Separation surfaces through genetic programming

José C. Riquelme, Raúl Giráldez, Jesús S. Aguilar and Roberto Ruiz

Departamento de Lenguajes y Sistemas Informáticos.
Av. Reina Mercedes s/n 41012 Sevilla Spain
e-mail:{riquelme,giraldez,aguilar,rruiz}@lsi.us.es

Abstract. The aim of this paper is to describe a study for the obtaining, symbolically, of the separation surfaces between clusters of a labelled database. A separation surface is an equation with the form $\phi\ (x)=0$, where ϕ is a function of $\Re^n \rightarrow \Re$. The calculation of function ϕ is begun by the development of the parametric regression by means of the use of the Genetic Programming. The symbolic regression consists in approximating an unknown function's equation, through knowledge of certain points' coordinates and the value that a function reaches with the same ones. This possibility was propose in [Koza92a] and its advantage in front of the classic statistical regressions is that it is not necessary previously to know the form the function. Once this surface is found, a classifier for the database could be obtained. The technique has been applied to different examples and the results have been very satisfactory.

Key words: genetic programming, classification, dynamical systems.

1. Introduction: Problem statement

The Evolutionary Algorithms (EA), according to [Goldberg89], are adaptive procedures for the search of solutions in complex spaces inspired by the biological evolution, through operation patterns based on the Darwin's theory about reproduction and survival of the individuals that better adapt to the surroundings where they live. In 1989, J. R. Koza, in [Koza89], introduced the paradigm of the genetic programming (GP), which was developed in his texts [Koza92b] and [Koza94] later.

The search space for the GP is the hyperspace of the valid trees, which can be recurrently created from the composition of possible functions and terminals. The initial information is made up by a database where each register is composed of two fields: a n-tuple of real values, that represent the coodinates of a point in a domain Ω_p, which we denominated *parameters* or *characteristics*, and of a *class* or *label* associate to that point, which takes values in a discreet finite space. Therefore, we have N points P_j (j=1..N) in $\Re^n$, each one with an assigned label. This is a typical problem of classification in supervised learning. Multiple approaches to the problem exist in bibliography: neuronal networks, decision trees, nearest neighbours, etc.

This simplification to limit itself two regions, which does not constitute a limitation to a more general classification problem, because if there exist more than two different regions, is enough with repeating the scheme for all the possible pairs. It is necessary to consider that to speak about a surface that separates three regions does not have sense, and to assume a connected region of the space is conformed by points of some type. If the points of a same type did not conform an only connected region, we can calculate the connected regions ([Riquelme97] exposes an approach to this problem) and apply the algorithm these.

The evolutionary process will try to find a function f^* of $\Re^n$ in $\Re$, like optimal function to approximate to a function ϕ which theoretically defines the surface. Basically, the method consists of choosing the points of Ω_p for a training file, assigning to each point a value of function ϕ that we want to approximate. Thus each point of Ω_p becomes one tuple of a training file for the parametric regression. The training file is organised like N records with the coordinates of each P_j and the value of $\phi(P_j)$.

An evolutionary process like the GP applied to this file of training obtains that the parametric regression provides a function, which will tend to separate two subspaces constituted by two sets of points with different characteristics. This is thus because the function will have to take positive values for the points from a set and negatives for those of the other. So, the function ϕ $(x_0, x_1 ..., x_n)=0$ will be a separation surface between the points of both types.

The advantages of the GP as opposed to other techniques of the classification would be the following ones: With regard to the trees of the decision, we not only can work with any type of separation surface, also with hyperplanes, which are used habitually by these techniques. With regard to the neuronal networks, the GP does not have to find the architecture of the network, nor a parametric equation for the surface, task that is not simple in a neuronal network. The main disadvantage is the time of computation, mainly with regard to using decision trees. Nevertheless, the classification problems usually do not need a learning in real time, although once this learning are done, its application yes must be immediate, and our technique fulfils this characteristic.

2. Proposed solution

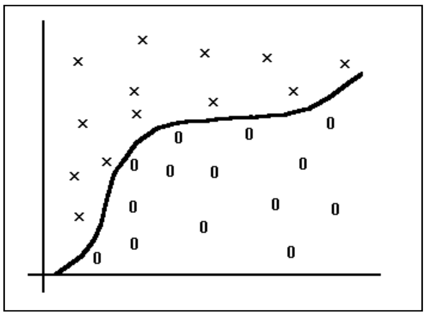
2.1 Representation of the individuals of the population

The tree structure will be, with arithmetic operators F={ +, -, *, \ } and operand T = { x_0, $x_1 ..., x_n$ } which are coordinate of the space to classify, the chosen one for the representation of the individuals of the population. The operands will only appear in the terminal nodes, whereas the internal nodes will be labelled with the operators.

