Global Induction of Oblique Decision Trees: An Evolutionary Approach

Marek Krętowski and Marek Grześ

Faculty of Computer Science, Białystok Technical University, Wiejska 45A, 15-351 Białystok, Poland

Abstract. A new evolutionary algorithm for induction of oblique decision trees is proposed. In contrast to the classical top-down approach, it searches for the whole tree at the moment. Specialized genetic operators are developed, which enable modifying both the tree structure and the splitting hyper-planes in non-terminal nodes. The problem of over-fitting can be avoided thanks to suitably defined fitness function. Experimental results on both synthetical and real-life data are presented and compared with obtained by the state-of-the-art decision tree systems.

1 Introduction

There exists dozens of decision tree induction algorithms [13]. However, a wide diversity among decision tree systems is somehow seeming. Almost all approaches are based on top-down strategy, where the learning set is associated with the root node and recursive procedure of optimal split searching, sub-nodes creation and feature vectors redistribution is applied until the stop condition is met. This greedy search technique is fast, easy to implement and, what is probably the most important, efficient in practical situations. On the other hand, it is evident that for many problems, the top-down induction fails to find the optimal solution and more sophisticated methods can be indispensable. However, it should be clearly stated that more complex algorithms are generally more time consuming.

In this paper, a global approach to induction of decision trees is investigated. In contrast to the classical step-wise manner of tree building, the whole tree (its structure and all splits) is searched at the time. Moreover, the top-down approach is often combined with the post-pruning applied in order to prevent the over-fitting problem. In the proposed method, any restructuring procedure is not necessary, because the assurance of the optimal tree size is embedded into the search process. As a result, the global approach allows to find suitable trees, both in terms of the classification power and the complexity.

The proposed method consists in designing a specialized evolutionary algorithm for decision tree building. Evolutionary algorithms (EA) are stochastic, search techniques, which were inspired by the process of biological evolution [11]. Their main advantage is ability to avoid the local optima, which

is especially important in such a difficult optimization problem as induction of decision trees.

The simplest type of decision trees is called *univariate*, because tests in non-terminal nodes are based on single attributes. Such a test is equivalent to partitioning the feature space with an axis-parallel hyper-plane. In case of non-axis parallel decision border, applying only univariate test can lead to their approximation by a very complicated stair-like structure. The aforementioned problem is eliminated by *oblique* (perceptron, linear) trees, where linear combinations of attributes are utilized in tests. The first such a system was CART [6], but it had a strong preference for univariate tests. One of the most well-known oblique tree system is *Oblique Classifier 1* (OC1) [12], which combines deterministic and randomized techniques in search for optimal splits. Other interesting top-down based oblique tree systems were proposed by Gama *et al.* [9] and Bobrowski *et al.* [3]. APDT (*Alopex Perceptron Decision Tree*) [16] can be seen as a first step toward more global induction of decision trees, because it evaluates goodness of a split based also on the degree of linear separability of sub-nodes.

The first application of evolutionary approach to linear tree induction was done in BTGA (*Binary Tree-Genetic Algorithm*) system [8], where standard genetic algorithm (SGA) with binary representation was used to find a splitting hyper-plane in each non-terminal node. In [7], the original OC1 system was successfully extended by using two standard algorithms: (1+1) evolution strategy and SGA. The system described in [10] utilized the specialized evolutionary algorithm for optimization of hyper-plane locations based on the dipolar criteria. After the greedy recursive partitioning the potentially over-specialized decision tree is post-pruned. That system can be treated as a top-down predecessor of the approach proposed in this paper.

A global approach to decision tree induction was investigated in genetic programming (GP) community. Nikolaev *at el.* applied [14] standard GP framework with specialized fitness function to evolve univariate decision tree-like programs. In [5] GP allows to induce classification trees with limited oblique splits.

The rest of the paper is organized as follows. In the next section the proposed evolutionary algorithm is detailed. Its experimental validation on both artificial and real-life problems is described in the section 3. In the last section, conclusions and plans for future work are presented.

