2] http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

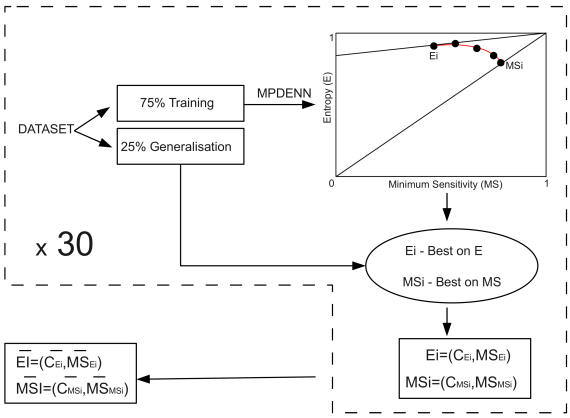

Figure 5: Scheme to obtain statistical results.

C4.5: Classifier for generating a pruned or unpruned C4.5 decision tree. The confidence factor used for pruning is 0.25 and the minimum number of instances per leaf is 2.

LMT: Classifier for building logistic model trees, which are classification trees with logistic regression functions at the leaves. The algorithm can deal with binary and multi-class target variables, numeric and nominal attributes and missing values. The minimum number of instances for a node to be considered for splitting is set to 15 and the beta value used for weight trimming is 0.

NaivesBayes: It is a classifier whose numeric estimator precision values are chosen based on analysis of the training data.

SLogistic: Classifier for building linear logistic regression models. LogitBoost with simple regression functions as base learners is used for fitting the logistic models. The optimal number of LogitBoost iterations is cross-validated, which leads to automatic attribute selection. The maximum of iterations is set to 500.

MLogistic: Classifier for building a multinomial logistic regression model with a ridge estimator. The ridge value used in the log-likelihood is $1.0 * 10^{-8}$.

LibSVM: This is a software package for the optimization of Support Vector Machines (SVM). This library contains a script for automatically adjusting the hyper-parameters associated to this kind of models, including the cost parameter and the width of the Gaussian kernels. The library searches the best hyper-parameter values using a grid search and choosing the best configuration by a 10-fold cross-validation process (this cross-validation process is guided by the AUC measure).

For MPDENN, the population size is established at $M = 25$. The crossover probability is 0.8 and the mutation probability is 0.1. For iRprop$^+$, the parameters adopted are $\eta^+ = 1.2$, $\eta^- = 0.5$, $\Delta_0 = 0.0125$ (the initial value of the Δ_{ij}), $\Delta_{\min} = 0$, $\Delta_{\max} = 50$ and $Epochs = 5$, see [16] for iRprop$^+$ parameter description. The optimization process is applied 3 times during execution (every 33.33% generations) and uses $num = 5$ cluster in the clustering algorithm. To start processing data, each one of the input variables was scaled in the ranks $[-1.0, 1.0]$ to avoid the saturation of the signal. In addition, categorical variables have been transformed into so many binary variables as possible categories.

To analyze the robustness of the proposed method, we used C, MS and others 3 metrics of comparison [3, 5]:

AUC **or Area Under Curve:** The AUC of a binary classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance

$$AUC = \frac{1}{N_0 N_1} \sum_{n=1}^{N_0} \sum_{m=1}^{N_1} c(\mathbf{p}_n, \mathbf{p}_m),$$

where N_0 is the number of examples for GS class and N_1 for GNS class, $\mathbf{p}_n = [p_{0n}, p_{1n}]$ the estimated probability vector for members of GS class and $\mathbf{p}_m = [p_{0m}, p_{1m}]$ the estimated probability vector for members of GNS class, being $c(\mathbf{p}_n, \mathbf{p}_m) = 1$ if $p_{0n} > p_{0m}$ and 0 in otherwise.

$KAPPA$**:** This is originally a measure of agreement between two classifiers, although it can also be employed as a classifier performance measure.

$$KAPPA = \frac{p(A) - p(E)}{1 - p(E)},$$

where $p(A)$ is the relative observed agreement among classifiers, and $p(E)$ is the probability that agreement is due to chance. In this case, $p(A)$ is just the accuracy of the classifier.

$RMSE$ **or Root Mean Square Error:** It measures how much predictions deviate from the true targets.

$$RMSE = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2},$$

where y_n is the real value and $\hat{y}_n$ is the estimated value.

