Hybrid trees for supervised learning: towards extraction of decision rules

Florence d'Alché-Buc, Jean-Pierre Nadal [*], Didier Zwierski
Laboratoires d'Electronique Philips
B.P. 15, 22 avenue Descartes, 94453 Limeil-Brévannes Cedex France
e-mail: dalche@lep.lep-philips.fr

Abstract

We propose a new constructive approach, hybrid trees, that combines decision trees and neural learning methods to learn and build automatically neural systems for classification and regression tasks. Neural trees can be described as decision trees where each node is a formal neuron. In hybrid trees, internal nodes segment the input space into regions where the terminal nodes perform the classification or regression task . Different kinds of functions can be used as terminal nodes such formal neurons, radial basis functions, small multilayered networks. As in simple decision trees, once the tree is built, one can extract decision rules associated with each leaf of the tree. The interpretability of the rule depends on the choice of the terminal nodes. We describe the hybrid trees in the case of a binary classification task. As an example, application to handwritten character discrimination is presented.

1 Introduction

Different approaches have been recently developped to combine symbolic and numeric processes. In this paper, we focus on rule extraction within a numerical framework. One approach consists in directly using neural architecture well appropriate for implementing numerical inference such as fuzzy logic [Lin and Lee, 1991], [d'Alché-Buc et al., 1992], probabilistic inference [Higgins et al., 1993]. Another approach consists in constraining the weights of a multilayered network in such way that the network can be interpretable in terms of crisp rules [Denoeux, 1992], [Kane et al., 1992] . Whatever the approach, one faces the same problems than in learning neural networks e.g. : the choice of the architecture, the problem of fixing the complexity (number of units, number of layers), and the learning and localizing local units. In this paper we propose another approach based on decision trees that combines a constructive method and the rule extraction goal : hybrid trees. Hybrid trees generalize decision trees [Breiman et al., 1984], [Quinlan, 1986] and especially neural trees that consists of a constructive algorithm for building system formally equivalent to a three-layered perceptron [Frean, 1990], [Marchand and Golea, 1990], [Sirat and Nadal, 1990]. Neural trees are extended in such way that it is possible to learn different kinds of nodes (not only perceptrons). Thus the work we present here can be considered with two respects : one deals with automatic construction of multilayered networks based on different kinds of functions, and the other deals with rule extraction especially in case of classification where some discriminant functions such as radial basis functions, can describe decision regions in a way more interpretable for the user.

[*]Permanent address: Laboratoire de Physique Statistique, Ecole Normale Supérieure, 24 rue Lhomond 75231 Paris Cedex 05 France

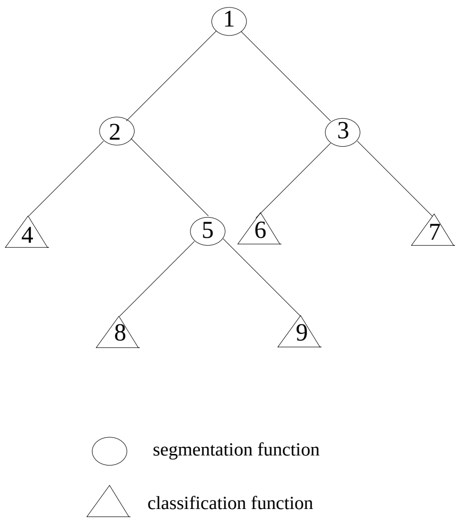

Figure 1: Architecture of an hybrid tree

The section 2 briefly presents the general principles of hybrid trees method. A detailed description of algorithmic implemenation can be found in [d'Alché-Buc et al. 1993].The section 3 concerns more specifically the extraction of classification rules and the interpretation of hybrid trees. We conclude in section 4 by summarize the characteristics of the approach.

2 Hybrid trees for building different kinds of neural networks

Hybrid trees are proposed to generalize neural trees to the building of neural networks based on different kinds of units for solving different tasks such as classification, regression, density estimation. We present here the architecture dedicated to classification applications. We illustrate our approach on a difficult binary discrimination problem of handwritten characters such as "H" and "M", "b" and "l", "S" and "5" in the framework of dynamical coding.

2.1 Hybrid tree architecture

The hybrid tree is a binary decision tree composed of two kinds of nodes that compute different tasks: internal nodes segment the input decision space and terminal nodes classify the data (see figure 1). In resolution mode, the classification of a new pattern consists to go from the root of the tree to the expert terminal node for final decision.

