Applied Intelligence 22, 167–181, 2005

Evolutionary Radial Basis Functions for Credit Assessment

ESTEFANE LACERDA
Center of Informatics, Federal University of Pernambuco, Brazil
egml@cin.ufpe.br

ANDRÉ C. P. L. F. CARVALHO
ICMC, University of São Paulo, Brazil
andre@icmc.usp.br

ANTÔNIO PÁDUA BRAGA
Department of Electronics, Federal University of Minas Gerais, Brazil
apbraga@cpdee.ufmg.br

TERESA BERNARDA LUDERMIR
Center of Informatics, Federal University of Pernambuco, Brazil
egml@cin.ufpe.br

Abstract. Credit analysts generally assess the risk of credit applications based on their previous experience. They frequently employ quantitative methods to this end. Among the methods used, Artificial Neural Networks have been particularly successful and have been incorporated into several computational tools. However, the design of efficient Artificial Neural Networks is largely affected by the definition of adequate values for their free parameters. This article discusses a new approach to the design of a particular Artificial Neural Networks model, RBF networks, through Genetic Algorithms. It presents an overall view of the problems involved and the different approaches employed to optimize Artificial Neural Networks genetically. For such, several methods proposed in the literature for optimizing RBF networks using Genetic Algorithms are discussed. Finally, the model proposed by the authors is described and experimental results using this model for a credit risk assessment problem are presented.

Keywords: RBF networks, neural networks, genetic algorithms

1. Introduction

In recent years, there has been a large increase in the number of finance institutions that employ analysis tools based on Artificial Intelligence techniques, among them, Artificial Neural Networks (ANNs) [1–6]. There is a wide range of financial applications using ANNs. This paper addresses the problem of credit risk assessment, which is essentially a classification problem involving the evaluation of reliability and profitability of a credit application. To this end, Radial Basis Function (RBF) [7] networks are investigated as an alternative for solving credit risk assessment problems.

The performance of RBF networks strongly depends on their topology and learning parameters. The determination of these parameters has a considerable effect on the ANN behavior, as measured by learning time, accuracy, precision, noise tolerance and generalization capability. Thus, these parameters must be adequately set in order to assure an efficient performance.

The optimization of RBF networks has been achieved through a number of different techniques,

such as the Orthogonal Least Square (OLS) [8], Resource Allocating Network (RAN) [9] and Regularization Theory [10]. Evolutionary approaches have been presented in the literature as a global search method for the design of RBF networks [11–13]. The need for a multi-objective approach to train ANNs has also been discussed in the literature [14]. The multi-objective approach has been shown to present the best generalization performance. The results presented in this paper indicate that the use of an evolutionary multi-objective method for RBF networks design may also result in improved generalization performance. The proposed approach outperformed other well known approaches in a set of experiments performed related to the problem of credit risk assessment using a benchmark dataset.

This paper is organized as follows: Sections 2 and 3 discuss important issues related to credit risk assessment. RBF networks and their usual training approaches are briefly described in Section 4. Section 5 analyzes encoding issues and current techniques for training RBF networks with evolutionary computation. Section 6 presents the proposed method for optimizing RBF Networks based on multi-objective optimization. Finally, the experimental results and conclusions are presented at the end of the paper.

2. The Credit Risk Assessment Problem

There are a large number of political, economic and psychological aspects that affect the behavior of the financial market. These are correlated and interact in a rather complex way. The majority of these relations seem to be probabilistic and nonlinear. Thus, it is hard to express these relations by way of deterministic rules.

Simon, in [15], classifies the financial management decisions in a continuous interval whose limits range from the non-structured to the highly structured decisions. Highly structured decisions are those where the processes necessary for finding an adequate solution are known beforehand and for which several computational tools supporting the decisions are readily available. For non-structured decisions, the only aspects that come into play are the manager's intuition and experience. Specialists may support these managers, but the final decision involves a substantial number of subjective elements. Highly non-structured problems do not adapt easily to conventional computer-based analysis methods or decision support systems [16].

Credit risk analysis is one of the main areas of financial management and has attracted a large deal of attention lately [2–5]. Credit risk analysis is essentially a classification problem that involves evaluation of the reliability and profitability of a credit or loan application. In most cases concerning credit assessment, bank managers must contend with a variety of information from a large number of sources. Much of this information may be incomplete, ambiguous, partially incorrect, or of doubtful relevance. The traditional approach is dependent on the bank manager's experience and follows the procedures and guidelines defined by their institutions. The following section discusses the main approaches employed for credit risk assessment.

3. Main Approaches to Credit Risk Assessment

Credit can be defined as the delivery of a value in exchange for the promise that this value will be paid back in the future. However, there is the risk of this promise not being fulfilled. The employment of formal contracts is an attempt to guarantee the lender the right of receiving the debt. However, these contracts do not ensure that the debt will be paid off, as the debtor may not have the resources necessary to make the required payment.

Credit risk assessment is concerned with the evaluation of the profitability and guarantee of a credit application. Credit applications originate either from companies or consumers. Examples of consumer credit include student loans, personal loans, credit card concessions and home mortgages. Examples of company credit include loans, stocks and bonds. The experiments carried out in this article deal with consumer credit.

In order to reduce the risk of unpaid debts, a technical analysis is generally carried out before the credit is approved. This process is known as credit risk analysis or assessment. Assessment considers several factors that may contribute to non-payment of the debt, such as bankruptcy, dishonesty, economic crisis, etc.

Credit risk assessment in the form of credit cards, direct credit to the consumer and debt cards is usually carried out either empirically or through a credit scoring system. Traditional credit scoring systems are based on discriminate or logistical regression analysis [2, 17]. Thus, the methods that are employed in credit risk evaluation may be roughly divided into two

separate approaches:

– Methods based on previous experience (subjective methods);
– Methods based on numerical estimators (quantitative methods).

The use of previous experience by the credit analyst is the oldest and the most widely used method of credit risk assessment. It is a valuable method, especially when the issues involved are rather subjective. With this approach, a diverse and dynamic range of knowledge is taken into account. Thus, before making his/her decision, the analyst may consider, for example, the customer's employer, the customer's potential in his/her work, the job market related to the customer, the customer's social habits, where the customer lives, the customer's education, signs indicating the customer's honesty, the size of the customer's family, an analysis of the current national economic situation, etc.

However, this previous experience is not easily acquired . Only time and exposure to relevant situations may provide it. Moreover, it is common to have situations where different analysts reach different conclusions for the same application.

Quantitative methods, such as credit scoring, are largely used in order to determine if a credit should be granted. This is clearly a Pattern Recognition problem.

In principle, any Pattern Recognition technique can be used for credit assessment. Previous works have employed techniques like Statistics [2, 18, 19], Knowledge Based Systems [17], Expert Systems [5], Artificial Neural Networks [16, 20], etc. Hybrid Intelligent Systems, where two or more approaches are combined, have also been investigated [4, 21–23].

In this paper, the authors propose the use of RBF networks designed by Genetic Algorithms for credit risk assessment. The following section presents a brief introduction to RBF networks.

4. Radial Basis Function Network

Consider the set of sampled data $\Gamma = \{(\mathbf{x}_i, t_i)\}_{i=1}^{p}$, where p is the number of samples, $\mathbf{x}_i \in \Re^n$ is the input vector and $t_i \in \Re$ is the target output. The function approximation learning problem consists in finding a function $\hat{f}(\mathbf{x})$ that fits the data from Γ. The estimated function $\hat{f}(\mathbf{x})$ is expected not only to minimize the error $(t_i - f(\mathbf{x}_i))^2$, but also to generalize when presented with previously unseen data $(\mathbf{x}_j, t_j)$, not found in Γ. In order to provide $\hat{f}(\mathbf{x})$ with this generalization capacity, several methods have been proposed in the literature, as previously mentioned. This paper proposes the use of RBF networks designed by an evolutionary multi-objective method.

