EMORBFN: An Evolutionary Multiobjetive Optimization algorithm for RBFN design

Pedro L. López, Antonio J. Rivera, M. Dolores Pérez-Godoy, María J. del Jesus, Cristóbal Carmona

Department of Computer Science. University of Jaén. Jaén (Spain)
pllc0001@estudiantes.ujaen.es; {arivera, lperez, mjjesus, ccarmona}@ujaen.es

Abstract. In this paper a multiobjective optimization algorithm for the design of Radial Basis Function Networks is proposed. The goal of the design algorithm is to obtain networks with a high tradeoff between accuracy and complexity, overcoming the drawbacks of the traditional single objective evolutionary algorithms. The main features of EMORBFN are a selection mechanism based on NSGA-II and specialized operators. To test the behavior of EMORBFN a similar mono-objective optimization algorithm for Radial Basis Function Network design has been developed. Also C4.5, a Multilayer Perceptron network or an incremental method to design of Radial Basis Function Networks have been included in the comparison. Experimental results on six UCI datasets show that EMORBFN obtains networks with high accuracy and low complexity, outperforming other more mature methods.

Keywords. Evolutionary Multi-objective Optimization, Radial Basis Function Networks, Classification.

1 Introduction

Radial Basis Function Networks (RBFNs) are one of the most important Artificial Neural Network (ANN) paradigms in the machine learning design field. An RBFN is a feed-forward ANN with a single layer of hidden units, called radial basis functions (RBFs). The first research on neural networks based on RBFs [4] was carried out at the end of the eighties. Since then, the overall efficiency of RBFNs has been proved in many areas like pattern classification [5], function approximation [13] and time series prediction [20].

The main features of a RBFN are a simple topological structure, with only one hidden layer, of neurons/RBFs that have a particular locally-tuned response that depends on the center and the width (radius) of each RBF. Also, they have universal approximation capability [13].

The goal of any RBFN design process is to determine centers, widths and the linear output weights connecting the RBFs to the output neuron layer. The most traditional learning procedure has two stages: first, unsupervised learning of centers and widths is used, and finally output weights are established by means of supervised learning. Clustering techniques [14] are commonly used to adjust the centers. Regarding the

widths, they may all be given the same value, may reflect the width of the previously calculated clusters (i.e., RBFs), or may be established as the average distance between RBFs, among other possibilities. In order to obtain the weights in the second stage, algorithms such as Least Mean Square (LMS) [21] can be used.

Another important paradigm concerning RBFN design is Evolutionary Computation (EC) [3][10], a general stochastic optimization framework inspired by natural evolution. Typically, in this paradigm each individual represents a whole network that is evolved in order to increase its accuracy.

Managing only the accuracy objective to optimize RBFNs, like in traditional evolutionary computation, may lead to obtain nets with a high number of RBFs. This is because it is easier to reduce error with those RBFNs having many RBFs than with those having few neurons. In order to overcome this drawback, Evolutionary Multiobjective Optimization (EMO)[6], can be used. Among the different multiobjective methods, Nondominated Sorting Genetic Algorithm NSGA-II [7] is one of the most efficient and used techniques.

In this paper, a new evolutionary multiobjective optimization algorithm for the design of RBFNs (EMORBFN) applied to classification problems is presented. This algorithm follows the basic lines of a standard evolutionary algorithm but uses a NSGAII-based strategy to select the population of the new generation. In order to promote diversity, new operators to recombine and mutate individuals of the population are introduced. Other important characteristic of EMORBFN is the use of a dissimilarity measure (HVDM)[22] to calculate distances between nominal attributes. To test the performance of the proposed algorithm a traditional monoobjective evolutionary algorithm for the design of RBFNs (EORBFN) has been developed. Also, other well-know classification algorithms such as an RBFN-incremental algorithm, C4.5 or a Multilayer Perceptron Network have been used to compare the results of our proposed method.

The rest of paper is organized as follows: in section 2, a revision of the multiobjective optimization of RBFNs is shown. The EMORBFN algorithm is explained in section 3. The experimentation and the analysis of the results are shown in section 4. Conclusions are outlined in section 5.

2 Evolutionary design of RBFNs

An RBFN is a feed-forward neural network with three layers: an input layer with n nodes, a hidden layer with m neurons or RBFs, and an output layer with one or several nodes (Figure 1). The m neurons of the hidden layer are activated by a radially-symmetric basis function, $\phi_i: R^n \rightarrow R$, which can be defined in several ways.

