A clustering genetic algorithm for extracting rules from multilayer perceptrons trained in classification problems

E. R. Hruschka[1] & N.F.F. Ebecken[2]
[1] *Universidade Tuiuti do Paraná, Brazil.*
[2] *COPPE /Universidade Federal do Rio de Janeiro, Brazil.*

Abstract

This paper deals with the task of using supervised neural networks in data mining applications. The proposed methodology makes use of a clustering genetic algorithm, which is applied in the hidden units activation space in order to extract rules from multilayer perceptrons trained in classification problems. We illustrate the proposed method by means of two examples: Iris Plants Database and Meteorological dataset.

1 Introduction

Multilayer perceptrons (MP) adjust their internal parameters performing vector mappings from the input to the output space. Although they may achieve high accuracy of classification, the knowledge acquired by such neural networks is usually incomprehensible for humans [1]. This fact can be a major obstacle in domains such as data mining where it is important to have symbolic rules or other forms of knowledge structure [2]. Therefore, many methods have been developed to get explicit knowledge from these models. One observes that, fundamentally, the knowledge acquired by a neural network is codified on the connection weights that, in turn, determine the activation function values. Thus, the knowledge acquisition process from supervised neural networks implies the use of algorithms based either on the connection weight values or on the hidden unit activation values. The algorithms designed to perform this task are usually called *algorithms for rule extraction from neural networks.*

This paper describes the application of a Clustering Genetic Algorithm (CGA) for rule extraction from MP. The rule extraction algorithm basically consists of two steps. First, we apply the CGA to find clusters of activation values in the first hidden layer. Second, we generate a set of rules that describes the hidden unit discretized values in relation to the inputs. In this way, the *CGA* provides a set of rules in the following form:

$$\text{If } \{ v^1_{min} \le a_1 \le v^1_{max} \text{ and } \dots \text{ and } v^n_{min} \le a_n \le v^n_{max} \} \text{ then class } C_j; \quad (1)$$

where a_is are the hidden unit activation expressions, v^i's are the maximum or minimum values of the obtained clusters for each class C_j and n is equal to the number of hidden units.

Several authors notice the need for simplification of neural networks to facilitate the rule extraction process and are in favor of using specialized training schemes and architectures. We argue that it is also important to develop algorithms of *general application*, since their utilization is twofold, because they would allow the use of supervised neural networks in new data mining classification problems as well as they would provide an approach for clarifying the knowledge encoded in previously defined neural network models. In principle, however, our method can only provide satisfactory results when the neural network uses linear transformation functions in the input units, because in this way the hidden unit activation expressions are also linear functions of the attributes, providing more comprehensible rules.

Our paper is organized as follows: the next section describes the CGA whereas the third section presents the results obtained in the Iris Plants database [3], providing a pedagogical illustration of the proposed method. In the fourth section we describe the results achieved in a meteorological dataset and in the fifth section we conclude our work.

2 Clustering genetic algorithm

Clustering is a task where one seeks to identify a finite set of categories or clusters to describe the data. Basically, there are three kinds of clustering techniques: overlapping, hierarchical and partitioning. This work deals with the partitioning approach, which assigns each object to exactly one cluster. Thus, this work considers that clustering involves the partitioning of a set X of objects into a collection of mutually disjoint subsets C_i of X. Formally, let us consider a set of N objects $X = \{X_1, X_2, \dots, X_N\}$ to be clustered, where each $X_i \in \Re^\rho$ is an attribute vector consisting of "ρ" real measurements. The objects must be clustered into non-overlapping groups $C = \{C_1, C_2, \dots, C_k\}$ where k is the number of clusters, such that:

$$C_1 \cup C_2 \cup \dots \cup C_k = X, \quad C_i \neq \varnothing, \text{ and } C_i \cap C_j = \varnothing \text{ for } i \neq j. \quad (2)$$

The problem with finding an optimal solution to the partition of N data into C classes is NP-complete [4] and because the number of distinct partitions of n objects into m clusters increases approximately as $m^n/m!$, attempting to find a globally optimum solution is usually not computationally feasible [5]. Genetic algorithms are widely believed to be effective on NP-complete global optimization problems and they can provide good sub-optimal solutions in reasonable time [6]. Thus, a clustering genetic algorithm can provide a way of finding the right clustering. Some approaches that describe the application of genetic algorithms to clustering problems can be found in [4,6,7].