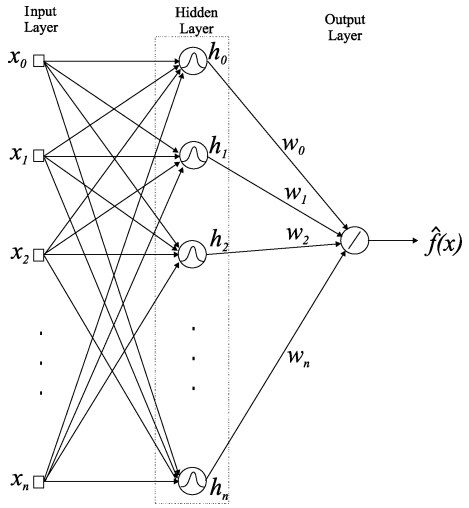

Figure 1. Schematic view of a RBF network.

In the formulation of RBF networks, the function $\hat{f}(\mathbf{x})$ is expressed as a linear combination of functions, as shown in Eq. (1):

$$\hat{f}(x) = \sum_{i=1}^{m} w_i h_i(\mathbf{x}) \tag{1}$$

where $h_i(\mathbf{x})$ is the function associated with the hidden layer node i, also named *basis function*, and the parameter w_i is the weight between the hidden node output $h_i(\mathbf{x})$ and the network output node (see Fig. 1).

Different radial functions can be employed for the hidden nodes. Among them, the Gaussian function, presented in Eq. (2), is the most common.

$$h_i(\mathbf{x}) = \exp \frac{-\|\mathbf{x} - \mathbf{c}_i\|^2}{2\sigma_i^2} \tag{2}$$

where $\|\cdot\|$ is the Euclidean distance norm, $\mathbf{c}_i = (c_{i1}, c_{i2}, \ldots, c_{in})^{\mathsf{T}}$ is the vector *center* and σ_i its radius (also called width).

The training of RBF networks occurs in two steps. The first step includes the selection of appropriate centers and radii for the hidden functions, which is a nonlinear problem. The second step involves the adjustment of the output weights w_i of the output layer, which is a linear problem. Unsupervised learning algorithms

can be applied to the first step, whereas linear algebra solutions can be applied to the second. Simultaneous supervised learning of centers and output weights can also be carried out and involves non-linear optimization methods. In any case, the learning algorithm should be aimed at obtaining a balance between the *bias* and *variance* [24] of the model, which is expected to avoid under-fitting and over-fitting.

Simple learning algorithms make $\mathbf{c}_i = \mathbf{x}_{\alpha_i}$, for $i = 1, \ldots, m$, where $\alpha_i \in \{1, \ldots, p\}$ is randomly chosen. Nevertheless, this approach is prone to generate large networks, overfitting, and numerical problems (mainly when the dataset is noisy) [10]. A more efficient approach employs clustering techniques, such as K-means [25] or Self-Organizing Feature Maps [26]. Another approach is to partition the input space into regions using a decision-tree and fix the centers at strategic positions within these regions [27].

In [28], the authors propose an iterative clustering algorithm for RBF networks that takes into account cluster membership. By using this approach, the authors were able to improve the classification performance of RBF networks.

An algorithm to generate RBF-like networks is presented in [29]. Based on linear programming, this algorithm creates a variety of overlapping Gaussians to act as global and local feature detectors. This network was evaluated on several classification datasets and, for most of them, presented better performance than conventional RBF networks. Like the approach proposed here, this new algorithm significantly reduced the number of hidden nodes.

The widths are usually defined by computationally inexpensive heuristics. Moody and Darken, in [30], suggest that a single value σ for all basis functions offers good results. Moody and Darken used $\sigma = \langle \|\mathbf{c}_i - \mathbf{c}_j\| \rangle$, where $\mathbf{c}_j$ is the nearest center from $\mathbf{c}_i$ and $\langle \cdot \rangle$ indicates the average of all such pairs. Other methods use a different value σ_i for each basis function. In [31], the width σ_i is defined as $\sigma_i = a\|\mathbf{c}_i - \mathbf{c}_j\|$, where a is an overlap factor and $\mathbf{c}_i$ and $\mathbf{c}_j$ are defined in [30].

They also present an interesting solution for the selection of the widths of the Gaussian kernel function.

In the supervised learning step, the training of a RBF network with fixed centers and widths can be interpreted as a linear regression using the training set:

$$\mathbf{t} = \mathbf{H}\mathbf{w} + \mathbf{e} \tag{3}$$

where $\mathbf{t} = [t_1, t_2, \ldots, t_p]^\mathsf{T}$, is the target output vector, $\mathbf{H}$ is the *design matrix*, which is a matrix whose jth column is given by the vector $[h_j(\mathbf{x}_1), h_j(\mathbf{x}_2), \ldots, h_j(\mathbf{x}_p)]^\mathsf{T}$, $\mathbf{w} = [w_1, w_2, \ldots, w_m]^\mathsf{T}$ is the output layer weight vector and $\mathbf{e}$ is the error vector. The vector $\mathbf{w}$ is determined by minimizing the *sum-of-squared-errors* (SSE):

$$\text{SSE} = \mathbf{e}^\mathsf{T}\mathbf{e} \tag{4}$$

with respect to the weights. The solution to this minimization problem can be obtained by solving the well-known linear system:

$$(\mathbf{H}^\mathsf{T}\mathbf{H})\mathbf{w} = \mathbf{H}^\mathsf{T}\mathbf{t} \tag{5}$$

In order to avoid possible numerical problems (ill-conditioning) in solving (5), the use of the Singular Value Decomposition (SVD) has been recommended [32]. SVD computes the pseudo-inverse matrix $\mathbf{H}^+$. Thus, $\mathbf{w} = \mathbf{H}^+\mathbf{t}$, where $\mathbf{H}^+ = (\mathbf{H}^\mathsf{T}\mathbf{H})^{-1}\mathbf{H}^\mathsf{T}$.

Regularization is a technique frequently used to avoid overfitting in ANNs. Penalty or regularization functions are added to the SSE in order to control the smoothness properties of the network. A particular type of regularization is called weight decay (or ridge regression). This procedure minimizes the cost function:

$$C = \text{SSE} + \lambda \mathbf{w}^\mathsf{T}\mathbf{w} \tag{6}$$

where λ is the regularization parameter, which controls the smoothness of the network. The solution to this least-mean-square problem is obtained solving the linear system:

$$(\mathbf{H}^\mathsf{T}\mathbf{H} + \lambda\mathbf{I})\mathbf{w} = \mathbf{H}^\mathsf{T}\mathbf{t} \tag{7}$$

where $\mathbf{I}$ is the identity matrix.

SVD is not necessary to solve the system (7), since regularization itself avoids numerical problems. Thus, faster algorithms, such as Cholesky or LU decomposition [32], capable of solving linear systems can be used instead of SVD. This faster approach is very useful for the design of RBF networks using Genetic Algorithms, since it reduces the overall processing time for the networks evaluation, once computational time in regard to evaluating RBF networks is critical to the overall performance.

5. Evolutionary Optimization of RBFs

After briefly describing Genetic Algorithms, this section discusses the redundancy encoding problem in the

evolutionary optimization of RBF. Previous relevant works on this issue are presented.

5.1. Genetic Algorithms

Genetic Algorithms (GAs) [33, 34] have been successfully employed in search and optimization problems by simulating natural evolution. The traditional GA is based on three biologically-inspired operators, namely:

– Selection;
– Mutation;
– Crossover.