From all the possible choices for ϕ_i, the Gaussian function is the most widely used: $\phi_i(\vec{x}) = \phi_i(e^{-(\|\vec{x}-\vec{c}_i\|/d_i)^2})$, where $\vec{c}_i \in R^n$ is the center of basis function ϕ_i, $d_i \in R$ is the width (radius), and $\| \ \|$ is usually the Euclidean norm on R^n. The output of one basis function will be high when the input vector and the center of this basis function are close, always taking into account the value of the radius. The weights w_{ij} show the

contribution of an RBF to the respective output node, and therefore output nodes implement the weighted sum of RBF outputs (equation (1)).

$$f_j(\vec{x}) = \sum_{i=1}^{m} w_{ij}\phi_i(\vec{x}) \quad (1)$$

Nowadays, Evolutionary Computation [3] is one of the most important paradigms in the design of RBFN [10][5]. Typically, in these algorithms, the individuals are codified using the Pittsburgh scheme, where an individual represents a whole RBFN. In the other hand, a cooperative-competitive evolutionary approach represents an RBF in each individual [16].

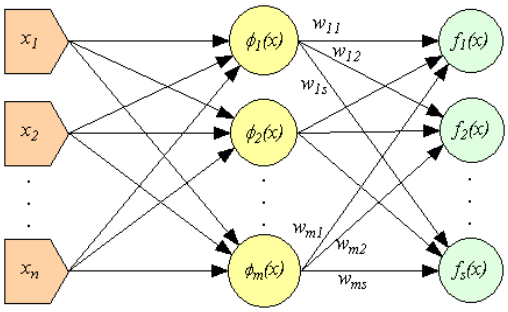

Fig. 1. RBFN Topology

Most of Pittsburgh methods only optimize the accuracy of the individuals/RBFNs without having into account the complexity of the networks obtained. As have been said, managing only the objective of the accuracy may lead to obtaining nets with a high number of RBFs. That's because it is easier to get less error in RBFNs with many RBFs than in RBFNs with few neurons. Evolutionary Multiobjective Optimization [6] can be used in order to optimize several objectives such as the accuracy and complexity (the number of RBFs) of the models.

Standard Multiobjective Evolutionary Algorithms (MOEAs) have been used in the design of RBFNs for estimation or function approximation, such as Multiobjective Genetic Algorithms MOGAs in [8] or [19]. [17] hybrids NSGA-II with rough-sets and surrogate techniques in order to achieve a more efficient multiobjective algorithm. [11] uses spatially distributed surrogates (RBFNs) in a NSGA-II framework. In [12] simple RBFNs (i.e.: with the same width for all RBFNs), obtained by a NSGA-II algorithm, are used as ensembles in the final model proportioned by the algorithm. In a more specific application regarding estimations, [9] uses the NSGA-II in the mineral reduction environment. Also, there are specific MOEAs such as [23] that present a Hierarchical Rank Density Genetic Algorithm (HRDGA) to evolve the neural network's topology and parameters simultaneously.

3 EMORBFN: Evolutionary Multiobjetive Optimization algorithm for RBFN design

Basically, EMORBFN is an Evolutionary Multiobjective algorithm for the design of RBFN. It follows the main steps shown in Fig. 2. In step 6, in order to create the next population, the NSGA-II algorithm is used. The objectives taken into account to optimize and sort the individuals are their accuracy and complexity (number or neurons). The implemented codification is real and Pittsburgh-based, where each individual represents a whole network. A dissimilarity measure HVDM distance [22] has been chosen in order to work with nominal attributes. Bellow, the main steps of EMORBFN are explained.

1. P_t = Initialization (Data)
2. Training/Evaluation (P_t)
3. R_t = Recombination (P_t)
4. M_t = Mutation (P_t)
5. Training/Evaluation (R_t, M_t)
6. P_{t+1} = Fill_next_population (P_t, R_t, M_t)
7. If the stop-condition is not verified go to step 2

Fig. 2. Main steps of EMORBFN

RBFN initialization.- For each initial network, a random number, *m*, of neurons (i.e. the size of population) is also randomly allocated among the different classes of the training set. Each RBF center, $\vec{c}_i$, is randomly established to a pattern of the training set, taking into account that the RBFs must be distributed equally among the different classes. The RBF widths, d_i, will be set to half of the average distance between the centers. Finally, the RBF weights, w_{ij}, are set to zero.