2 Evolutionary Algorithm for Global Induction of Oblique Decision Trees

General structure of the proposed evolutionary algorithm follows the typical framework [11] and only application-specific issues are described in this section.

2.1 Preliminaries

A learning set is composed of M objects: N-dimensional feature vectors $\mathbf{x}^j = [x_1^j, ..., x_N^j]^T$ $(j = 1, ..., M)(\mathbf{x}^j \in R^N)$ belonging to one of K classes. The feature space could be divided into two regions by a hyper-plane:

$$H(\mathbf{w}, \theta) = \{\mathbf{x} : \langle \mathbf{w}, \mathbf{x} \rangle = \theta\}, \tag{1}$$

where $\mathbf{w} = [w_1, ..., w_N]$ $(\mathbf{w} \in R^N)$ is a weight vector, θ is a threshold and $\langle \mathbf{w}, \mathbf{x} \rangle$ represents an inner product. If $\langle \mathbf{w}, \mathbf{x}^i \rangle - \theta > 0$, it can be said that the feature vector $\mathbf{x}^i$ is on the positive side of the hyper-plane $H(\mathbf{w}, \theta)$.

A *dipole* [4] is a pair $(\mathbf{x}^i, \mathbf{x}^j)$ of feature vectors. A dipole is called *mixed* if and only if feature vectors constituting it belong to different classes and a pair of the vectors from the same class constitutes *pure* dipole. Hyper-plane $H(\mathbf{w}, \theta)$ splits the dipole $(\mathbf{x}^i, \mathbf{x}^j)$ if and only if:

$$(\langle \mathbf{w}, \mathbf{x}^i \rangle - \theta) \cdot (\langle \mathbf{w}, \mathbf{x}^j \rangle - \theta) < 0 \tag{2}$$

It means that the input vectors $\mathbf{x}^i$ and $\mathbf{x}^j$ are situated on the opposite sides of the dividing hyper-plane.

2.2 Representation, Initialization and Termination Condition

An oblique decision tree is a binary tree with splitting hyper-planes in non-terminal nodes and class labels in leaves. Each hyper-plane in the tree can be represented by a fixed-size $N + 1$-dimensional table of real numbers corresponding to the weight vector $\mathbf{w}$ and the threshold θ. However, the size and the structure of the classifier for a given learning set cannot be known in advance of induction. In such a situation, a variable-length representation is indispensable. Furthermore, during the induction process, additional information concerning, among other things, the learning vectors associated with each node are necessary. As a result, decision trees are not especially encoded in the form of traditional (binary or real-valued) chromosomes and they are represented in their actual form in the presented system.

An initial population is created by applying for each individual the following simple top-down algorithm combined with selection of optimal tree size according to the fitness function. An effective test in non-terminal nodes is searched based on randomly chosen mixed dipole $(\mathbf{x}^i, \mathbf{x}^j)$. The hyper-plane $H_{ij}(\mathbf{w}, \theta)$ is placed to split it:

$$\mathbf{w} = \mathbf{x}^i - \mathbf{x}^j \qquad \text{and} \qquad \theta = \delta \cdot \langle \mathbf{w}, \mathbf{x}^i \rangle + (1 - \delta) \cdot \langle \mathbf{w}, \mathbf{x}^j \rangle, \tag{3}$$

where $\delta \in (0, 1)$ is a randomly drawn coefficient, which determines the distance to the opposite ends of the dipole. $H_{ij}(\mathbf{w}, \theta)$ is perpendicular to the segment connecting dipole ends.

The EA terminates if the fitness of the best individual does not improve during the fixed number of generations (default value is equal to 1000) or the maximum number of generations is reached (default value: 10000).