GAs apply these operators to a population of individuals, or chromosomes. When employed in a optimization problem, each chromosome represents a possible solution. A chromosome can be seen as a state (or a point) in the search space. The selection operator directs the population to regions with better chromosomes (solutions). Mutation and crossover operators direct the population to explore unknown regions within the search space. Eventually, the population converges to the best solution.

The selection operator employs a fitness function to evaluate the chromosomes from the population, establishing the fitness for each chromosome according to a user-defined criterion (e.g. the performance of a RBF network for a given validation set).

When evolutionary optimization is applied to RBF networks, each chromosome can be seen as a state in the space of possible RBF networks. The GA starts the process with a population of chromosome encoding networks, usually randomly generated. During the selection phase, networks are selected with probability proportional to the their fitness (the selection operator mimics the natural selection of biological organisms by selecting the fittest individuals from a population). Next, the selected networks are submitted to a reproduction stage, where the crossover and mutation operators are applied, producing new chromosomes. As a result, a new generation is obtained, which is expected to have chromosomes of higher fitness. Several generations may be needed before a suitable solution is found.

5.2. Genetic Encoding

The choice of the appropriate encoding for the chromosomes is a central issue for the optimization of RBF networks through GAs. The encoding defines the class of neural architectures that can be evolved. Moreover, the genetic operators are defined based on the type of encoding chosen. These factors contribute, either directly or indirectly, to the efficiency (with respect to processing time and the fitness values obtained) of the genetic optimization [35].

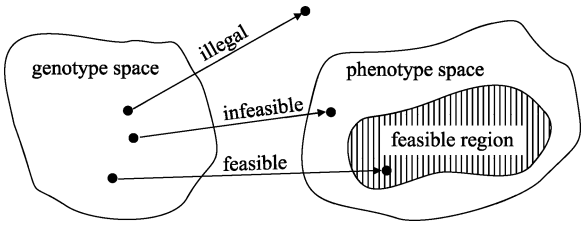

Figure 2. Infeasibility and illegality.

Traditional encodings use binary strings. However, in order to provide a representation that is more suitable to the characteristics of the problem being solved, a large range of encodings have been proposed [36]. Encodings have varied from real strings (used mostly in numerical optimization) and integer permutation encodings (used in some combinatorial optimization problems) to general data structures, often used in engineering problems.

To evaluate a chromosome, GAs map a point from the genotype space to the phenotype space. From this mapping, several important issues related to genetic encodings may rise. Here, it is useful to distinguish two important concepts concerning the genotype-phenotype mapping: infeasibility and illegality (see Fig. 2).

A phenotype is infeasible if it lies outside the feasible region of the optimization problem. A genotype is illegal if it cannot be mapped to the phenotype space. Note that infeasibility originates from the nature of the constrained optimization problems, whereas illegality originates from the nature of problem-specific encodings. Therefore, infeasibility and illegality are unrelated concepts. In order to better explain this, two examples of illegality are given:

Example 1. Consider the Traveling Salesman Problem, TSP: the seller visits N cities (e.g., cities A, B, C, D, and E), returning to the first city. Each city is visited only once. In this example, two possible tours are (BACDE) and (EBDAC). The application of a 2-point crossover [34] to these tours results in:

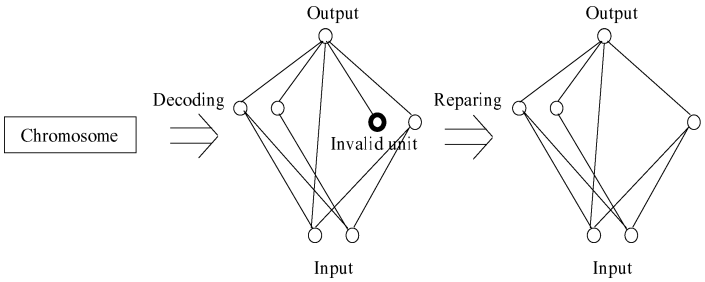

Figure 3. Reparing an illegal network.

Tour 1	(BA|CD|E)
Tour 2	(EB|DA|C)
Offspring	(BA|DA|E) $\Rightarrow$ illegal tour

Clearly, this offspring is illegal because its tour is invalid, since the city A is visited twice. Repair techniques are usually employed to convert an illegal chromosome to a legal one. For example, the well-know PMX crossover [34] often used in the TSP is essentially a 2-point crossover combined with a repair procedure to fix the illegal chromosome produced by the crossover operation.

Example 2. Some encondings used for ANN optimization [37] can produce invalid networks. For example, the ANN from Fig. 3 has one hidden unit with no input connections. Thus, the chromosome that produced this network is illegal. A repair procedure would delete the invalid unit.

Some properties have been proposed in order to evaluate a given encoding [36]:

1. *Nonredundancy*. The mapping between encodings and solutions must be one-to-one. If the n-to-one mapping occurs, the GA wastes search time because one or more individuals may be duplicated in the genotype space. Hence, the one-to-one mapping (nonredundancy) is a desirable property for an encoding. The section to follow shows this property for RBF networks.
2. *Legality*. Any instance of an encoding must correspond to a solution.
3. *Completeness*. Any solution has a corresponding encoding. This property guarantees that any point of the search space is accessible to the GA search.
4. *Casuality*. Small variations in the genotype space due to mutation must imply small variations in the phenotype space.

This focuses on whether the neighborhood of a chromosome (in the genotype space) is also preserved in the corresponding phenotype space [38].

Similar properties have been proposed by [35] in the context of Multilayer Perceptron Neural Networks.

5.3. *Redundancy and Illegality in RBF Network Encodings*

The encoding of RBF networks may suffer from a problem known as redundancy [39]. In the literature on the genetic optimization of ANNs, redundancy is also called by different names: functional equivalence problem [40], competing conventions problem [41] and permutation problem [42]. Redundancy occurs if the mapping from chromosomes (genotype space) to RBF Networks (phenotype space) is not one-to-one (i.e., it is n-to-one).

Two chromosomes are redundant if their associated RBF networks perform the same input-output mapping. An example of redundant encoding is shown next. Consider the chromosome:

$$\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m) \tag{8}$$

where $\mathbf{p}_i$ encodes parameters (e.g., centers and widths) associated with the basis function h_i. By using this encoding, the RBF networks on the left and right sides of Fig. 4 could be encoded by the chromosomes (**a**, **b**, **c**)

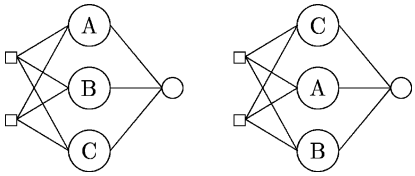

Figure 4. Redundant RBF networks.

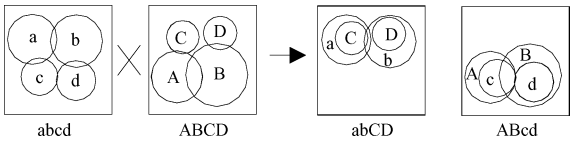

Figure 5. Overlapping Gaussian functions.

and (**c**, **a**, **b**), respectively. Although these networks can perform the same input-output mapping (since they have the same units), they have distinct chromosomes. This significantly increases the search space.

According to [42], the traditional crossover operator is not appropriate for avoiding redundant encoding, since it may generate offsprings with duplicated basis functions (illegality), as Fig. 5 illustrates. Although two identical basis functions in the same chromosome is unlikely, it is possible to have two similar basis functions, as can be seen in Fig. 5 [42]. In this figure, the basis functions *a* and *C* are not identical, but they are very similar because they play similar roles in the network (as they have overlapping Gaussian functions). Such problems make the design of the crossover operator very difficult and may significantly increase the search time.