2.2 Building the tree with trio based learning

Building the hybrid tree is based on a recursive learning strategy called "trio-learning" [d'Alché-Buc et al., 1993)] that allows to optimize each node taking into account that it is followed by two sons. This general method allows to optimize the construction not locally, node by node as in standard neural trees or decision trees, but using a basic structure composed of three nodes called the *hybrid trio*.

A *hybrid trio* (whose structure is represented on the figure 2) is a binary decision tree restricted to three nodes: one father node that segments the decision space into two regions R^+ and R^- and two son nodes that classify the inputs that belong to the regions R^+ and R^-. Denoting by

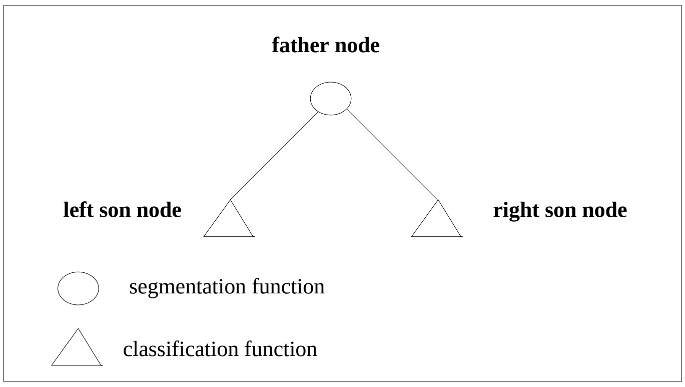

Figure 2: An hybrid trio architecture

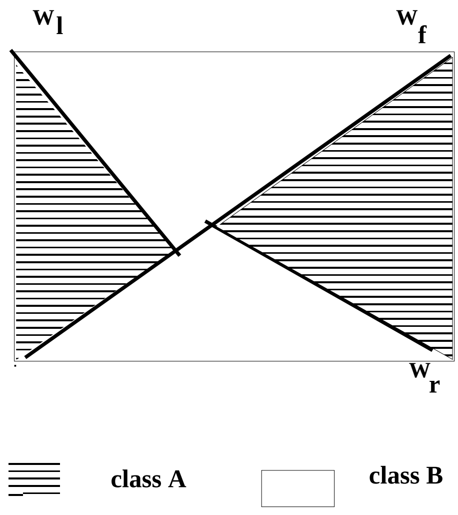

Figure 3: Classification induced by hybrid trio based on formal neurons

f^+ and f^- the functions computed by the two sons, and by f_{father} the function computed by the father node, the function f computed by the trio, is defined, for all the patterns x that are available at this stage:

$$f(x) = g^+(x).f_{father}(x) + g^-(x).(1 - f_{father}(x))$$

The hybrid trio structure can be closely related to the hierarchical mixture of experts proposed by [Jordan and Jacobs, 1993]. Different choices can be adopted for the functions f, g^+ and g^-. The father node can be a formal neuron defined by a sigmoidal activation function: in this case, the segmentation is performed using an hyperplane in the decision space. g^+ and g^- can be chosen as radial basis functions, or more generally as any classification function, e.g. a neural network. The figures 3 and 4 represent two different choices for the functions g: formal neurons and radial basis functions (RBF), in the plan.

Training such a trio can be performed by applying a gradient descent algorithm on a well chosen cost function such as the quadratic cost or an entropy-based cost , using the patterns of the training set (we suppose that the base of examples is divided into three sets of the same

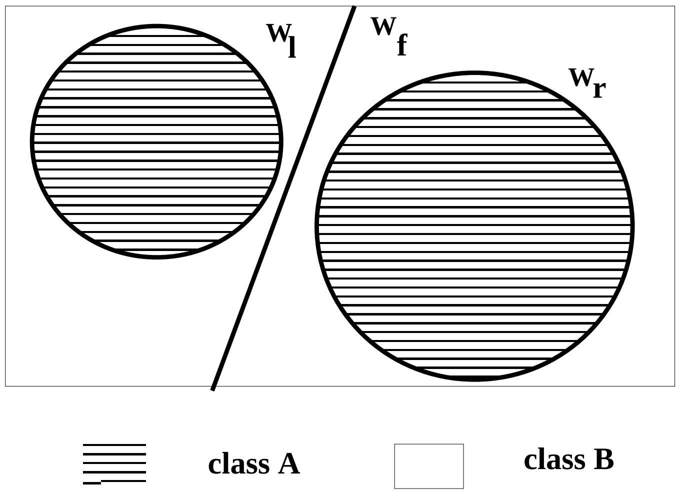

Figure 4: Classification induced by hybrid trio based on RBF

size, the training set, the test set and the validation set). When using radial basis functions as terminal nodes, the entropic cost has to be preferentiably used because it does not require to choose *a priori* which class will be represented by the interior of the region defined by radial basis function.