There are several complications concerning the application of traditional genetic algorithms for clustering problems. The use of a simple encoding scheme, that yields to constant-length chromosomes, usually causes the problems of redundant codification and context insensitivity [4]. However, the CGA uses such schemes and avoids all of their problems.

2.1 Encoding scheme

Considering that there are N objects to be clustered, a phenotype is represented as a one dimensional integer array with (N+1) elements. As each data unit can be numbered from 1 to N, the *ith* element of a genotype represents the *ith* data unit, whereas the last gene represents the number of clusters. Therefore, each gene of a chromosome has a value over the alphabet {1,2,3,...,k}, where *k* is the maximum number of clusters. For example, considering a dataset with 20 objects one can get the following clustering: 22345123453321454552 5.

This means that the cluster whose label is *2* is formed by 5 objects{1,2,7,13,20}, and the cluster whose label is *1* has 2 objects {6,14}. The last gene corresponds to the number of clusters encoded by the solution. The first problem with this straightforward encoding scheme is that it is redundant (there are 5! chromosomes encoding the same solution to the original problem). Thus, the search space is much larger than the original one. In addition, there is another problem caused by this encoding scheme when it is applied with the traditional genetic operators. It is the undesirable effect of casting context-dependent information *out of context* under the standard crossover, i.e. the context insensitivity [4]. This problem happens when one is dealing with the classic genetic operators of crossover and mutation, which usually produce invalid solutions. To avoid these problems, we have developed genetic operators that are group-oriented. Basically, the developed CGA is as described in Figure 1.

```
1) Initialize a population of genotypes;
2) Evaluate each genotype in the population;
3) Apply a linear normalization;
4) Select genotypes by proportional selection;
5) Apply crossover and mutation;
6) Replace the old genotypes by the ones formed by 5);
7) If the convergence is attained, stop; if not, go to step 2).
```

Figure 1. Clustering Genetic Algorithm (CGA).

2.2 Objective function

Determining the optimal number of clusters in a dataset is one of the most difficult aspects in the clustering process [7]. Most of the approaches found in the literature determine the right number of clusters according to criteria based on given clusterings. However, the CGA is suppose to optimize not only the clusterings for a given number of clusters but also the number of clusters. Therefore, the CGA employs an objective function based on the Average Silhouette Width [8]. Basically, let us consider an object “i” belonging to cluster A. So the average dissimilarity of “i” to all other objects of A is denoted by a(i). Now consider a different cluster C and let us calculate the average dissimilarity of “i” to all objects of C, which will be here denoted by d(i,C). After computing the d(i,C) for all clusters C≠A we select the smallest of those, b(i)=min d(i,C) for C≠A .This number represents the dissimilarity of “i” to its neighbor cluster. Now one defines the silhouette s(i) like follows:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \tag{3}$$

When the cluster A contains only one object we consider that s(i)=0, which is the most neutral choice [8]. In addition, it is easy to see that: $-1 \leq s(i) \leq 1$. The objective function is the average of s(i) for i=1,2,...,N. It implies that the best value of “k” happens when the objective function value is as high as possible.

2.3 Selection

The genotypes that make part of each generation are selected according to the roulette wheel selection strategy. As this strategy does not allow negative objective function values, a constant number equals to one is summed up to each objective function value before the selection process takes place. Besides, the best genotype is always copied into the succeeding generation.

2.4 Operators

Classic genetic operators are not suitable for clustering problems [4] because they usually work with a set of unrelated genes. In clustering problems the genes are related, i.e. equal allele genes mean that the objects they represent belong to the same cluster. In other words, the groups are the meaningful building blocks of a solution. Thus, it is necessary to use operators that work with groups of genes, instead of with the genes themselves. This fact lead us to develop operators referred to as group oriented [4].

The crossover operator combines together pieces of information coming from different genotypes and it works in the following way: First, two genotypes (A and B) are selected; Second, considering that A represents $k1$ clusters, the CGA chooses randomly n [1,k1] clusters to copy in B. The unchanged groups of B are maintained and the changed ones have their objects allocated to the cluster that has the nearest centroid. In this way the child C is obtained. This same process is employed to get child D, but now considering that the changed clusters of B are copied in A.