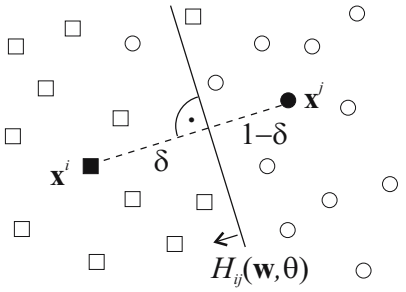

Fig. 1. Hyper-plane initialization based on randomly chosen mixed dipol

2.3 Fitness Function

The ultimate goal of any classification system is correct prediction of classes for new unlabelled feature vectors, which were not accessible during the learning phase. It is obvious that such a target function cannot be directly defined. Instead the classification quality on the training data is often applied as an estimate measure of the predictive power of the classifier. However, it should be underlined that optimizing only the re-classification quality leads to the over-fitting problem. In classical systems this problem is usually mitigated by post-pruning techniques. In our approach another solution is proposed. A complexity term is introduced into the fitness function. This term works as a certain type of penalty, which is proportional to the size of the tree. This way more compact trees are promoted and it allows avoiding the over-specialization.

Finally, the fitness function, which is maximized, has the following form:

$$Fitness(T) = Q_{Reclass}(T) - \alpha \cdot S(T), \tag{4}$$

where $Q_{Reclass}(T)$ is the re-classification quality, $S(T)$ is the size of the tree T expressed as a number of nodes and α - is a relative importance of the complexity term (default value is 0.005) and a user supplied parameter. It is rather obvious that there is no one, optimal value of α for all possible dataset. When the concrete problem is analyzed, tuning this parameter may lead to the improvement of the results (in terms of accuracy or classifier complexity).

2.4 Genetic Operators

It is now commonly accepted opinion, that task specific operators are highly useful in improving optimization process. In our system two complex genetic operators are currently employed: *MutateNode* and *CrossoverTrees*. Both of them can alter the tree structure as well as splitting hyper-planes in non-terminal nodes.

The first composite operator can be seen as a combination of the typical mutation-like operator with the dipolar operator introduced in [10]. *MutateNode* is applied to every single node of the tree with the given probability

(default value is equal to 0.1). The operator can cause with equal probability a modification of the test or a change of the node role. If a non-terminal node is concerned it can be pruned to a leaf or the corresponding hyper-plane can be altered. The hyper-plane position can be modified due to an application of the dipolar operator or by standard mutation. The dipolar operator chooses at random one dipole from the set of not divided mixed and divided pure dipoles. Then it shifts the hyper-plane by modifying only one randomly chosen weight w_i in such a way that the chosen mixed dipole is divided or division of pure one is avoided. When a leaf is concerned it can be only replaced by a new non-terminal node unless it is not reasonable.

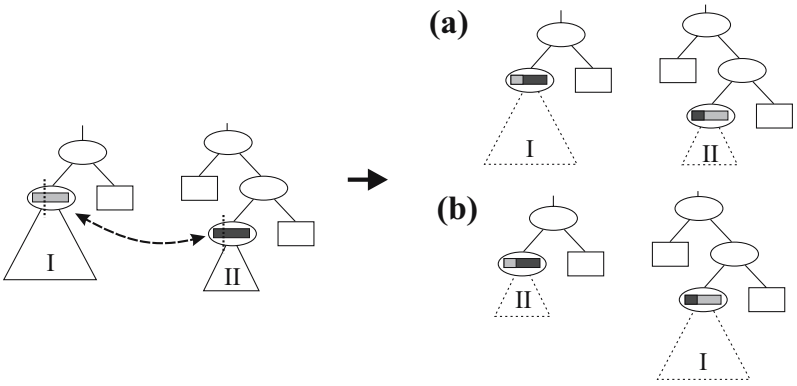

Fig. 2. *CrossoverTrees* operator: (**a**) exchange limited only to tests (**b**) exchange of the whole sub-trees

The second operator is an equivalent of the standard crossover operator and alters two individuals by replacing parts of the input trees. The exchange can be limited only to nodes or can encompass also sub-trees (see Fig. 2). At the beginning, the type of the exchange is randomly drawn (two variants are equally probable) and then, regardless of the type, one node in each tree is chosen also at random. If both nodes are non-terminal ones, the typical one-point crossover is applied on weights vectors and thresholds. In other cases nodes are just substituted. After the exchange concerning the nodes, depending on range of the operator, sub-tree starting from the altered nodes can be also replaced.