Let us describe the complete construction algorithm of a hybrid tree. The first steps of the learning process are illustrated on the figure 4.

The first step of the recursive construction algorithm consists in trying to solve the problem with only one son node. If this first step is successful, there is no need for a tree structure!

Otherwise, the second step consists in replacing the son node, by an hybrid trio. After training the trio, all the parameters are frozen : this means that one pattern has to be classified by the son node, dedicated to the region R^+ or R^- where it falls. The performance of each son node is then evaluated on the patterns of the test set that are available at this segmentation stage: if the performances are too low compared to a fixed threshold, then the son node is replaced by a new trio and the learning algorithm is recursively applied until reaching sufficiently good performances.

2.3 Pruning

Whereas the trio-learning improves the building of the tree in terms of complexity, it is still necessary to prune the tree after building for globally optimizing the structure, considering all the layers.The pruning algorithm we use depends on the test set. This algorithm aims at finding the subtree of the tree obtained by trio-learning, that obtains the best classification rate on the test data. It requires to store the classification rate obtained at each node of the tree before it is transformed into the father of a new trio.

As announced at the beginning of the section, construction based on trio strategy and pruning algorithms have been tested on handwritten character discrimination. We are interested in knowing how the performances evolve when the size of the learning set increases. Of particular interest are the asymptotic performances obtained in the limit of an infinite number of examples.

As an example of result, we show in the figure 6 the results for formal neurons used both in segmentation and classification nodes. The figure represents the value of the classification