5.4. *Selecting Centers from Patterns*

Billings and Zheng address the combinatorial aspect of RBF network optimization [11]. In their work, a GA selects a subset of the input patterns to become the center vectors. Each chromosome is a string of variable length and represents a subset of the training patterns. For example, the chromosome (100, 7, 411, 286) represents a RBF network with four centers placed in the patterns labeled 100, 7, 411 and 286. The genetic operators are equal to those employed to solve the so-called *subset selection problem* [43]. To evaluate the performance of their approach, the authors calculated the chromosomes fitness using the Akaike's Information Criterion (AIC) [44] for the training and validation sets with a Multi-objective GA (a similar approach is shown in Sections 6 and 7). This simple representation significantly reduced the number of centers used in traditional approaches.

Maillard and Gueriot [45] modified the model previously described [11] by allowing the centers to assume other points besides the training input vectors. They also investigated the use of several types of basis functions in the same network. According to the authors, networks with different basis functions presented a smaller number of hidden nodes and achieved lower error rates than those using only Gaussian functions.

5.5. *Crossing Hypervolumes*

Carse and Fogarty [12] proposed a method able to genetically optimize centers by crossing hypervolumes of the input space. In this work, each chromosome is represented by a list of tuples $(\mathbf{p}_1, \ldots, \mathbf{p}_m)$. Each tuple is given by:

$$\mathbf{p}_j = (c_{1j}, \sigma_{1j}, c_{2j}, \sigma_{2j}, \ldots, c_{mj}, \sigma_{mj}). \tag{9}$$

The tuple $\mathbf{p}_j$ encodes the parameters of the following basis function:

$$h_j(\mathbf{x}) = \prod_{k=1}^{n} \exp\left(-\frac{(x_k - c_{jk})^2}{\sigma_{jk}^2}\right) \tag{10}$$

which may have a different width for each component of the center vector. This representation is similar to the redundant representation shown in Section 5.3. The authors overcame this problem by using a modified 2-point crossover that exchanges hypervolumes of the input space instead of chunks of the chromosome structure. This hypervolume is determined by two crosspoint vectors $\mathbf{a}, \mathbf{b} \in \Re^n$, whose elements are given by:

$$a_j = \min_j + (\max_j - \min_j) r_1 \tag{11}$$

$$b_j = a_j + (\max_j - \min_j) r_2^{1/n} \tag{12}$$

where r_1 and r_2 are randomly selected from the range $[0, 1]$ with uniform probability density and $[\min_j, \max_j]$ is the allowed range for the component x_j of the input vector $\mathbf{x}$. By presenting experiments in which the performance of the modified 2-point crossover is better than that of the conventional 2-point crossover, the work indicates that the redundancy problem affects the GA performance.

5.6. *Handling Redundancy in RBF Networks*

This section shows another approach to deal with the redundancy problem. Neruda, in [40], formally expresses the functional equivalence (redundancy) between chromosomes as:

Definition 1. Let $\mathbf{p}_i = (w_i, \sigma_i, c_1, c_2, \ldots, c_n)$. Two chromosomes $\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m)$ and $\mathbf{P}' = (\mathbf{p}'_1, \mathbf{p}'_2, \ldots, \mathbf{p}'_m)$ are functionally equivalent if and only if there is a permutation π of the set $(1, \ldots, m)$, such that $\mathbf{p}_i = \mathbf{p}'_{\pi(i)}$, for each $i \in \{1, \ldots, m\}$.

Neruda suggests a unique representation, named *canonical parameterization*, for a class of functionally equivalent chromosomes. He uses a lexicographic ordering for the tuples $\mathbf{p}_i$, which works as follows:

Consider the $(n + 2)$-tuples $\mathbf{p}$ and $\mathbf{q}$. One can say that $\mathbf{p}$ precedes $\mathbf{q}$ ($\mathbf{p} \prec \mathbf{q}$) if there is an index $k \in \{1, \ldots, n + 2\}$ such that $p_j = q_j$, for $j < k$ and $p_k < q_k$. The chromosome $\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m)$ is a canonical parameterization if:

$$\mathbf{p}_1 \prec \mathbf{p}_2 \prec \cdots \prec \mathbf{p}_m \tag{13}$$

The GA proposed here requires chromosomes of canonical parameterization. Its genetic operators were adapted to preserve this property. Mutation is applied to elements of a randomly chosen tuple $\mathbf{p}_i$ generating a new tuple $\mathbf{p}'_i$, which is restricted to the limits:

$$\mathbf{p}_{i-1} \prec \mathbf{p}'_i \prec \mathbf{p}_{i+1} \tag{14}$$

The application of a 1-point crossover on the parents $\mathbf{P} = (\mathbf{p}_1, \ldots, \mathbf{p}_m)$ and $\mathbf{Q} = (\mathbf{q}_1, \ldots, \mathbf{q}_m)$, with a cut point in the position i, produces the offspring:

$$(\mathbf{p}_1, \ldots, \mathbf{p}_i, \mathbf{q}_{i+1}, \ldots, \mathbf{q}_m)$$

which is valid only if $\mathbf{p}_i \prec \mathbf{q}_{i+1}$; otherwise, another cut point must be chosen.

5.7. Other Models

Whitehead and Choate [46] developed a genetic approach that evolves space-filling curves to set the center vectors. The underlying idea involves the mapping of the centers from a n-dimensional region of the input space (defined by such space-filling curves) to a 1-dimensional space in which the chromosomes are encoded. This approach reduces the number of degrees of freedom in the genetic encoding. In another article, the same authors evolved the centers and widths of the radial basis functions through a cooperative-competitive GA [47]. With this method, each individual encodes one single hidden unit. The whole population represents a unique RBF network. The individuals compete and cooperate amongst themselves in order to improve the overall performance of the network represented by the population.

Chen et al. [13], trained RBF networks with a combination of GAs and the ROLS algorithm (Recursive Orthogonal Least Squares). First, a GA evolves the widths and a regularization parameter of the ROLS algorithm. Next, the ROLS algorithm defines the number and positions of the center vectors.

6. The Proposed Approach

This section presents the evolutionary approach proposed by the authors. It describes the encoding, the genetic operators and the objective functions employed.

6.1. Encoding

According to Section 5.2, the encoding is one of the key aspects to be considered when using GAs. In this work, the chromosome is a variable length list of tuples given by:

$$\mathbf{P} = \left(\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_{m_\mathbf{P}}\right) \tag{15}$$

where $m_\mathbf{P}$ is the number of hidden units coded in the chromosome $\mathbf{P}$. Each tuple $\mathbf{p}_j$, $1 \leq j \leq m_\mathbf{P}$, encodes a hidden unit and is given by:

$$\mathbf{p}_j = (r_j, \sigma_j, c_{j1}, c_{j2}, \ldots, c_{jn}) \tag{16}$$

which represents the jth basis function with width σ_j and center $\mathbf{c}_j = [c_{j1}, \ldots, c_{jn}]^\mathsf{T}$. The integer identifier r_j indicates that the center $\mathbf{c}_j$ resides within the region R_{r_j} of the input space (this region is discussed next).

6.2. Partitioning the Input Space

The partition of the input space creates a set of K regions $\{R_1, \ldots, R_K\}$, which are placed in the areas with the largest density of training patterns, as shown in Fig. 6. Each region R_i is obtained from the cluster S_i generated by the K-means algorithm. A region R_i has the shape of a hypercube[1] whose length l_i of its edge is given by:

$$l_i = 2 \max_{\mathbf{x}_j \in S_i} \|\mathbf{m}_i - \mathbf{x}_j\| \tag{17}$$

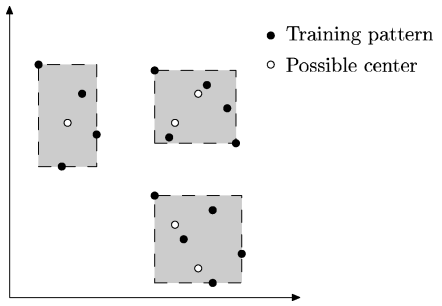

Figure 6. Regions of the input space.