There are two operators for mutation. The operator 1, which only works in clusterings formed by more than two groups, eliminates a randomly chosen group and places all their objects to the remaining cluster that has the nearest centroid. The operator 2 divides a randomly selected group into two new ones. The first group is formed by the objects closer to the centroid, whereas the other group is formed by those objects nearer to the farthest object to the centroid. These operators are suitable for grouping problems because they just change the genotypes in the smallest possible way, i.e. dividing groups and eliminating others in the hope of finding better solutions to the problem being solved. Besides, 50% of the genotypes are crossed-over, 25% are mutated by operator 1 and 25% are mutated by operator 2.

2.5 Initial population

Fundamentally, we have also employed the methodology developed in [8] to set up the initial population. The initial clustering process is based on the selection of representative objects. The first selected object is the most centrally located in the set of objects. Subsequently, other objects are selected. Basically, the chance of selecting an object increases when it is far from the previously selected ones and when there are many objects next to it. After selecting the representative objects, the initial population is formed considering that the nonselected objects must be clustered according to their proximity to the representative ones. Considering k representative objects, the first genotype represents two clusters, the second genotype represents three clusters,..., and the last one represents k clusters. Thus, we have employed initial populations formed by (k-1) genotypes, where each genotype represents a different clustering.

3 Pedagogical illustration – Iris Plants Database

This database consists of three classes with 50 examples of each class. The class Setosa is linearly separable from the others, whereas the classes Versicolour and Virginica are not linearly separable. There are four attributes (sepal and petal length and width). However, we did not use the original attributes to get classification rules, because using the sepal and petal areas as inputs to the MP provides better results than using the original attributes [9]. The sepal area is obtained by multiplying the sepal length by the sepal width and the petal area is calculated in an analogous way.

We want to compare the results obtained by the CGA in the original space of attributes with those obtained in the hidden units activation space. Thus, we applied the CGA to the space formed by the petal and sepal areas, which are also the inputs to the neural network model. In this way, we can compare models based on the same variables and one can glimpse situations in which it is useful to use MP to extract classification rules.

Our simulations consider the maximum number of generations equals to 2,000. Besides, we just run the CGA one time, but we believe that the obtained results are likely the best ones, because the CGA always converged before the last generation. In fact, we think that running the CGA several times would not provide a different final solution, but indeed the CGA could converge after a different number of generations. In addition, we did not apply any crossvalidation process, because they cannot be so useful to evaluate rule extraction methods [1].

3.1 Applying the CGA to the sepal and petal areas

We applied the CGA using both the Euclidean and the Manhattan distance as dissimilarity measures. Our best result was achieved using the Manhattan distance. The CGA employed a population formed by 72 genotypes. The best solution (Figure 3) was obtained after 68 generations (fitness function = 0.49) and it classifies 88% of the objects correctly. One can see the right clustering in Figure 2 (sepal area in the horizontal axis and petal area in the vertical axis).

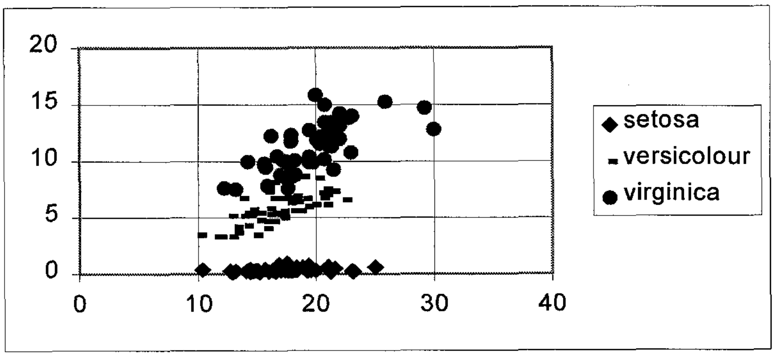

Figure 2. Right Clustering - areas.

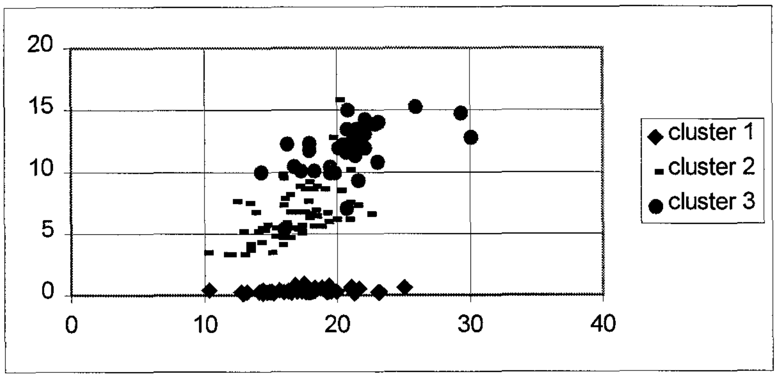

Figure 3. Clustering using the Manhattan Distance.