Training/evaluation.- During this stage, the well-known LMS algorithm [21] has been used to calculate the RBF weights of each individual. At this moment, the RBFN is evaluated, obtaining its classification error.

Recombination.- With the crossover operator two individuals/RBFNs parents are randomly chosen to obtain one RBFN offspring. The number of RBFs of the new individual will be delimited between a minimum and a maximum value. The minimum value is set to the number of RBFs of the parent with fewer RBFs. The maximum value is set to sum of the number of RBFs of both parents. In order to generate the offspring, RBFs will be chosen in a random way from the parents.

Mutation.- Six mutation operators, usually considered in the specialised bibliography [10] have been implemented. They can be classified as random operators or biased operators. The random operators are:

- *DelRandRBFs*: randomly eliminates a *pm* percent of the total number of RBFs in the RBFN.

- *InsRandRBFs*: randomly adds a *pm* percent of the total number of RBFs in the RBFN.
- *ModCentRBFs*: randomly modifies the center of *pm* percent of the total number of RBFs in the RBFN. The center of the basis function will be modified in a *pr* percent of its width.
- *ModWidtRBFs*: randomly modifies a *pm* percent of the total number of RBFs in the RBFN. The width of the basis function will be modified in a *pr* percent of its width.

Biased operators, which exploit local information, are:

- *DelInfRBFs*: deletes a *pm* percent of the total number of RBFs in the RBFN.
- *InsInfRBFs*: inserts a *pm* percent of the total number of RBFs in the RBFN.

Filling the next population.- In order to create the new population, the basic lines of NSGA-II have been followed. In this way the rank of parents P_t and offspring (Q_t, R_t) is calculated depending on the values of their objectives by means of a dominance count. Next, the all non-dominated fronts $F = \{F_1, F_2, \ldots\}$ are generated, where individuals in front F_1 have less ranking value that individuals in F_2 and so on. The new population is filled inserting the fronts F_1, F_2, … until the maximum number of the individuals of the population is reached. If the number of the individuals of the last front (F_l) is greater than the number of individuals needed to fill P_{t+1}, then crowding distance is calculated for the individuals of F_l. Thus, individuals with large crowding measure will complete P_{t+1}.

4 Experimentation and results

To test our multiobjective method against a monoobjective based proposal, a typical Evolutionary algorithm for the Optimization of RBFNs, EORBFN has been developed. The design lines of EORBFN are the classical ones for this kind of algorithms [10]. In order to promote an adequate comparison between EMORBFN and EORBBFN, similar operating conditions have been established. In this way EORBFN and EMORBFN have the same pseudocode, codification scheme (Pittsburgh), operators and dissimilarity measure (HVDM). Obviously, the main differences can be found in step 6 when the new population is created. To fill the new population P_{t+1} from the populations of parents (P_t) and offspring (Q_t, R_t) a tournament selection mechanism is considered. The diversity of the population is promoted by using a low value for the tournament size (k = 3). The objective considered in this evolutionary algorithm is to minimize the classification error.

In order to increase the efficiency of the evolutionary algorithms, the search space of this method has been restricted. In this experimentation, the search space has been reduced by fixing the complexity (number of RBFs) between a minimum and a maximum number of neurons. The minimum number of neurons has been set to the number of classes for the problem and the maximum to four times this number. The parameter values used for experimentation with EMORBFN and EORBFN are shown in Table 1.

Table 1. Experimentations parameters used for EMORBFN and EORBFN

Parameter	Value
Iterations of the main loop	200
Individuals	40
Crossover probability	0.6
Mutation probability	0.1
Pm	0.2
Pr	0.2

Table 2. Results with Hepatitis dataset

Algorithm	*#nodes/ Rules*	*Classification rate (%)*
C4.5	7.1	**89.647**
MLP-Back	30.0	71.817
RBFN-Incr	120.1	76.637
EORBFN	7.5	86.711
EMORBFN	**4.0**	87.598

Table 3. Results with Iris dataset

Algorithm	*#nodes/ rules*	*Classification rate (%)*
C4.5	4.8	94.000
MLP-Back	30.0	64.667
RBFN-Incr	29.8	94.667
EORBFN	10.4	95.067
EMORBFN	**4.0**	**95.789**