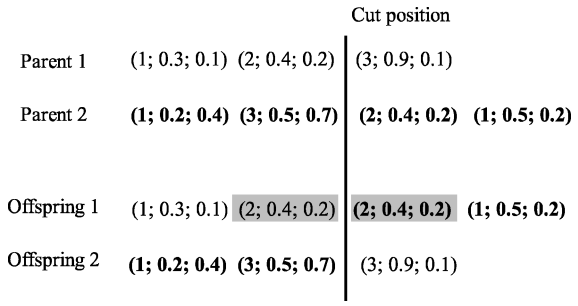

Figure 7. Traditional crossover generates duplicated genes.

where $\mathbf{m}_i$ is the mean of the data points in the set S_i: $\mathbf{m}_i = |S_i|^{-1} \sum_{\mathbf{x} \in S_i} \mathbf{x}$. Thus, the region R_i embraces all points of the cluster S_i, as shown Fig. 6.

6.3. *Decoding*

Consider the following chromosome to be decoded: $\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m)$ where $\mathbf{p}_j = (r_j, \sigma_j, c_{j1}, c_{j2}, \ldots, c_{jn})$. The parameter $r_j \in [1, K]$ is encoded as a integer. The parameters $\sigma_j, c_{j1}, c_{j2}, \ldots, c_{jn}$ from $\mathbf{p}_j$ are encoded as floating-point values and normalized in the interval $\in [0, 1]$. The following decoding is employed:

$$c'_{jk} = m_{r_j,k} + l_i(c_{jk} - 0.5), \quad \text{for } k = 1, \ldots, n. \quad (18)$$

$$\sigma'_j = \alpha \sigma_j \quad (19)$$

where $m_{r_j,k}$ is the kth component of the cluster mean $\mathbf{m}_{r_j}$ from the region R_{r_j} and the coefficient α is a scaling factor (in this work α is equal to the half of the maximum distance separating pairs of training input patterns).

6.4. *Operators*

The use of the traditional crossover operator may produce duplicated genes in a single chromosome. For example, consider a case in which the centers are in the 1-dimensional space. In this simple case, the chromosomes are formed by a variable list of 3-tuples (region, width, center). Figure 7 shows the traditional crossover producing an offspring with duplicated genes. In order to avoid this illegality problem, a cluster crossover operator, illustrated by Fig. 8, is proposed.

The main steps of the cluster crossover operator is described in Fig. 9. It works like the traditional uniform crossover, but crosses regions R_i instead of structural chunks of chromosomes (usually performed by the standard uniform crossover). The cluster crossover is intended to perform the crossing in the phenotype space, whereas the standard uniform crossover operator is performed in the genotype space. It is similar in spirit to Carse and Fogarty's crossover (see Section 5.5).

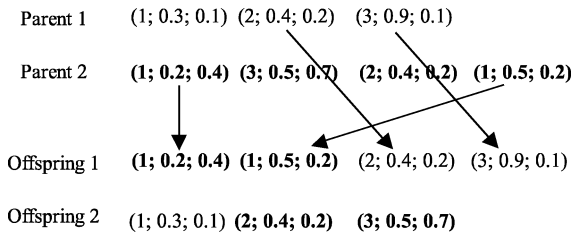

Figure 8. The cluster crossover operator.

Create a template bit string randomly $(b_1, b_2, \ldots, b_K)$, $b_i \in \{0, 1\}$

for $i = 1$ **to** K **do**
  if $b_i = 1$ **then**
    Move all genes $\mathbf{g}_j = (i, \sigma_j, c_{1j} \ldots c_{nj})$
    from the 1st parent to the 1st child
  else
    Move all genes $\mathbf{g}_j = (i, \sigma_j, c_{1j} \ldots c_{nj})$
    from the 2nd parent to the 1st child
  endif
  Move all remainder genes to the 2nd child.
endfor

Figure 9. The cluster crossover algorithm.

The following mutation operators are used: (1) The uniform mutation replaces widths, centers and region identifiers with a uniform random number. (2) The creep mutation adds Gaussian noise to the value of widths and centers. The added noise is small, so creep mutation plays the role of a local search. (3) The addition and delete operators add and delete randomly selected hidden units.

6.5. Choosing the Objective Function

After decoding a chromosome, the weights associated to the nodes in the output layer are determined by a pseudo-inverse matrix. Once the weights are defined, the chromosome is evaluated. Since the main goal is to optimize the topology, such evaluation should not only consider the network performance, but also establish a balance between performance and complexity.

The value of the objective function could be defined by using cross-validation. However, this method is computationally intensive. For RBF networks with fixed centers and widths, there are other computationally more efficient criteria. A good option for evaluating the objective function is the Generalized Crossvalidation (GCV) [48]. When used to evaluate the performance of RBF networks, the GCV can be defined as:

$$f = \text{GCV} = \frac{p\,\text{SSE}}{(p-m)^2} \tag{20}$$

where SSE denotes the sum of squared errors on the training set, m is the number of free parameters (i.e. number of weights) and p is the number of the training patterns. Equation (20) has some modifications if the RBF network is trained with regularization (see [49] for details).

In previous studies, the objective function employed has been based on training set error rates. As this may result in overfitting, the experiments reported in this article employ generalized cross-validation (GCV) for both the training and validation sets. As a result, two objective functions are optimized:

$$f_1 = \frac{p_1\,\text{SSE}_1}{(p_1-m)^2} \tag{21}$$

$$f_2 = \frac{p_2\,\text{SSE}_2}{(p_2-m)^2} \tag{22}$$

where p_1 and p_2 are the number of patterns from the training and validation sets, respectively, and m is the number of free parameters. SSE_1 and SSE_2 are the sum of squared errors for the training and validation sets, respectively.

The method employed by the authors to deal with the multi-objective optimization problem using GAs is described in the next section.

7. The Multi-Objective Genetic Algorithm

Let $f_1, f_2, \ldots, f_q$ be the set of objective functions to be minimized. Instead of a single objective function, each chromosome is now evaluated by a multi-objective function. The fitness of a chromosome is defined by a vector where each component is the value of an objective function. In order to compare chromosomes based on these vectors, the following definitions are employed [50]:

Definition 2. Let $\mathbf{a}$ and $\mathbf{b}$ be vectors of objective function values. A vector $\mathbf{b}$ is said to be dominated by (or inferior to) a vector $\mathbf{a}$ if $\mathbf{a}$ is partially-less-than $\mathbf{b}$ ($\mathbf{a} <_p \mathbf{b}$), where:

$$\mathbf{a} <_p \mathbf{b} \iff \forall i(a_i \leq b_i) \wedge \exists i(a_i < b_i) \tag{23}$$

Definition 3. A vector $\mathbf{a}$ is said to be non-dominated (or non-inferior) if there is no other vector (in the population) that dominates $\mathbf{a}$.

The set of all non-dominated vectors of the population is called the *Pareto-optimal set* (also known as the Pareto frontier). In such cases, the goal of the multi-objective optimization is to find the Pareto-optimal set.

7.1. Ranking

An appropriate selection pressure is necessary for successful evolution of fit individuals [34]. Ranking methods have been applied to control the level of selection pressure [77]. Using the concept of non-domination, it is possible to rank the population. Pareto ranking methods to rank the population are proposed in [34,50]. According to [50], an individual $\mathbf{P}$ that is dominated by n individuals in the current population has its rank given by:

$$\text{rank}(\mathbf{P}) = 1 + n. \tag{24}$$

All nondominated individuals are assigned rank 1. Figure 10 shows an example where rank 4 is absent. A multi-objective optimization approach for training ANNs was recently described in [14].