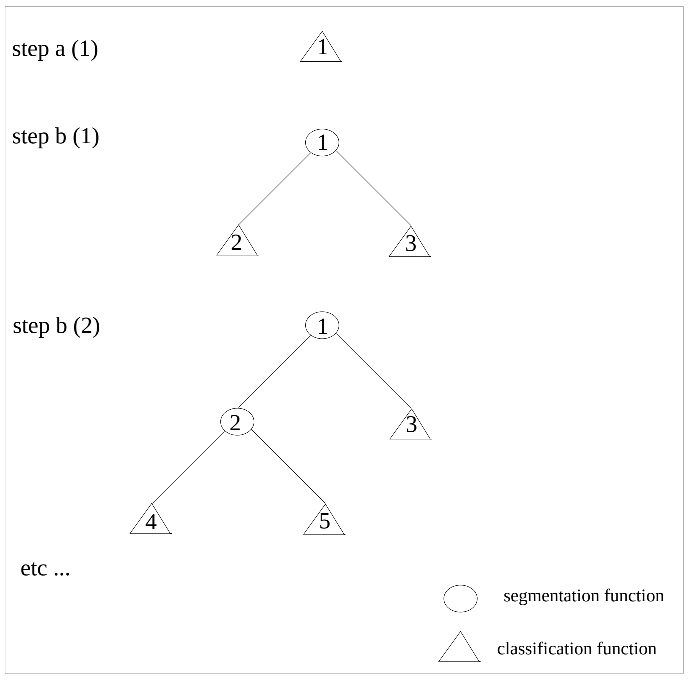

Figure 5: First steps of the application of trio-learning strategy

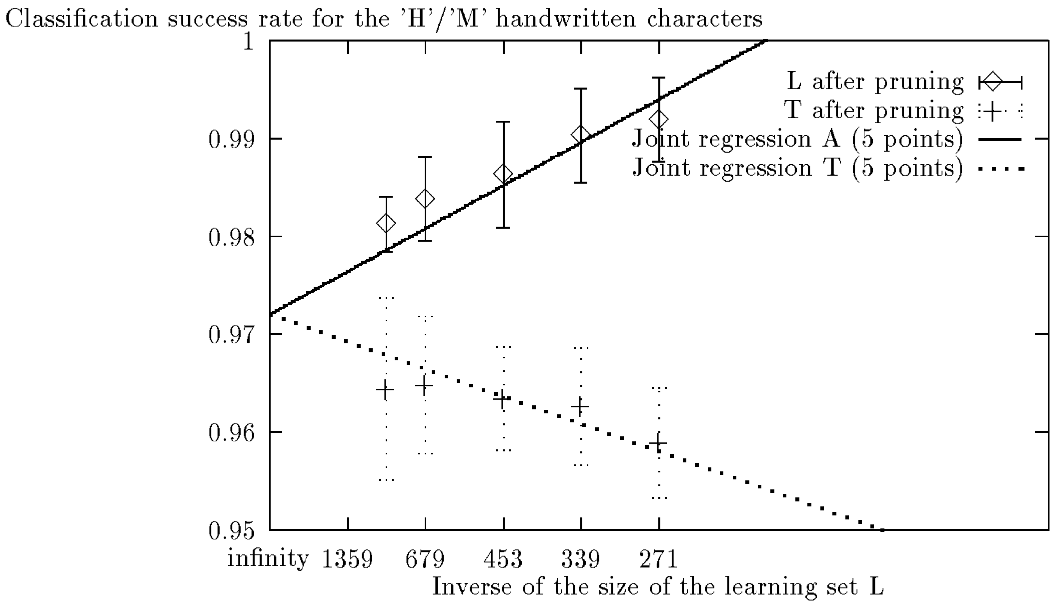

Figure 6: Average classification rates versus the size of the learning set for discrimination of handwritten characters "H"/"M"

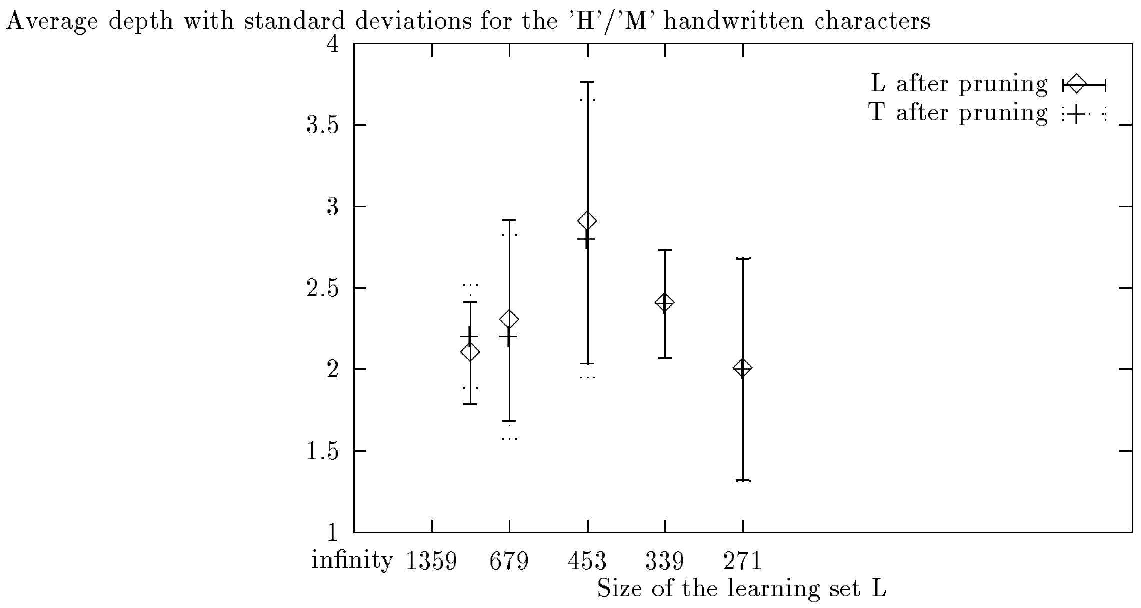

Figure 7: Average depth of the trees versus the size of the learning set for discrimination of handwritten characters "H"/"M"

rate obtained in function of the inverse of the size of the learning set, for the discrimination of two handwritten characters "H" and "M" which is a difficult case. Each point corresponds to the average of ten results obtained for ten random choice of the learning and test set (the complete base of examples is divided into two complementary parts). The standard deviation for each average point is also represented. We also tested our algorithm on three kinds of hybrid trees whose terminal nodes were respectively radial basis functions, formal neurons and formal neurons of second order (with a quadratic term) [d'Alché-Buc et al.].