To generate each tree in the initial population is made through the random obtaining of a tree with the labelled nodes. It will be begun at random selecting one of the functions, which will be the root's label of the tree. We will restrict the election of this label to the set F because we want to generate a structure hierarchic, not a degenerated structure only composed of an only terminal.

Whenever a point of the tree is labelled with an operator of F we will generate two subtrees which will represent the parameters of this operator. Then, an element is selected randomly to label those new subtrees. If a function is chosen to be the label, then the generation process continues recurrently as we described previously.

However, if a terminal is chosen to be the label in any point, that point becomes a leaf of the tree and the generation process finishes for that subtree.



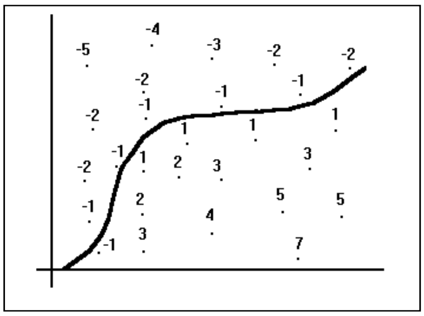


Fig. 1. and **Fig. 2.** Example: Assignment of values to the points of the training file.

2.2 Fitness Function

The notation in this section is as follows: the function ϕ is the searched ideal function, that is to say, the one that defines the separation surface. This function will be approximated by a set of individuals f_i during the evolutionary process. When this process has finished, an approach f^* will be obtained. And finally, the function φ is defined like the function that measures the error of approach of f_i to ϕ. In other words, φ is the function that must be minimised.

The function ϕ, in each point P of the training file, is defined as follows: If P is a point of the Region A, the nearest points to P (V_j j=1..Nv) are calculated, but only if these belong to B. Then the average value of the points V_j and P is assigned to the function ϕ in P. Otherwise, if P is a point of Region B, the operation previous is repeated with the class A neighbours, but the function ϕ take the value of the average distance but with negative sign. This allocation of values for the training file tries that the space of points is classified clearly in two subspaces of values (positive and negative) and where the transition of the points of a type to those of the other is made gradually (Fig. 1 and 2).

The adjusted function φ has to be optimise. This function is inversely proportional to the average of the errors of the individuals of a generation. This error can be calculated through two methods:

1. The error of an individual f_i in a point P_j in the training file can define itself as the absolute value of the difference between the functions f_i and ϕ for that point. If we added the error of each point P_j, we obtain a possible function of error that will try to approach the searched function of separation ϕ by means of f_i:

$$Error_1(f_i) = \sum_{j=1}^{N} \left| f_i(P_j) - \phi(P_j) \right|$$

2. Another possibility is to calculate the error of a function f_i based on the number of

incorrectly classified points. If the aim is that the functions f_i come near to a function ϕ, which separates the points of class A of the B, an individual f_i commits an error in P_j if the value of its sign is not equal than the previously assigned sign. That is to say, if the P_j is of class A, then the values taken by ϕ are positives, and therefore, f_i is punished if it is negative in P_j. And if the P_j is of class B, the opposite happens. Formally:

$$Error_2(f_i) = \sum_{j=1}^{N} g(P_j) \quad where \quad g(P_j) = \begin{cases} 0 & if\ P_j \in regionA \wedge f_i(P_j) > 0 \\ 0 & is\ P_j \in regionB \wedge f_i(P_j) < 0 \\ 1 & otherwise \end{cases}$$

After making diverse tests, we choose a linear combination of both possibilities: on the one hand, the number of incorrectly classified points which gave rise to the integer part of the error; on the other, the fractional part that had been formed by the sum of the differences between the approach *fi* and ϕ for each point P. This means that, definitively, the error of a function *fi* is

$$Error(f_i) = Error_2(f_i) + 10^{-m} * Error_1(f_i)$$

where m>1 is selected so that the fractional part is smaller than 1.

2.3 Reproduction model

We must solve how to produce the next generation of trees (functions) from the initial population, once well-known the convenience of the same ones. To make this task, we must choose a reproduction elitist model, is to say, the best individual of the present generation is duplicated and introduced in the next one. Besides a constant percentage of the new generation is obtained by means of copy of the selected individuals randomly in the previous generation with a probability based on its fitness. Finally, the remaining individuals of the new generation are obtained through crossovers between the individuals of the precedent generation.

The sizes of population are 100 and 200. We worked with initial heights from 4 to 5 and with maximum heights between 7 and 9. The passage by the generations 100, 200 and 400 will be analysed. With the percentage of the best individuals that pass directly from a generation to another one, we can regulate that the generations are more or less elitist. The values with which we work are 10%, 15%, and 20%.