It should be underlined that trees modified by genetic operators require renewed determination of locations of all input feature vectors in the affected parts. As a result of this process some parts of the tree can be even pruned, because they do not contain any input feature vectors.

As a selection mechanism the proportional selection with linear scaling is applied. Additionally *elitist* strategy is used, which means that the best

tree in terms of the fitness function in each iteration is copied to the next population.

It was observed by Bennett *et al.* [1] that in oblique trees enlarging the margin, defined as the distance between decision boundary and the input feature vectors, is profitable in term of classifcation accuracy. In the presented system, a simple mechanism based on this observation was introduced. In each non-terminal node, two the closest feature vectors ($\mathbf{x}^+$ and $\mathbf{x}^-$) to the splitting hyper-plane $H(\mathbf{w}, \theta)$ are determined on the opposite sides of it. If the found dipole is mixed, the hyper-plane is centered by modifying the threshold:

$$\theta' = \frac{1}{2}[\langle \mathbf{w}, \mathbf{x}^+ \rangle + \langle \mathbf{w}, \mathbf{x}^- \rangle]. \tag{5}$$

It should be noted that such an operation does not change the fitness function.

3 Experimental Results

In this section experimental validation of the proposed approach on both artificial and real-life datasets is described. All results presented in the tables correspond to averages of 5 runs and were obtained by using test sets (mainly in case of synthetic datasets) or by 10-fold stratified cross-validation. Average number of nodes is given as a complexity measure. For the purpose of the comparison, results obtained by C4.5 (release 8) [15] and OC1 [12] with default parameters are also presented. If not otherwise stated, the proposed system (described as GEA-ODT in tables) is run with default values of parameters.

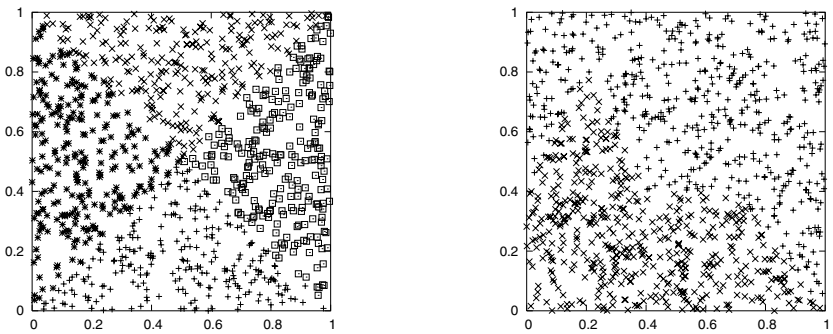

Fig. 3. Examples of artificial datasets (rotated *chessboard4* and *house*)

3.1 Artificial Datasets

In this subsection, synthetical datasets with analytically defined decision borders are analyzed. Analogous experiments are described in [12] and [16],

but original datasets are not available, hence similar configurations were generated by using random number generator. All these datasets are two-dimensional, except LS10 problem which is defined with 10 features. Number of feature vectors in the learning sets is 1000. Examples of synthetical datasets are depicted in Fig. 3.

Before presenting the definitive results, two experiments with parameters of the proposed system are presented. In the first one, usefulness of introducing the dipolar operator is investigated on the 2-class rotated chessboard problem. In Fig. 4 two learning curves (in fact, their initial parts) corresponding to induction without and with the use of dipolar operator are illustrated. As it could be expected applying the dipolar-based approach allows to speed up the search process.