Firstly, we can observe that as expected, the larger the learning set, the lower the classification rate obtained for this set;generalization is better when a lot of examples have been learnt.

Secondly, we have applied a joint linear regression to each group of points corresponding to the learning and test results : it appears that as obtained for multilayered perceptrons [Cortès et al., 1993], the correct classification rate behaves almost linearly in function of the the size of the learning set.

For all the experiments, we have also measured the average depth of the trees obtained after learning and pruning. For all kinds of tested hybrid trees, we observe that the depth of a pruned tree tends to a finite value when the size of the learning base tends to infinity. This is an interesting result since it shows that the structure of the tree is automatically adapted to the complexity of the data. The figure 7 shows the average depth of a pruned hybrid tree versus the size of the learning set.

3 Hybrid trees for rule extraction

3.1 Formal equivalence with a three-layered network

We denote by f the function computed by an hybrid tree. Each terminal node i of the tree is associated with a function f_i and a domain D_i of the input space define by the composition of segmentation functions used in the path from the root to this node. Once built, an hybrid tree

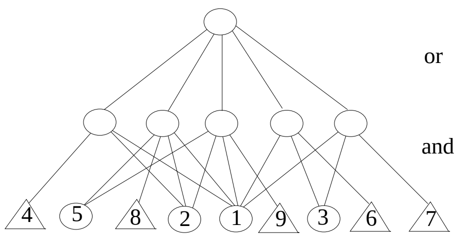

Figure 8: Architecture of the multilayered network equivalent to the hybrid tree

is formally equivalent to a three-layered neural network composed of:

- a first layer that includes of the nodes appearing in the tree
- a second layer composed of N units if N is the number of terminal nodes, where each unit corresponds to a terminal node and computes a binary conjunction (eventually with negations) between all the nodes of the first layers that belong to the path from the root to the associated terminal node.
- A third layer that provides the output, is composed of as many units that there are classes, each unit giving the value of the membership function of the input to the class.

The figure 8 represents the multilayered network equivalent to the tree of the figure 1.

3.2 Rule extraction and interpretation of a tree

As announced, this method can be applied to different classification functions. In every case, it is possible to extract decision rules from the terminal nodes. All the rules have the following form: let D_i be the region associated with the terminal node i and x be the input pattern:

if x belongs to D_i, then the class of x is given by $g_i(x)$

The interpretability depends on the choice of the function g_i : for instance, using radial basis functions improves the readability of the rule compared to formal neurons that induce arbitrary hyperplanes as discriminant separation surfaces.

Let us consider a radial basis function hybrid tree after automatic building and pruning to solve the 'S' and '5' discrimination problem. The maximal depth has been fixed to three for simplicity: at this stage, the classification rate obtained is 93cdp 5. The architecture of the tree is represented on the figure 9. The three terminal of the tree are associated with radial basis functions parameterized by a point (the center) and a parameter we call radius for simplicity. We use the above notations. The input vector corresponds to the sequence of ten points coding a character in the plan. Using the values of the parameters we have learnt, we can provide an explicit description of the rule associated with the terminal nodes. For instance, the rule associated with the terminal node 2 is:

if (x_1, y_1) belongs to B1(93.5, 117.6; 122.1) and (x_2, y_2) belongs to B2(78.2, 122.4; 122.1) and (x_3, y_3) belongs to B3(50, 116.4; 122.1) and ... and (x_{10}, y_{10}) belongs to B10 (7.3, 7.3; 122.1) then

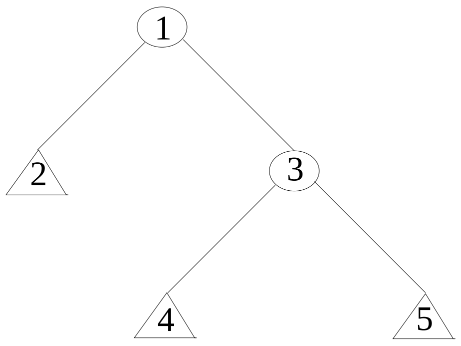

Figure 9: Architecture of an hybrid tree based on radial basis functions for'S' and '5' discrimination