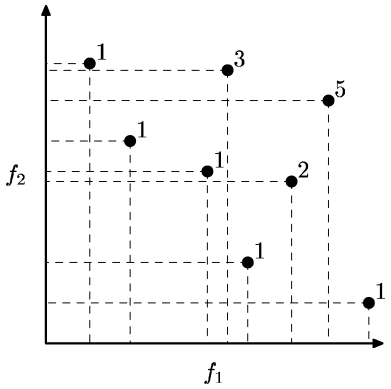

Figure 10. Pareto ranking method.

Table 1. GA parameters.

Parameter	Choice
Population	100
Generations	200
Crossover rate	0.8
Mutation rate	0.01
Creep rate	0.05
Creep Std Deviation	0.01
Addition rate	0.05
Subtraction rate	0.05
Act. function	Gaussian
Maximum number of hidden units	20

7.2. *Fitness Assignment*

Let $\mathbf{P}_1, \ldots, \mathbf{P}_N$ be the individuals from a population sorted in ascending order of rank, where N is the population size. The fitness of each individual is given by:

$$\text{fitness}(\mathbf{P}_k) = s - \frac{2(k-1)(s-1)}{N-1} \tag{25}$$

where $s \in \{1, 2\}$ is a user-defined parameter called selection pressure (best/median fitness ratio). Experiments have shown that s equal to either 1.1 or 1.2 provides good results. In order to ensure that individuals with the same rank have the same fitness, they are averaged by:

$$\text{fitness}(\mathbf{P}_k) = \langle \text{fitness}(\mathbf{P}_i) \rangle_k \tag{26}$$

where $\langle \cdot \rangle_k$ denotes the average of all individuals with rank equal to rank($\mathbf{P}_k$). Parents are sampled for crossover and mutation using the Stochastic Universal Sampling (SUS) technique [51].

8. Experiments

This section investigates the performance achieved by the proposed approach in a credit risk assessment dataset. For such, the performance of evolutionary designed RBF networks is compared to that achieved by RBF networks (trained in other techniques), other ANN models and Support Vector Machines (SVMs).

The credit risk assessment dataset used in the experiments was obtained from the Proben1 repository [52]. This dataset is a conversion of a credit assessment dataset from the UCI Machine Learning Repository [53]. The Proben1 repository has a set of datasets and guidelines to perform experiments with ANNs. The experiments carried out in this article followed the Proben1 recommendations.

This credit assessment dataset represents credit card applications and consists of 690 examples with 51 inputs and 2 outputs each. The input values have continuous and discrete attributes, along with missing attributes in 5% of the samples. The meaning of the individual attributes is unexplained for reasons of confidentiality. The two output values determine approval (44.5% of the samples) or non-approval (55.5% of the samples) of credit. The dataset was randomly separated three times into three subsets for training (50% of the examples), validation (25% of examples) and test (25% of examples). The experiments were conducted three times for each evaluated model, shuffling the data selected for the subsets.

In the experiments, the GAs were run with the parameters illustrated in Table 1 (the maximum number of hidden units parameter is used to reduce processing). Experiments were also carried out with RBF networks trained with different clustering algorithms. The clustering algorithms used were the batch K-means [54], the on-line K-means [25], the iterative optimization, IO, algorithm [55], a clustering algorithm based on depth-first search, DF [56], two clustering algorithms combining IO and DF, called DFIO and IODF [57] and the optimal adaptive K-means algorithm [58]. Table 2 shows the average and standard deviation of the percentage of wrongly classified patterns, PWCP, obtained for the test set.

The results obtained suggest that the evolutionary approach proposed by the authors is more accurate

Table 2. Credit assessment by RBF networks.

Clustering	PWCP (%)
Batch	16.67 ± 3.87
DF	16.28 ± 2.54
IO	17.83 ± 3.96
DFIO	16.67 ± 4.28
IODF	17.25 ± 4.44
On-line	16.86 ± 4.39
Optimal	15.89 ± 4.66
GA	13.95 ± 3.53

Table 3. Credit assessment by other models.

Model	PWCP (%)
MLP-Backprop	17.05 ± 1.77
Cascade correlation	18.02 ± 3.03
Tower	14.73 ± 3.20
Pyramid	16.86 ± 2.10
SVM	16.67 ± 2.63

than the other clustering techniques for the dataset employed. Furthermore, it is worth mentioning that the proposed approach generated the smallest number of basis functions, 15, yielding more parsimonious networks.

In order to provide a better picture of the potential of the proposed approach, the authors investigated the performance achieved with other models for the same dataset. The authors ran experiments with the same dataset using Multi Layer Perceptron (MLP) networks trained by the backpropagation algorithm [59], three constructive learning algorithms Cascade Correlation [60], Tower and Pyramid [61], all using early stopping, and SVMs [62]. In all simulations using SVMs, RBF kernels with a variance equal to 1 and an upper limit for the Lagrange Multipliers equal to 500 were used. Table 3 presents the PWCP for the test set obtained by these models.

The authors applied the t-test [63] to analyze the statistical significance of the results obtained. The results achieved by the Genetic approach are better than those obtained by the other approaches with 95% of confidence.

These results obtained demonstrate the potential of the proposal approach. The genetically designed RBF networks presented the lowest average of wrong classifications and had among the lowest standard deviations. Besides, in comparison with the other approaches investigated for the training of RBF networks, the proposed approach selected a smaller number of hidden nodes.

One of the positive aspects of the proposed algorithm is the change of focus from RBF to GA parameters tuning. It was observed in the experiments that it was easier to set the GA parameters than the RBF network parameters. There are two reasons for this, which are discussed next.

Conventional methods for RBF optimization have many user-defined parameters that are not known a priori and whose values are often problem specific. Thus, for a particular problem, a wide variety of values must be tried to find a global optimal (or approximated optimal) solution. This trial-and-error search for finding optimal parameter settings is common in ANN literature and is one of the limitations to its widespread use. As opposed to conventional methods, the GAs parameters often are predefined based on past research [64]. For example, the crossover rate (0.8) and the mutation rate (0.01) were taken from the traditional GA algorithm [34].

A second reason why it is easier to set the GAs parameters is the robustness of the search performed by GAs. The basic mechanism of a GA is so robust that, within fairly wide margins, the GA parameters are not critical [65]. For example, a small change in the parameter settings of the conventional methods can completely change the RBF network functionality, whereas changes on GA parameters usually do not cause large variations in the GA search results.

Next, results obtained by other authors for the same dataset are presented. Although the authors used different partitions, making comparison difficult, it is interesting to see the different performances obtained.

In [17], the authors report that average wrong classification error rates of 21.6% and 19.6% were obtained using a discriminate analysis-based method [66] and the ID3 algorithm [67], respectively.

In [68], two versions of the C4.5 algorithm [69] and three different versions for the traditional and extended Foil algorithm [70] achieved the results presented in Table 4.

The authors also carried out experiments using other datasets. In particular, the authors evaluated their method using the chaotic time series Mackey-Glass [71], a heart disease dataset from the UCI Machine

Table 4. Credit assessment by C4.5 and Foil.

Algorithm	PWCP (%)
C4.5 rules	15.5
C4.5 trees	15.1
Foil trad.[1]	17.8
Foil trad.[2]	17.4
Foil trad.[3]	17.0
Foil exd.[1]	18.0
Foil exd.[2]	16.4
Foil exd.[3]	16.4

Learning repository [53] and data that model the Hermite polynomial. The performances achieved by the RBF networks trained with GAs for these datasets were also shown to be superior to those achieved by other methods for the training of RBF networks.