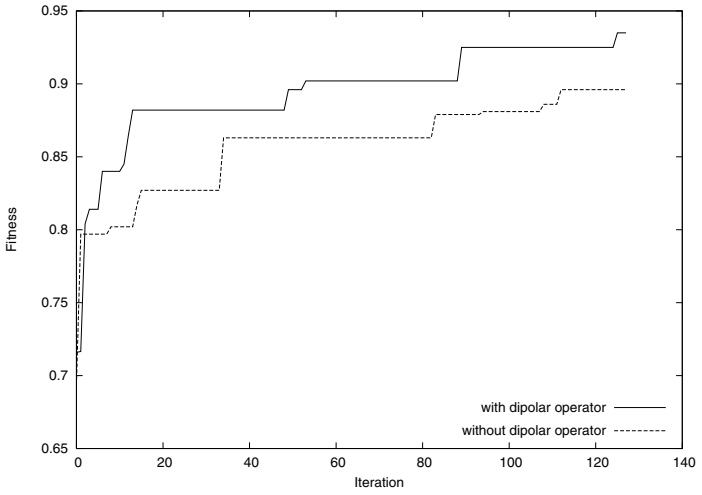

Fig. 4. Results of initial experiments on *chessboard2* problem: comparison of learning curves with and without dipolar operator

Sore point of any decision tree induction method is finding appropriate balance between re-classification quality and generalization power related to the tree complexity. In the global approach parametr α from the fitness function seems crucial for this problem. In the next experiment, we investigated how the classification quality and the tree complexity are influenced by changes of α. The relationships obtained on *chessboard2* dataset are presented in Fig. 5.

It can be observed that for relatively broad range of values (0.05-0.001) optimal trees were found. Further decrease of a parameter results in performance deterioration in terms of tree simplicity.

The final results of experiments concerning synthetical datasets are gathered in Table 1. For all domains GEA-ODT perform very well, both in accu-

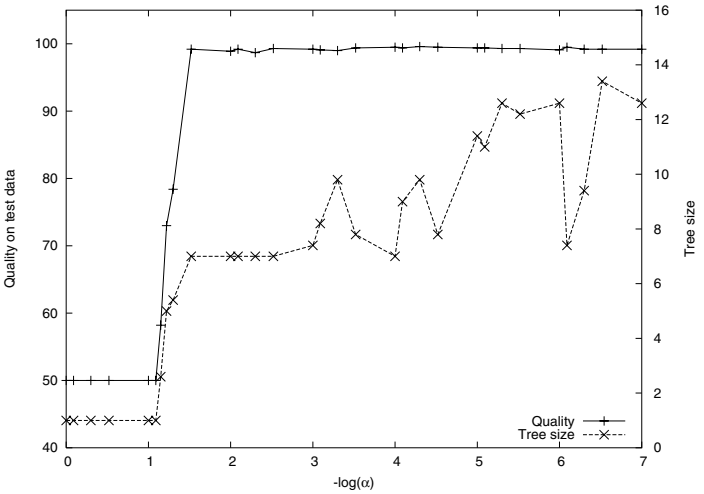

Fig. 5. Results of initial experiments on *chessboard2* problem: impact of α on the classification quality and the tree complexity

Table 1. Experimental results obtained for artificial datasets

Dataset	GEA-ODT Quality	GEA-ODT Size	OC1 Quality	OC1 Size	C4.5 Quality	C4.5 Size
chessboard2	99.1	7	99.3	11	93.3	47
chessboard4	99.3	7	99.5	7	94.6	87
zebra1	98.5	7	98.2	15	50	1
zebra2	99.0	5	99.8	7	96.9	55
zebra3	98.2	15.8	95.1	29	91.6	99
house	95.8	5.4	95.9	5	98	39
LS10	92.6	4.2	96	7	76.2	295

racy and tree complexity. It could be observed that the global method was able to find slightly less complicated trees than generated by other methods.

3.2 Real-life Datasets

In the second series of experiments a few well-known real-life datasets taken from UCI Machine Learning Repository [2] were analyzed and obtained results are collected in Table 2.

The proposed system performed very well on almost all analyzed datasets. Only for vehicle domain it was significantly worse that OC1 and C4.5. For this

dataset EA is converging very slowly and it was observed that the maximum number of iteration was used to stop the algorithm. When this constraint is relaxed the quality raise to more than 69%, but the computation time is really long (a few hours). Concerning complexities of the trees, it can be found that the global EA approach is generally more efficient than other systems.