3.2 Applying the CGA to the hidden units activation space

This section describes the application of a MP to get explicit rules that describe the involved classes. First, several MP were trained to get satisfactory results in terms of the average classification rate. The complete dataset was used for training the neural networks. The best neural network model is formed by two hidden units considering the areas of sepals and petals as inputs.

The linear [-1,+1] transformation function was used in the input units, the hyperbolic tangent function in the hidden units and the logistic function in the output units. After 15,000 epochs (learning rate = 0.6, momentum term = 0.9), the average quadratic error rate was equal to 0.0003. Then, the hidden units activation values for each example are computed according to:

$$a_1 = 0.22A_s - 2.16A_p + 0.84 \tag{4}$$
$$a_2 = 3.12A_s - 11.91A_p + 41.99 \tag{5}$$

The CGA was then applied to these values. The best fitness function value (0.72) was found in the initial generation. Figure 4 shows the corresponding clusters, which are described by the following rules:

If ($2{,}27 \le a_1 \le 5{,}43$ and $69{,}64 \le a_2 \le 113{,}09$) then c_1 (50 objects) -setosa;
If ($-13{,}09 \le a_1 \le -3{,}46$ and $3{,}47 \le a_2 \le 46{,}54$) then c_2 (49 objects) - versicolour;
If ($-29{,}04 \le a_1 \le -12{,}46$ and $-84{,}56 \le a_2 \le -1{,}14$) then c_3 (51 objects) - virginica;

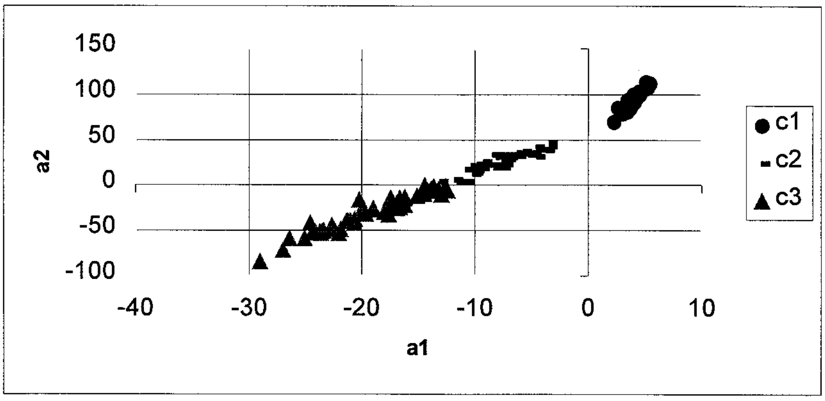

Figure 4. Clusters of activation values.

These rules classify 96% of the examples correctly. In this way, the methodology of applying the CGA in the hidden unit activations provided better results. It happens because it is simpler to extract patterns in a space transformed by the hidden units, which are feature detectors. Putting it in another way, the hidden units try to map the classes, which in the original space can be not linearly separable, into the hidden unit activation space, where the classes are likely linearly separable [10]. From the geometrical point of view, these crisp logical rules correspond to a division of the feature space with hyperplanes perpendicular to the axes, into areas with symbolic names. If the classes in the input space are correctly separated with such hyperplanes, accurate logical description of the data is possible and worthwhile [1]. Besides, the methodology here described was already evaluated in other two benchmarks and the results are described in [11].

4 Meteorological database

This dataset was collected at the International Airport of Rio de Janeiro and it is formed by 17,443 examples. Each example is described by means of 38 attributes and the associated class, which describes weather conditions. The examples that contain missing values were eliminated. The remaining 1,226 were divided in three subsets: training, validation (employed during training) and test - according to the following table:

Class	Examples	Training	Validation	Test
Normal	302	180	61	61
Cloudy	34	20	7	7
Dry fog	125	75	25	25
Wet fog	512	306	103	103
Drizzle	67	39	14	14
Rain	152	90	31	31
Lightning	34	20	7	7
Total	1226	730	248	248

Table 1. Number of examples for each class.