9. Conclusion

With a growing number of customers, the credit industry needs more sophisticated methods for assessing credit risks. ANNs have provided efficient solutions to this problem. However, their performance depends on the adequate setting of the network parameters.

In this article, RBF networks designed by GAs were investigated for credit risk assessment. The performance obtained through this approach was compared to those obtained by other clustering techniques for RBF training and additional machine learning techniques, such as Support Vector Machines and other ANN models. The results obtained suggest the superiority of the proposed genetic approach. Among the RBF training methods, the genetically designed RBF networks presented the lowest average classification error and the smallest average number of hidden nodes. Moreover, while the other techniques had their parameters defined by an extensive trial and error process, the parameters of the proposed approach were designed automatically by GAs.

It must be pointed out that the use of GAs leads to a longer processing time when compared with the other techniques. However, the time required to design ANNs can be divided into conception time and processing time. By automatically defining the main network parameters, the evolutionary design largely reduces conception time.

Further improvement can be achieved by looking for alternatives for reducing the optimization time (e.g. replacing SVD by another method, such as Cholesky or LU decomposition [32]) and including other parameters in the chromosomes (such as different radial basis functions). Another option is the employment of alternative objective functions.

Acknowledgment

The authors would like to thank CNPq, FAPESP, FACEPE, FAPEMIG, and FINEP for their support. The authors would also like to thank Humberto de Souza, Murilo Brizotti and Marcelo Barros for helping with the experiments and David Calvert for his valuable comments.

Note

1. Alternatively, the shape might be an ellipsoid since it avoids the hypercube corners overlap. For simplicity, the hypercube shape was adopted in this work.

References

1. H. White, "Economics prediction using neural networks: The case of IBM daily stock return," in *Proceedings of IEEE International Conference on Neural Networks*, 1996, pp. II-451–II-459.
2. L. Richeson, R. Zimmerman, and K. Barnett, "Predicting consumer credit performance: Can neural networks outperform traditional statistical methods?" *International Journal of Applied Expert Systems*, vol. 2, no. 2, pp. 116–130, 1994.
3. R. Trippi and E. Turban, *Neural Networks in Finance and Investing*. Irwin Professional Publishing, 1996.
4. E. Mendes Filho, A. de Carvalho, and A. Matias, "Credit assessment using evolutionary MLP networks," in *Decision Technologies for Computational Finance, Proceedings of the fifth International Conference on Computational Finance, CF'97*, Advances in Computational Management Science, Kluwer Academic Publishers, 1997, pp. 365–371.
5. E. Nikbakht and M. Taft, "Application of expert systems in evaluation of credit card borrowers," *Managerial Finance*, vol. 15, no. 5, pp. 19–27, 1997.
6. A. Atiya, "Bankruptcy prediction for credit risk using neural networks: A survey and new results," *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 929–935, 2001.
7. D.S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Systems*, vol. 2, pp. 321–355, 1988.
8. S. Chen, C.F.N. Cowan, and P.M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 2, no. 2, pp. 302–309, 1991.

9. J. Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 3, no. 2, pp. 213–225, 1991.
10. M.J.L. Orr, "Regularisation in the selection of radial basis function centres," *Neural Computation*, vol. 7, no. 3, pp. 606–623, 1995.
11. S.A. Billings and G.L. Zheng, "Radial basis function network configuration using genetic algorithms," *Neural Networks*, vol. 8, no. 6, pp. 877–890, 1995.
12. B. Carse and T.C. Fogarty, "Fast evolutionary learning of minimal radial basis function neural networks using a genetic algorithm," in *AISB Workshop on Evolutionary Computing*, edited by T.C. Forgaty, *Lectures Notes in Computer Science*, no. 1143, Springer-Verlag, 1996, pp. 1–22.
13. S. Chen, Y. Wu, and K. Alkadhimi, "A two-Layer learning method for radial basis function networks using combined genetic and regularised OLS algorithms," in *Proceedings of the 1st IEE/IEEE International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, 1995, pp. 245–249.
14. R.A. Teixeira, A.P. Braga, R.H.C. Takahashi, and R.R. Saldanha, "Improving generalization of MLPs with multiobjective optimization," *Neurocomputing*, vol. 35, pp. 189–194, 2000.
15. H. Simon, *The New Science of Management Decision*, Harper and Row: New York, 1960.
16. D. Hawley, J. Johnson, and D. Raina, "Artificial neural systems: A new tool for financial decision making," in *Neural Networks in Finance and Investing*, edited by R. Trippi and E. Turban, revised edition, New York, Irwin, 1996, pp. 25–44.
17. C. Carter and J. Catlett. "Assessing credit card applications using machine learning," in *IEEE Expert*, 1987, pp. 71–79.
18. D. Hand and W. Henley, "Statistical classification methods in consumer credit scoring: A review," in *Journal of the Royal Statistical Society*, Series A, vol. 160, pp. 523–541, 1997.
19. D. Hand and W. Henley, "Some developments in statistical credit scoring," in *Machine Learning and Statistics: The Interface*, edited by G. Nakhaeizadeh and C. Taylor, John Wiley & Sons, 1997, pp. 221–237.
20. H. Sousa and A. de Carvalho, "Credit analysis using constructive neural networks," in *Proceedings of the 3rd International Conference on Computational Intelligence and Multimedia Applications, ICCIMA'99*, IEEE Computer Press: New Deli, 1999, pp. 40–44.
21. E. Reategui, J.A. Campbell, and S. Borghetti, "Using a neural network to learn general knowledge in a case-based system," in *Proceedings of First International Conference on Case-Based Reasoning*, edited by A. Aamodt and M. Veloso, 1995, pp. 528–537.
22. E. Lacerda, and A. de Carvalho, "Credit analysis using radial basis function networks," in *Proceedings of the 3rd International Conference on Computational Intelligence and Multimedia Applications, ICCIMA'99*, IEEE Computer Press: New Delhi, 1999, pp. 138–142.
23. P. Horst, T. Padilha, C. Rocha, S. Rezende, and A. de Carvalho, "Knowledge acquisition using symbolic and connectionist algorithms for credit evaluation," in *Proceedings of the IEEE World Congress on Computational Intelligence, WCCI'98*, CD media, Anchorage, USA, 1998.
24. S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias-variance dilemma," *Neural Computation*, vol. 4, pp. 1–58, 1992.
25. J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkley Symposium Math. Stat. Prob.*, vol. 1, pp. 281–297, 1967.
26. T. Kohonen, "Self-organized formation of topologically correct feacture maps," *Biological Cybernetics*, vol. 43, pp. 59–69, 1982.
27. M. Kubat, "Decision trees can initialize radial-basis function networks," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 813–821, 1998.
28. M.T. Musavi, W. Ahmed, K.B. Faris, and D.M. Hummels, "On the training of radial basis function (RBF) Classifiers," *Neural Networks*, vol. 5, no. 4, pp. 595–603, Elsevier, 1992.
29. A. Roy, S. Govil, and R. Miranda, "An algorithm to generate radial basis function (rbf)-like nets for classification problems," *Neural Networks*, vol. 8, no. 2, pp. 179–201, Elsevier, 1995.
30. J. Moody and C.J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, no. 2, pp. 281–294, 1989.
31. A. Saha and J.D. Keller, "Algorithms for better representation and faster learning in radial basis function networks," in *Advances in Neural Information Processing Systems*, edited by D.S. Touretzki, vol. 2, 1990, pp. 482-489.
32. W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes in C*. Cambridge University Press, 1988.
33. J.H. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press: Ann Arbor, 1975.
34. D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
35. K. Balakrishnan and V. Honavar, "Properties of genetic representations of neural architectures," in *Proceedings of the World Congress on Neural Networks (WCNN'95)*, Washington, D.C., July 17–21, 1995, pp. 807–813.
36. M. Gen and R. Cheng, *Genetic Algorithms and Engineering Optimization*, Wiley, 2000.
37. S.A. Harp, T. Samad, and A. Guha, "Towards the genetic synthesis of Neural Networks," in *Proceedings of the 4th International Conference on Genetic Algorithms*, Morgan Kaufmann, 1991, pp. 360–369.
38. B. Sendhoff, M. Kreutz, and W. von Seelen, "A condition for the genotype-phenotype mapping: Causality," in *Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA'97)*, edited by T. Bäck, Morgan Kauffman: San Francisco, 1997, pp. 354–361.
39. J.D. Schaffer, D. Whitley, and L.J. Eschelman, "Combinations of genetic algorithms and neural networks: A survey of the state of the art," in *Proceedings of the International Workshop on Combinations of Genetic Algorithms and Neural Networks (COGANN-92)*, IEEE, 1992, pp. 1–37.
40. R. Neruda, "Functional equivalence and genetic learning of RBF networks," *Artificial Neural Nets and Genetic Algorithms*, edited by D.W. Pearson, N.C. Steele and R.F. Albrecht, Springer-Verlag, pp. 53–56, 1995.
41. J.D. Schaffer, D. Whitley, and L.J. Eshelman, "Combinations of genetic algorithms and neural networks: A survey of the state of the art," in *Proc. of the Conf. on Combinations of Genetic Algorithms and Neural Networks*, 1992, pp. 1–37.
42. P.J.B. Hancock, "Genetic algorithms and permutation problems: A comparison of recombination operators for neural net structure specification," in *Proceedings of the IEEE Workshop*