Table 2. Results obtained for UCI datasets

Dataset	GEA-ODT Quality	GEA-ODT Size	OC1 Quality	OC1 Size	C4.5 Quality	C4.5 Size
breast-w	96.7	3	95.3	5	94.9	26
bupa	67.7	4.9	67.5	12.9	64.7	44.6
iris	97.0	5	96.6	5	94.7	8.4
page-blocks	94.6	3.7	97	23	97	82.2
pima	73.5	3.2	72.6	9.1	74.6	40.6
sat	83.1	11	85.4	45	85.5	435
vehicle	65.4	14.7	70.2	30.4	72.3	129
waveform	81.5	6.2	78	5	73.5	107

4 Conclusions

In the paper, new evolutionary algorithm for induction of oblique decision trees is presented. The greedy top-down technique is replaced by the global approach, where the whole tree is searched at the moment. The experimental validation indicates that the accuracy of the proposed method is at least comparable with the results obtained by leading decision tree systems. In terms of the tree complexity it seems that global algorithm is able to find more compact classifiers than the competitors.

The presented system is constantly improved and currently feature selection is embedded into the algorithm, which will allow eliminating redundant and noisy features at each non-terminal node. Furthermore several directions of possible future research exist. One of them is designing more robust fitness function, which has a critical influence on the performance of the system. We also want to incorporate into the induction process the variable misclassification cost and feature's cost.

The proposed approach is not the fastest one now but hopefully it is known that evolutionary algorithms are well suited for parallel architecture. We plan to speed up our system by re-implementing it in the distributed environment.

Acknowledgments This work was supported by the grant W/WI/05 from Białystok Technical University.

References

1. Bennett, K., Cristianini, N. et al. (2000) Enlarging the margins in perceptron decision trees. Machine Learning **41**, 295–313
2. Blake, C., Keogh, E. et al. (1998) UCI repository of machine learning databases,[http://www.ics.uci.edu/˜mlearn/MLRepository.html]. Irvine, CA: University of California, Dept. of Computer Science
3. Bobrowski, L., Krętowski, M. (2000) Induction of multivariate decision trees by using dipolar criteria. In: Proc. of PKDD'00. Springer LNCS 1910, 331–336
4. Bobrowski, L. (1996) Piecewise-linear classifiers, formal neurons and separability of the learning sets, In: Proc. of 13^{th} Int. Conf. on Pattern Recognition 224–228
5. Bot, M., Langdon, W. (2000) Application of genetic programming to induction of linear classification trees. In: EuroGP 2000. Springer LNCS 1802, 247–258
6. Breiman, L., Friedman, J., Olshen, R., Stone C. (1984) Classification and Regression Trees. Wadsworth Int. Group
7. Cantu-Paz, E., Kamath, C. (2003) Inducing oblique decision trees with evolutionary algorithms. IEEE Trans. on Evolutionary Computation **7**(1), 54–68
8. Chai, B., Huang, T. et al. (1996) Piecewise-linear classifiers using binary tree structure and genetic algorithm. Pattern Recognition **29**(11), 1905–1917
9. Gama, J., Brazdil, P. (1999) Linear tree. Inteligent Data Analysis **3**(1), 1–22
10. Krętowski, M. (2004) An evolutionary algorithm for oblique decision tree induction, In: Proc. of ICAISC'04. Springer LNCS 3070, 432–437
11. Michalewicz, Z. (1996) Genetic Algorithms + Data Structures = Evolution Programs. 3^{rd} edn. Springer
12. Murthy, S., Kasif, S. et al. (1994) A system for induction of oblique decision trees. Journal of Artificial Intelligence Research **2**, 1–33
13. Murthy, S. (1998) Automatic construction of decision trees from data: A multidisciplinary survey. Data Mining and Knowledge Discovery **2**, 345–389
14. Nikolaev, N., Slavov, V. (1998) Inductive genetic programming with decision trees. Intelligent Data Analysis **2**, 31–44
15. Quinlan, J. (1993) C4.5: Programs for Machine Learning. Morgan Kaufmann
16. Shah, S., Sastry, P. (1999) New algorithms for learning and pruning oblique decision trees. IEEE Trans. on Systems, Man, and Cybernetics - Part C **29**(2), 494–505