Several neural networks were trained in order to get satisfactory results in terms of the average error rate. The best obtained model is formed by six input units ([-1,+1] linear transformation function), four hidden units (logistic function), and seven output units (logistic function). The selected attributes were: x_1(visibility), x_2(dew temperature), x_3(pressure), x_4(wet bulb temperature), x_5(precipitation) and x_6(air relative humidity). This model provided an average quadratic error equals to 0,2143 (validation set) after 638 training epochs (learning rate=momentum term=0.1). The hidden unit activation expressions are:

$a_1 = -22{,}02+0{,}0125x_1+0{,}0671x_2+0{,}0026x_3-0{,}0067x_4-0{,}00535x_5-0{,}03x_6.$
$a_2 = -28{,}33-0{,}0014x_1+0{,}4571x_2-0{,}0029x_3-0{,}059x_4+0{,}02827x_5+0{,}44636x_6.$
$a_3 = 17{,}08-0{,}0081x_1+0{,}0771x_2+0{,}00034x_3+0{,}0215x_4-0{,}0026x_5-0{,}15636x_6.$
$a_4 = 10{,}02-0{,}0015x_1+0{,}1228x_2+0{,}00809x_3+0{,}0077x_4-0{,}0297x_5-0{,}15848x_6.$

Considering that wet fog conditions are very important to airport operation, we applied the CGA in the activation values formed by the examples of this class in order to get classification rules that describe the conditions in which the wet fog occurs. The best rule set, found by using the Manhattan distance, is described by two clusters (formed by 104 and 202 objects). The fitness function was equal to 0.46 after 53 generations. Thus, the resulting classification rules, which classify correctly 72.18% and 71.37% of the validation and test set examples respectively, are:

If{$-22{,}8 \le a_1 \le -11{,}5$ and $0{,}3 \le a_2 \le 12{,}7$ and $0{,}3 \le a_3 \le 9{,}4$ and $-3{,}3 \le a_4 \le 2{,}5$} then wet fog;
If{$-13{,}0 \le a_1 \le 0{,}7$ and $-1{,}9 \le a_2 \le 17{,}1$ and $-8{,}2 \le a_3 \le 1{,}8$ and $-10{,}0 \le a_4 \le 1{,}3$} then wet fog;
Else another weather condition.

We also applied the CGA in the original space of attributes. First, it was applied to get classification rules describing the wet fog conditions considering all the attributes and, in this way, the average classification rates were equal to 60.48% and 59.68% in the validation and test sets respectively. Second, we applied the CGA in the dataset formed by the six selected attributes (used to get the best neural network model). Thus, the CGA found eight clusters, formed by {45,52,85,53,5,24,37,5} objects (fitness function=0.57) after 86 generations. The best rule set was found using the Manhattan distance and classifies correctly 72.18% and 73.79% of the validation and test set examples respectively. The corresponding rules, where WF stands for *wet fog*, were:

If$\{400 \leq x_1 \leq 700 \wedge 15 \leq x_2 \leq 24 \wedge 89 \leq x_3 \leq 261 \wedge 170 \leq x_4 \leq 268 \wedge 0 \leq x_5 \leq 121 \wedge 80 \leq x_6 \leq 97\}$ then WF;
If$\{900 \leq x_1 \leq 1000 \wedge 15 \leq x_2 \leq 25 \wedge 46 \leq x_3 \leq 269 \wedge 155 \leq x_4 \leq 268 \wedge 0 \leq x_5 \leq 188 \wedge 80 \leq x_6 \leq 100\}$ then *WF*;
If$\{1500 \leq x_1 \leq 1800 \wedge 16 \leq x_2 \leq 24 \wedge 48 \leq x_3 \leq 266 \wedge 179 \leq x_4 \leq 270 \wedge 0 \leq x_5 \leq 60 \wedge 76 \leq x_6 \leq 96\}$ then *WF*;
If $\{x_1=1200 \wedge 14 \leq x_2 \leq 24 \wedge 83 \leq x_3 \leq 268 \wedge 151 \leq x_4 \leq 261 \wedge 0 \leq x_5 \leq 280 \wedge 80 \leq x_6 \leq 100\}$ then *WF*;
If $\{1800 \leq x_1 \leq 2000 \wedge 19 \leq x_2 \leq 22 \wedge 95 \leq x_3 \leq 177 \wedge 202 \leq x_4 \leq 272 \wedge 0 \leq x_5 \leq 4 \wedge 82 \leq x_6 \leq 96\}$ then *WF*;
If $\{800 \leq x_1 \leq 900 \wedge 18 \leq x_2 \leq 23 \wedge 92 \leq x_3 \leq 239 \wedge 182 \leq x_4 \leq 262 \wedge 0 \leq x_5 \leq 76 \wedge 80 \leq x_6 \leq 100\}$ then *WF*;
If $\{x_1=1400 \wedge 16 \leq x_2 \leq 25 \wedge 25 \leq x_3 \leq 244 \wedge 180 \leq x_4 \leq 280 \wedge 0 \leq x_5 \leq 255 \wedge 80 \leq x_6 \leq 96\}$ then *WF*;
If $\{160 \leq x_1 \leq 300 \wedge 19 \leq x_2 \leq 23 \wedge 79 \leq x_3 \leq 256 \wedge 210 \leq x_4 \leq 264 \wedge 0 \leq x_5 \leq 48 \wedge 81 \leq x_6 \leq 96\}$ then *WF*;
Else another weather condition.