Table 4. Results with Pima dataset

Algorithm	*#nodes/ Rules*	*Classification rate (%)*
C4.5	18.3	73.972
MLP-Back	30.0	70.819
RBFN-Incr	671.4	67.728
EORBFN	7.6	75.615
EMORBFN	**5.0**	**75.683**

Table 5. Results with Sonar dataset

Algorithm	*#nodes/ rules*	*Classification rate (%)*
C4.5	14.3	71.071
MLP-Back	30.0	67.762
RBFN-Incr	159.8	74.976
EORBFN	**7.7**	73.305
EMORBFN	8.0	**75.536**

Table 6. Results with Wbcd dataset

Algorithm	*#nodes/ rules*	*Classification rate (%)*
C4.5	12.4	94.995
MLP-Back	30.0	87.722
RBFN-Incr	319.9	94.303
EORBFN	6.2	96.713
EMORBFN	**4.0**	**97.535**

Table 7. Results with Wine dataset

Algorithm	*#nodes/ rules*	*Classification rate (%)*
C4.5	5.1	94.902
MLP-Back	30.0	93.301
RBFN-Incr	125.3	74.739
EORBFN	10.4	95.275
EMORBFN	**3.0**	**95,708**

EMORBFN has been compared with other traditional classification methods such as: C4.5 [15], an algorithm that creates classification rules in the form of decision trees from a dataset. MLP-Back [18], an algorithm for Multilayer Perceptron Networks design which uses the Backpropagation algorithm for learning. RBFN-Incr [1], an algorithm for the RBFNs design based on an incremental scheme. The implementation of these algorithms has been obtained from KEEL [1].

The collection of data sets used in this section (Hepatitis, Iris, Pima, Sonar, Wbcd, Wine) was obtained from the UCI Repository of Machine Learning Database [2]. The parameters used for C4.5, MLP-Back and RBFN-Incr are the ones proposed by the authors. In order to estimate the precision we use a ten-fold cross validation approach, that is, ten partitions for training and test sets, 90% for training and 10% for testing,

where the ten test partitions form the whole set. For each dataset we consider the average results of the ten partitions.

From Table 2 to Table 7 the classification test rate of the methods and their corresponding complexities (number of nodes -RBFs- in the RBFN or number of the rules for C4.5) are shown. For the EMORBFN algorithm, the best accuracy network obtained is chosen.

From these results it can be inferred that EMORBFN is the best method in accuracy in five of the six datasets; only in Hepatitis dataset, C4.5 outperforms EMORBFN in only two points. Despite this fact, EMORBFN obtains the second best result in accuracy for this dataset. Regarding the complexity of the models obtained, EMORBFN achieves the model with the lowest complexity in five of the six datasets. Only EORBFN obtains a model with lower complexity in Sonar Dataset. But again, EMORBFN obtains the second best result in complexity for this dataset.

Only attending to the results obtained by EMORBFN and EORBFN can be observed that: the results of EORBFN regarding accuracy are similar to the EMORBFN results; but studying complexity, EORBFN obtains more complex models that EMORBFN. Also, EORBFN models often reach the maximum limit of RBFs, set in its parameters. This fact confirms the hypothesis that EMORBFN overcomes the drawbacks of EORBFN.

5 Conclusions

In this paper a new evolutionary multiobjective algorithm for RBFN design, EMORBFN, has been presented. EMORBFN implements the basic pseudocode of any evolutionary algorithm but, in order to fill the population of the next generation, a NSGAII-based technique is used. In order to promote diversity, new operators to recombine and mutate individuals of the population are introduced. Another important characteristic of EMORBFN is the use of a dissimilarity measure (HVDM) to calculate distances between nominal attributes.

To analyze the behavior of EMORBFN, a traditional evolutionary monoobjective optimization algorithm (EORBFN) for RBFN design has been developed. EORBFN has the same operating elements that EMORBFN but only optimizes the accuracy of the networks. Also, the results for other well-known classification methods, such as C4.5, MLP-BackPropagation and an incremental RBFN design method have been obtained and used in the comparison.

An analysis of the results shows that EMORBFN mostly outperforms the other proposals, both in accuracy and complexity. EORBFN achieves competitive accuracy results but the models obtained reach the maximum limit of RBFs set in the parameters. This fact confirms our initial hypothesis and demonstrates that EMORBFN overcomes the drawbacks of traditional evolutionary design algorithms.