on Combinations of Genetic Algorithms and Neural Networks, 1992, pp. 108–122.
43. C.B. Lucasius and G. Kateman, "Towards solving subset selection problems with the aid of the genetic algorithm, in *Parallel Problem Solving from Nature*, edited by R. Manner and B. Manderick, vol. 2, Elsevier Science Publishers: Amsterdam, 1992.
44. H. Akaike, "A new look at the statistical model identification," *IEEE Transactions on Automatic Control*, vol. 19, pp. 716–723, 1974.
45. E.P. Maillard and D. Gueriot, "RBF neural network, basis functions and genetic algorithm," in *Proceedings of International Conference on Neural Networks*, vol. 4, pp. 2187–2192, 1997.
46. B.A. Whitehead and T.D. Choate, "Evolving space-fiiling curves to distribute radial basis functions over an input space," *IEEE Transactions on Neural Networks*, vol. 5, pp. 15–23, 1994.
47. B.A. Whitehead and T.D. Choate, "Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction," *IEEE Transactions on Neural Networks*, vol. 7, pp. 869–880, 1996.
48. G.H. Golub, M. Heath, and G. Wahba, "Generalised cross-validation as a method for choosing a good ridge parameter," *Technometrics*, vol. 21, no. 2, pp. 215–223, 1979.
49. M.J.L. Orr, "Introduction to radial basis function networks," TR Centre for Cognitive Science, Univ. of Edinburgh, Scotland, 1996.
50. C.M. Fonseca and P.J. Fleming, "Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization," in *Proceedings of the 5th International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, Inc.: San Mateo, 1993, pp. 416–423.
51. J. Baker, "Reducing bias and inefficiency in the selection algorithm," in *Proc. of the Second International Conference on Genetic Algorithms and their Applications*, edited by J. Grefenstette, Lawrence Erlbaum Associates: Hillsdale, New Jersey, 1987, pp. 14–21.
52. L. Prechelt, "Proben1—A set of neural networks benchmark problems and benchmarking rules," Fakultät füt Informatik, Universität Karlsruhe, Technical Report 21/94, 1994.
53. C.A. Murphy and D.W. Aha, "UCI repository of machine learning databases," Irvine, CA, University of California, 1994.
54. S.P. Lloyd, "Least square quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137. 1982.
55. R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, Wiley-Interscience Publication, 1973.
56. M.A. Ismail, S.Z. Selim, and S.K. Arora, "Efficient clustering of multidimensional data," in *Proceedings of the IEEE International Conference on Systems Man and Cybernetics*, 1984, pp. 120–123.
57. M.A. Ismail and M.S. Kamel, "Multidimensional data clustering utilizing hybrid search strategies," *Pattern Recognition*, vol. 22, pp. 75–89, 1989.
58. C. Chinrungrueng and C.H. Séquin, "Optimal adaptive K-means algorithm with dynamic adjustment of learning rate," *IEEE Transactions on Neural Networks*, vol. 6, pp. 157–169, 1995.
59. D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representation by error propagation," in *Parallel Distributed Processing*, MIT Press: Cambridge, 1986, pp. 318–362.
60. S.E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," Technical Report. School of Computer Science. Carnegie Mellon University, 1991.
61. R. Parekh, J. Yang, and V. Honavar, "Constructive neural network learning algorithms for multi-category real-valued pattern classification," Technical Report. Department of Computer Science. Iowa State University, 1997.
62. B. Scholkopf, *Support Vector Learning*. R. Oldenburg Verlag, 1997.
63. R. Mason, R. Gunst, and J. Hess, *Statistical Design and Analysis of Experiments*, John Wiley & Sons, 1989.
64. S.R. Sexton and J.N.D. Gupta, "Comparative evaluation of genetic algorithm and backpropagation for training neural networks," *Information Sciences*, vol. 129, pp. 45–59, 2000.
65. J.J. Grefenstette, "Optimization of control parameters for genetic algorithms 1986," *IEEE Trans SMC*, vol. 16, pp. 122–128.
66. B.S. Everitt and G. Dunn, *Applied Multivariate Data Analysis*, John Wiley & Sons, 1991.
67. J.R. Quinlan, "Discovering rules from large collection of examples: A case study," in *Expert Systems in the Microelectronic Age*, edited by, D. Michie, 1979.
68. R.M. Cameron-Jones and J.R. Quinlan, "First order learning, zeroth order data," in *Proceedings of the AI'93 Australian Joint Conference on Artificial Intelligence*, World Scientific: Melbourne, 1993.
69. J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.
70. J.R. Quinlan, "Learning logical definitions from relations," *Machine Learning*, vol. 5, pp. 239–266, 1990.
71. M.C. Mackey and L. Glass, "Oscillations and chaos in physiological control systems, *Science*, pp. 197–287, 1977.
72. D.L. Bailey and D.M. Thompson, "Developing neural network applications," *AI Expert*, vol. 5, no. 9, pp. 34–41, 1990.
73. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2002.
74. E. Mendes Filho and A. de Carvalho, "Evolutionary design of MLP neural network architectures," in *Proceedings of the IV Brazilian Symposium on Neural Networks, IV SBRN*, IEEE Computer Press, 1997, pp. 58–65.
75. L. Prechelt, "Automatic early stopping using cross-validation: Quantifying the criteria," *N*eural Networks, 1998.
76. B. Widrow and M.E. Hoff, "Adaptive switching circuits," in *IRE-WESCON Convention Record*, vol. 4, pp. 96–104, 1960, New York, edited by J.A. Anderson and E. Rosenfeld. *Neurocomputing: Foundations of Research*. MIT Press: Cambridge, MA, 1988.
77. D. Whitley, "The GENITOR algorithm and selective pressure," in *Proc. of the Third Int. Conf. on Genetic Algorithms and their Applications*, edited by J. Schaffer, Morgan Kaufmann: San Mateo, CA, 1989, pp. 116–121.
78. A. Zell et al., "SNNS: Stuttgart neural network simulator user's manual version 4.1. no. 6/95," 1995. (http://www.informatik.uni stuttgart.de/ipvr/bv/projekte/snns/).