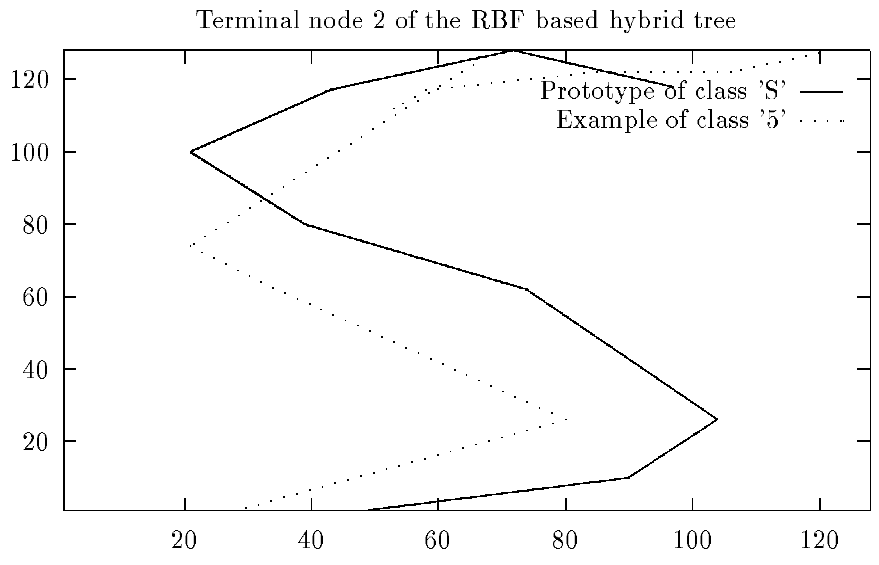

Figure 10: Terminal node 2 : prototype of class 'S' and example of class '5'

x is a 'S'. where Bi(cx,cy;r) is a ball of center (cx,cy), of radius r, whose membership function is a radial basis function (say, a gaussian).

The center of each RBF can be viewed as a prototype of the positive class. We have represented on the figures 10,11 and 12, the patterns the closest to the centers associated with the RBF. An example of the '5' class is also represented. Comparing the node 2 to the nodes 4 and 5, it appears that the right branch of the tree corresponds to the patterns with an acute angle whereas the left branch correspond to S and 5 with a right angle. This can be also revealed by looking at the analytic expression of the membership functions (RBF).

4 Conclusion

We have presented a generic constructive method for building hybrid neural trees where intermediate nodes segment the decision space into regions where "expert" terminal nodes can deal with the data. Trio-learning and pruning improves the optimization of the tree and reduces its

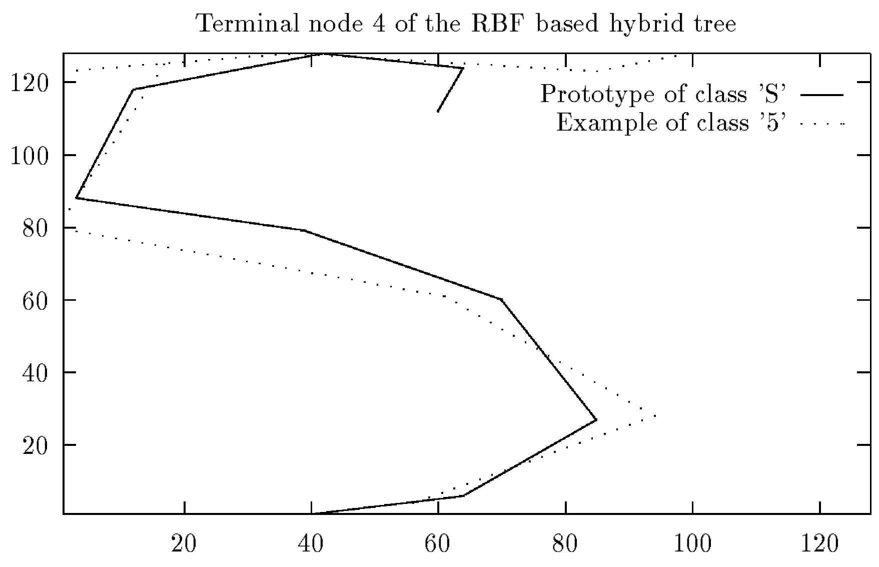

Figure 11: Terminal node 4 : prototype of class 'S' and example of class '5'