5. RESULTS

Table 1 show the results obtained with each method. From a descriptive point of view, MPDENN-C, LMT, LibSVM and SLogistic methods obtain similar results in C measure, MPDENN-C being the best model (with a value of 84.46%). For the MS metric, the best result is obtained by MPDENN-MS method (45.55%) followed by MLP (14.63%). The result obtained by the best MPDENN-MS model is very interesting because it has a minimum sensitivity of 56.10% (much higher than that obtained by the other methods). The best AUC value is obtained by NaiveBayes (0.6430) and the second by the best MPDENN-C model (0.6134); other methods give similar results (except C.45). The results obtained in $KAPPA$ metric are very similar, emphasizing that obtained by the best MPDENN-MS model (0.1119) and by the best MLP model (0.1004). For the $RMSE$ metric, the best MPDENN-C model obtains the best result (0.3607) followed by LibSVM (0.3698).

Table 1: Results obtained in mean and standard deviation for different methods using the metrics C, MS, AUC, $KAPPA$ and $RMSE$.

Generalization results					
Method	$\mathbf{C_G}$(%)	$\mathbf{MS_G}$(%)	$\mathbf{AUC_G}$	$\mathbf{KAPPA_G}$	$\mathbf{RMSE_G}$
Best MPDENN-C model	**84.46**	4.88	*0.6134*	0.0790	**0.3607**
Best MPDENN-MS model	60.96	**56.10**	0.5743	**0.1119**	0.4663
Best MLP model	80.47	*14.63*	0.5530	*0.1004*	0.4164
C4.5	82.07	0.00	0.4720	-0.0299	0.3940
LMT	*83.66*	0.00	0.5840	0.0000	0.3703
NaiveBayes	79.68	7.32	**0.6430**	0.0149	0.4055
SLogistic	*83.66*	0.00	0.5840	0.0000	0.3703
MLogistic	82.86	0.00	0.5650	-0.0154	0.3764
LibSVM	*83.66*	0.00	0.5957	0.0000	*0.3698*
Method	$\mathbf{C_G}$(%) Mean±SD	$\mathbf{MS_G}$(%) Mean±SD	$\mathbf{AUC_G}$ Mean±SD	$\mathbf{KAPPA_G}$ Mean±SD	$\mathbf{RMSE_G}$ Mean±SD
MPDENN-C	83.66 ± 0.23	0.81 ± 1.48	0.5818 ± 0.0411	0.0106 ± 0.0222	0.3690 ± 0.0035
MPDENN-MS	59.61 ± 5.44	45.55 ± 9.26	0.5627 ± 0.0426	0.0563 ± 0.0486	0.4898 ± 0.0286
MLP	76.52 ± 1.97	11.45 ± 4.30	0.5120 ± 0.0310	0.0087 ± 0.0609	0.4510 ± 0.0141

The best result is in **bold** face and the second best result in *italics*.

Several methods used achieve the same value of C (83.66%), MS (0.00%) and $KAPPA$ (0.0000). This is because all of these methods results in trivial classifiers.

6. CONCLUSIONS

In this paper, we present generalized radial basis functions neural networks models that can help medical experts in the donor-recipient allocation. These models are obtained by a multi-objective evolutionary algorithm guided by the Accuracy and the Minimum Sensitivity measures. Minimum Sensitivity is used to avoid the design of models with high global performance but bad performance when considering the classification rate for each class (*survival* or *non-survival*). Several methods, which do not take into account the minimum sensitivity while training, obtained trivial classifiers. These trivial classifiers are useless to solve any problem and, especially, one of biomedicine.

The results obtained with the best models of MPDENN-E and MPDENN-MS methods suggest that a combination of both would provide a useful tool for the problem of donor-recipient assignment. This combination could be a rule-based system, which would provide an understandable and comprehensible tool for medical experts. The system would receive as input a set of potential recipients and form a donor-recipient pair between each of them and donor/organ data. These pairs would be the input for these neural network models. With the results provided by these models and using a simple set of rules, the system would determine which of the recipients should receive the organ.

7. ACKNOWLEDGMENTS

This work has been partially subsidized by the TIN 2008-06681-C06-03 project of the Spanish Ministerial Commission of Science and Technology (MICYT), FEDER funds and the P08-TIC-3745 project of the "Junta de Andalucía" (Spain). M. Cruz-Ramírez's research has been subsidized by the FPU Predoctoral Program (Spanish Ministry of Education and Science), grant reference AP2009-0487. Francisco Fernández-Navarro's research has been funded by the "Junta de Andalucía" Predoctoral Program, grant reference P08-TIC-3745.

Finally, we would like to thank Astellas Pharma Company for partial support.

8. REFERENCES

[1] H. A. Abbass, R. Sarker, and C. Newton. PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems. In *Proceedings of the 2001 Congress on Evolutionary Computation*, volume 2, Seoul, South Korea, 2001.

[2] M. Astion and P. Wilding. Application of neural networks to the interpretation of laboratory data in cancer diagnosis. *Clin Chem*, 38:34–38, 1992.