These rules provide similar classification rates to the ones based on the activation values. However, it should be noticed that the neural network model allowed to perform a feature selection task, providing better classification rules. Considering the complexity involved in meteorological phenomena, we believe that the results are good, but they must be carefully evaluated by domain experts.

5 Conclusions and future work

This paper describes a way of using multilayer perceptrons in data mining applications. The proposed methodology makes use of a Clustering Genetic Algorithm (CGA), which is applied in the hidden units activation values in order to extract rules from multilayer perceptrons trained in classification problems. This algorithm allows both to find the *best* number of clusters and also to get the best clustering according to the average silhouette width criterion. We illustrated the method by means of two examples: Iris Plants Database and Meteorological database. The CGA was employed both in the original space of attributes and in the hidden units activation space and the resulting classification rules were compared in terms of the average classification rates. The results show that the proposed method is very promising. Besides, comprehensibility is often regarded as a very important issue in data mining applications. Therefore, one important future development is to evaluate the complexity of the obtained rule sets.

Acknowledgments

Financial support by the "Fundação Araucária de Apoio ao Desenvolvimento Científico e Tecnológico do Estado do Paraná" is gratefully acknowledged.

References

[1] DUCH, W., ADAMCZAK, R., GRABCZEWSKI, K. *A New Methodology of Extraction, Optimization and Application of Crisp and Fuzzy Logical Rules*, IEEE Transactions on Neural Networks, vol.11, n.2, pp.1–31, March 2000.

[2] KEEDWELL, E., NARAYANAN, A., SAVIC, D. *Creating Rules from Trained Neural Networks Using Genetic Algorithms*, International Journal of Computers, Systems and Signals (IJCSS), vol.1, n.1, pp. 30-42, December 2000.

[3] MERZ, C.J., MURPHY, P.M., UCI Repository of Machine Learning Databases, [http://www.ics.uci.edu]. Irvine, CA, University of California, Department of Information and Computer Science.

[4] FALKENAUER, E., Genetic Algorithms and Grouping Problems, John Wiley & Sons, 1998.

[5] ARABIE, P., HUBERT, L. J., An Overview of Combinatorial Data Analysis (Chapter 1). *Clustering and Classification*, ed. P. Arabie, L.J. Hubert, G. DeSoete, World Scientific, 1999.

[6] PARK, Y., SONG, M., "*A Genetic Algorithm for Clustering Problems*", Proceedings of the Genetic Programming Conference, University of Wisconsin, July, 1998.

[7] COLE, R. M., *Clustering with Genetic Algorithms*, Master of Science Thesis, Department of Computer Science, University of Western Australia, 1998.

[8] KAUFMAN, L., ROUSSEEUW, P. J., *Finding Groups in Data* – An Introduction to Cluster Analysis, Wiley Series in Probability and Mathematical Statistics, 1990.

[9] HRUSCHKA, E.R. and EBECKEN, N.F.F. " Rule Extraction from Neural Networks: Modified RX Algorithm ", *Proceedings of the IEEE International Joint Conference on Neural Networks* (IJCNN'99), Washington DC, USA, July, 1999.

[10] HAYKIN, S. S, Neural Networks: A Comprehensive Foundation, Second Edition, Prentice Hall, 1998.

[11] EBECKEN, N.F.F., HRUSCHKA, E.R., "Rules from Supervised Neural Networks in Data Mining Tasks", Frontiers in Artificial Intelligence and Applications, v.71, In: Logic, Artificial Intelligence and Robotics, J. M. Abe and J. I. Silva Filho Eds, pp. 84 -100, IOS Press, 2001.