This work has been partially supported by the CICYT Spanish Projects TIN2008-06681-C06-02 and TIN2007-60587.

References

1. Alcalá-Fdez J, Sánchez L, García S, Del Jesus MJ, Ventura S, Garrell JM, Otero J, Romero C, Bacardit J, Rivas VM, Fernández JC, Herrera F. KEEL: A Software Tool to Assess Evolutionary Algorithms to Data Mining Problems. Soft Comput. Doi: 10.1007/s00500-008-0323-y (2008)
2. Asuncion A, Newman DJ. UCI Machine Learning Repository [http://www.ics.uci.edu/ ~mlearn/MLRepository.html]. Irvine, CA: University of California (2007)
3. Bäck T, Hammel U, Schwefel H Evolutionary computation: comments on the history and current state. IEEE T Evolut Comput 1:1:3-17. (1997)
4. Broomhead D, Lowe D. Multivariable functional interpolation and adaptive networks. Complex System 2:321-355 (1988)
5. Buchtala O, Klimek M, Sick B. Evolutionary optimization of radial basis function classifiers for data mining applications. IEEE T Syst Man Cy B 35:5:928 – 947 (2005)
6. Deb K. Multi-objective optimization using evolutionary algorithms. Wiley, 1st ed. (2001)
7. Deb K, Pratap A, Agarwal, S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. on Evolutionary Computation 6 (2), pp. 182-197 (2002)
8. González J, Rojas I, Ortega J, Pomares H, Fernández FJ, Díaz AF. Multiobjective evolutionary optimization of the size, shape, and position parameters of radial basis function networks for function approximation. IEEE T Neural Networ 14:6:1478-1495 (2003)
9. Guillén A, Pomares H, Rojas I, González J, Herrera LJ, Rojas F, Valenzuela O. Output value-based initialization for radial basis function neural networks. Neural Process Lett DOI: 10.1007/s11063-007-9039-8. (2007)
10. Harpham C, Dawson C, Brown M. A review of genetic algorithms applied to training radial basis function networks. Neural Comput Appl 13:193-201 (2004)
11. Isaacs, A., Ray, T., Smith, W. An evolutionary algorithm with spatially distributed surrogates for multiobjective optimization . LNAI 4828, pp. 257-268 (2007)
12. Kondo, N., Hatanaka, T., Uosaki, K. Pattern classification via multi-objective evolutionary RBF networks ensemble 2006 SICE-ICASE art. no. 4108811, pp. 137-142 (2006)
13. Park J, Sandberg I. Universal approximation using radial-basis function networks. Neural Comput 3:246-257 (1991)
14. Pedrycz W. Conditional fuzzy clustering in the design of radial basis function neural networks. IEEE T Neural Networ 9:4:601-612 (1998)
15. Quinlan JR. C4.5: Programs for Machine Learning. Morgan Kauffman. (1993)
16. Rivera AJ, Rojas I, Ortega J, del Jesus MJ. A new hybrid methodology for cooperative-coevolutionary optimization of radial basis function networks. Soft Comput Doi: 10.1007/s00500-006-0128-9 (2007)
17. Santana-Quintero LV, Coello CA, Hernández-Díaz AG. Hybridizing surrogate techniques, rough sets and evolutionary algorithms to efficiently solve multi-objective optimization problems. GECCO'08 pp. 763-764 (2007)
18. Rojas R, Feldman J. Neural Networks: A Systematic Introduction. Springer (1996)
19. Teixeira CA, Ruano MG, Ruano AE, Pereira WCA. A soft-computing methodology for non invasive time-spatial temperature estimation. IEEE T Bio-Med Eng 55:2:572-580 (2008)
20. Whitehead B, Choate T. Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction. IEEE T Neural Networ 7:4:869-880 (1996)
21. Widrow B, Lehr MA. 30 Years of adaptive neural networks: perceptron, madaline and backpropagation. Proc IEEE 78:9:1415-1442 (1990)
22. Wilson DR, Martinez TR. Improved heterogeneous distance functions. J Artif Intell Res 6:1:1-34. (1997)
23. Yen GG. Multi-Objective evolutionary algorithm for radial basis function neural network design. Studies in Computational Intelligence 16:221-239 (2006)