[3] A. Ben-David. Comparison of classification accuracy using Cohen's Weighted Kappa. *Expert Systems with Applications*, 34(2):825–832, 2008.

[4] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is Nearest Neighbor meaningful? In *International Conference on Database Theory*, pages 217–235, 1999.

[5] R. Caruana and A. Niculescu-Mizil. Data mining in metric space: An empirical analysis of supervised learning performance criteria. In *Proceedings of the 10th International Conference in Knowledge Discovery and Data Mining*, pages 69–78, Seattle, USA, 2004.

[6] A. Castaño, F. Fernández-Navarro, C. Hervás-Martínez, P. A. Gutierrez, and M. M. García. Classification by Evolutionary Generalized Radial Basis Functions. *International Journal of Hybrid Intelligent Systems*, 7(1):1–10, 2010.

[7] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

[8] H.-Y. Chen, T.-A. Chen, D. Min, G. Fisher, and Y.-M. Wu. Prediction of tacrolimus blood levels by using the neural network with genetic algorithm in liver transplantation patients. *Ther Drug Monit*, 21:50–56, 1999.

[9] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving*

Multi-Objective Problems (Genetic and Evolutionary Computation). Springer, 2nd edition, September 2007.

[10] M. Cruz-Ramírez, J. Sánchez-Monedero, F. Fernández-Navarro, J. Fernández, and C. Hervás-Martínez. Memetic Pareto differential evolutionary artificial neural networks to determine growth multi-classes in predictive microbiology. *Evolutionary Intelligence*, 3(3-4):187–199, 2010.

[11] I. Dvorchik, M. Subotin, W. Marsh, J. McMichael, and J. Fung. Performance of multi-layer feedforward neural networks to predict liver transplantation outcome. *Methods Inf Med*, 35:12–18, 1996.

[12] J. C. Fernández, C. Hervás, F. J. Martínez, P. A. Gutiérrez, and M. Cruz. Memetic Pareto differential evolution for designing artificial neural networks in multiclassification problems using cross-entropy versus sensitivity. In *Hybrid Artificial Intelligence Systems*, volume 5572, pages 433–441. Springer Berlin / Heidelberg, 2009.

[13] J. C. Fernández-Caballero, F. J. Martínez-Estudillo, C. Hervás-Martínez, and P. A. Gutiérrez. Sensitivity versus accuracy in multiclass problems using memetic pareto evolutionary neural networks. *IEEE Trans. on Neural Networks*, 21(5):750 –770, 2010.

[14] F. Fernández-Navarro, C. Hervás-Martínez, J. Sánchez-Monedero, and P. A. Gutierrez. MELM-GRBF: A modified version of the Extreme Learning Machine for Generalized Radial Basis Function Neural Networks. *Neurocomputing*, 2010. In press.

[15] D. Francois. *High dimentional Data Analisis, From Optimal Metric to Feature Selection*, chapter Seeking on right metric, pages 54 – 55. VDM Verlag, Saarbrucken, Germany, 2008.

[16] C. Igel and M. HÃijsken. Empirical evaluation of the improved rprop learning algorithms. *Neurocomputing*, 50(6):105–123, 2003.

[17] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. U. C. Berkeley Press, 1967.

[18] S. Matis, H. Doyle, I. Marino, R. Mural, and E. Uberbacher. Use of neural networks for prediction of graft failure following liver transplantation. *IEEE Symposium on Computer-Based Medical Systems*, 0:133–140, 1995.

[19] G. Molino and A. Arrigoni. Design of a computer-assisted programme supporting the selection and clinical management of patients referred for liver transplantation. *Ital J Gastroenterol*, 26:31–43, 1994.

[20] S. Pedersen, J. Jorgensen, and J. Pedersen. Use of neural networks to diagnose acute myocardial infarction. II. A clinical application. *Clin Chem*, 42:613–617, 1996.

[21] K. Prank, C. Jurgens, A. von zur Muhlen, and G. Brabant. Predictive neural networks for learning the time course of blood glucose levels from the complex interaction of counterregulatory hormones. *Neural Comput*, 10:941–953, 1998.

[22] P. Sharpe, H. Solberg, K. Rootwet, and M. Yearworth. Artificial neural networks in diagnosis of thyroid function from in vitro laboratory tests. *Clin Chem*, 39:2248–2253, 1993.

[23] D. Sheppard, D. McPhee, C. Darke, B. Shrethra, R. Moore, A. Jurewitz, and A. Gray. Predicting cytomegalovirus disease after renal transplantation: an artificial neural network approach. *Int J Med Inf*, 54(1):55–76, 1999.

[24] R. Storn and K. Price. Differential evolution. A fast and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[25] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Data Management Systems. Morgan Kaufmann (Elsevier), 2nd edition, 2005.

View publication stats