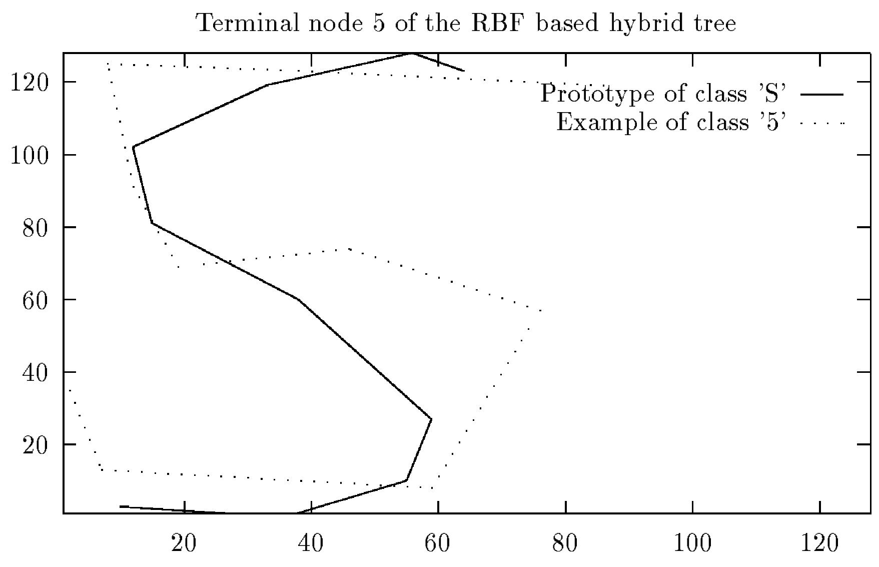

Figure 12: Terminal node 5 : prototype of class 'S' and example of class '5'

complexity. Moreover, as with decision trees used in Artificial Intelligence and Data Analysis, a base of decision rule can be extracted from hybrid neural trees: thus, it provides a way to extract knowledge more easily than in a multilayer network. In order to obtain interpretable rules, one should use interpretable terminal nodes such as radial basis functions or fuzzy neural networks . The integration of this last kind of network is the aim of our future work.

5 References

d'Alché-Buc, F., Andrès, V., and Nadal, J.-P. (1992). Learning Fuzzy Control rules with a Fuzzy Neural Network. In I. Aleksander and J. Taylor (Eds), *Artificial Neural Networks*, vol. 1, 715-719.

d'Alché-Buc, F., Zwierski, D., and Nadal, J.-P. (1993). Trio-learning: a new strategy for building hybrid neural trees. *Neural Networks for Computing Conference, Snowbird, Utah, submitted to Int. J. of Neural Systems.*

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. Stone. *Classification and regression Trees.* Wadsworth International group, Belmont, CA.

Cortès, C., Jackel, L. D., Solla, S., Vapnik, V., and Denker J. S (1993). Learning curves: asymptotic values and rates of convergence. In *Neural Networks for Computing Conference, Snowbird, Utah.*

Denoeux, T. (1992). Generation of Symbolic Rules in Backpropagation Networks. In I. Aleksander and J. Taylor (Eds), *Artificial Neural Networks II* , vol. 1, 707-711.

Frean, M. (1990). The upstart algorithm: a method for constructing and training feedforward neural networks.*Neurocomputing,(2)*, 423-438.

Gallant, S. I. (1988). Connectionist expert Systems. *Communications of the ACM,31*,152-169.

Higgins, C. M., Goodman, R. M., Smyth, P., and Miller J. W. (1992). Rule-based neural networks for classifiction and probability estimation. *Neural Computation, (4)*,781-804.

Jordan, M. I., and Jacobs, R. A. (1994). Hierarchical mixtures of experts and EM algorithm, Neural Computation, 1994.

Kane, R., Tchoumatchenko, I., and Milgram, M. (1992). Extracting knowledge from data using constrained neural networks. *Proc. of the European Conference on Machine Learning, Austria, Vienne, 5-8 April.*

Lin, C. T., and Lee, C. C.(1991). Neural network-based fuzzy logic control and decision systems, *IEEE Trans. Computers, 40(12)*,1321-1335.

Marchand, M., Golea. (1990). A growth algorithm for neural network decision trees. *Europhys. Lett.*, (12),105-109.

Quinlan, R. (1986). Induction of decision trees. *Machine Learning*, (1),81-106.

Sirat, J. A., and Nadal, J.-P. (1990). Neural trees: a new tool for classification. *Network*,1(1),198-209.