3. Application

3.1 Iris File

A typical database example for classification problems is IRIS DATA since it was proposed in [Fisher 36]. This is a file of 150 registers with four parameters and three labels. The method has been applied for the obtaining of an approach to the surface that separates the different labels. The first approach consists of separating the first type of the other two. This is a easy exercise, because the classes are clearly separated. The simplest solution is p2-p3=0, that is to say, if p2>p3 then type 1 and if p3>p2 then type 2 or 3. This solution does not commit any error.

More difficult is to separate types 2 and 3. Nevertheless, the reached solution to separate these two types has 4 errors (Fig 3. left) or only two errors with a more complex solution (Fig. 3 right)

$$(p_1 p_4)(p_4 - p_1 + p_3) - p_4 = 0 \qquad (p_2 + p_3 p_4 + p_3 - 2 p_1)(p_2 - p_3) + p_1 = 0$$

Fig. 3. Surfaces for IRIS data

3.2 Basins of attraction of a dynamical system

A more complex example of application is to obtain the surfaces that separate the basins of attraction of a dynamic system. The studied model is a predator-prey ecological system. The points to classify based on the obtained attractors are 500 with the following division: 317 points converge to a equilibrium points, 154 points are limit cycles and 15 points are indetermined attractors. The aim of this example is to try to find surface that separates the attraction basins, that is to say, the 317 equilibrium points from the 154 limit cycles. The figure 4 shows a projection of the types of attractors on coordinates *y* (axis of abscissa) and z (axis of ordinate) of the initial conditions, where the X represent balance points, the circles represent limit cycles and the crosses represent the undetermined points.

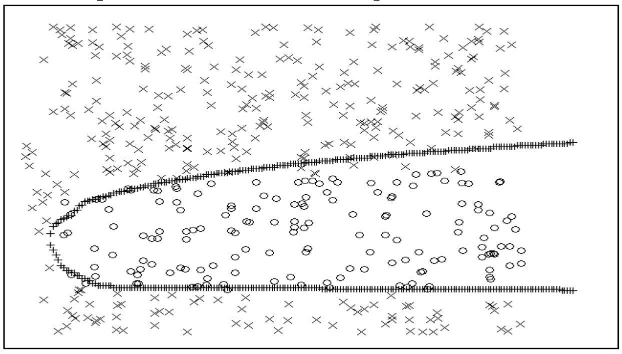


Fig. 4. Separation surface.

If we notice the figure 4, the limit cycles are concentrated in a strip determined by coordinate *z* and for almost all the values of the coordinate *y* (except for values very next to zero). This strip seems a parabola with the axis parallel to the axis of abscissas and with the almost parallel branches in the represented region. If it is tried to find an analytical approach to the surface, the found equation with smaller number of errors (3%) is as follows:

$$(3z^3 + 4y^2 - y - 3z)(4y^2 + 10yz) + y + 2z^6 - yz^2 = 0$$

A graphical approach of this function can be obtained. In the figure 4, these border points are shown by means of crosses.

4. Conclusions

An application of the genetic programming is shown to find the symbolic expression of a surface that separates two regions of points of different type. The symbolic parametric regression can be applied after assigning to a numerical value to each tuple of points in the space that we want to classify. This value has been chosen like the average distance of a point to the nearest points of the other type (for some it is positive and for others it is negative). Therefor, if we demand the continuity of the function, its value equal to zero provides us the searched surface and, in addition, a symbolic classifier for any database. The main disadvantage that we found is the time of computation, mainly if we related it to the decision trees. For example, the execution with the IRIS file can need about 15 minutes computation in a present PC. But the obtained error rates are very inferior and the classification is immediate once the surface is found.

6. Acknowledgements

This work has been supported by the Spanish Research Agency CICYT under grant TIC99-0351.

5. References

1. Fisher, R.: The use of multiple measurement in Taxonomic problems. Annals of Eugenics, nº 7, pp. 179-188, 1936.

2. Goldberg, D.E.: Genetic algorithms in search, optimization and machine learning. Addison-Wesley, 1989.

3. Koza, J.R.: Hierarchical genetic algorithms operating on populations of computer programs. Proc. of 11th Int. J. Conf. on A.I., I, pp 768-774, Morgan Kauffman,. 1989.

4. Koza, J.R.: The genetic programming paradigm: genetically breeding populations of computer programs to solve problems. Dynamic, Genetic and Chaotic Programming, Ed. Baranko Soucek e IRIS Group, pp 269-276. John Wiley & Sons. 1992.

5. Koza, J.R.: The genetic programming: on the programming of computers by means of natural selection. The MIT Press. 1992.

6. Koza, J.R.: Genetic Programming II. Automatic discovery of reusable programs. The MIT Press. 1994.

7. Riquelme, J. y M. Toro: Search and linguistic description of connected regions in quantitative data. Proc. of the IFAC Conference on the Control of Industrial Systems CIS-97, Vol. 3 pp. 310-316, Belfort (France), 1997.