A Further Study of Pruning Methods in Decision Tree Induction

Floriana Esposito, Donato Malerba and Giovanni Semeraro

Dipartimento di Informatica, Università degli Studi
via Orabona 4, 70126 Bari - Italy
{esposito, malerbad, semeraro@vm.csata.it}

Abstract

In this paper, a comparative study of six well-known post-pruning methods is presented. Each method is briefly described and its weaknesses and strengths are pointed out. Moreover, some results on a new experimentation on pruning methods have been reported. Eleven databases available via ftp at the UCI Machine Learning Repository have been selected. By comparing error rates of pruned trees with the corresponding unpruned trees we have found no clue supporting Mingers' claim that pruning methods exploiting an independent pruning set perform better than the others. On the contrary, our results confirm Holte's observation that even simple rules perform well on most commonly used data sets in the machine learning community.

1. Introduction

The problem of inducing the smallest decision tree consistent with a training set has been proven to be NP-complete when tree complexity is measured as the external path length, that is, the sum, over all leaves, of the number of edges on the path from the root to that leaf (Hyafil & Rivest, 1976). The NP-completeness has also been conjectured for other complexity measures, such as the number of bits necessary to encode a decision tree (Quinlan & Rivest, 1989). For this reason, various heuristic methods have been proposed for the construction of a decision tree, among which the most widely known is the *top-down* approach. In top-down induction of decision trees (TDIDT) it is possible to identify three tasks (Breiman et al., 1984):

1) definition of a decision process associated with the tree,
2) determination of the test to associate with each edge coming out from a node,
3) determination of the leaves.

This paper is mainly concerned with the third one. There are two different ways to cope with it: either deciding when to stop the growth of a tree or reducing the size of a fully expanded tree, T_{max}, by pruning some branches. Methods that control the growth of a decision tree during its construction are called *pre-pruning* methods, while the others are called *post-pruning* methods (Cestnik et al., 1987). The former methods establish stopping rules for preventing the growth of branches that do not seem to improve the predictive accuracy of the decision tree. One problem is that the search myopia of the hill-climbing search strategy adopted in TDIDT algorithms may prevent from finding a decision tree that is consistent with the training set, even though one exists. One way around this short-sightedness is that of considering multivariate (*polythetic*) tests when no univariate (*monothetic*) test seems good enough (Fisher & Chan, 1990). Another way, which is implemented in many TDIDT systems, consists in keeping on growing a tree T_{max} in any case, and then retrospectively *pruning* those branches that seem superfluous with respect to predictive accuracy (Niblett, 1987). The final effect is that in this way the intelligibility of a decision tree is improved, without really affecting its predictive accuracy.

Some authors push forward by claiming that tree pruning is a way of improving predictive accuracy of the tree, since it eliminates even harmful branches (Cestnik et al., 1987; Mingers, 1989). Informally, a branch can be defined harmful when it tests irrelevant attributes, fitting the noise in the training data and lowering accuracy on unseen cases. Thus, the presence of harmful branches, not only reduces the comprehensibility of a tree by making it more complex, but it also degrades the classification performance on new observations. When this happens, we say that the complex tree *overfits* the data. Recently, Schaffer has shown that overfitting avoidance by means of tree pruning is a form of *bias* (here intended as a set of factors that influence hypothesis selection) rather than a statistical improvement of the classifier (Schaffer, 1993). In fact, the definition of a complexity measure for a decision tree amounts to accepting a certain prior probability distribution over the set of possible subtrees (Malerba et al., 1994).

The variety of post-pruning (or simply pruning) methods proposed in literature does not encourage the comprehension of both the common and the individual aspects. In fact, while some methods proceed from the root towards the leaves of T_{max} when they examine the branches to prune (*top-down approach*), other methods follow the opposite direction, starting the analysis from the leaves and finishing it with the root node (*bottom-up approach*). Furthermore, while some methods use only the *training set* in order to evaluate the accuracy of a decision tree, other methods exploit an additional *pruning set*, sometimes improperly called test set, that allows them to get less biased estimates of the predictive accuracy of a pruned tree.

In this paper, we compare six of the most renowned pruning methods, by emphasizing their weaknesses and strengths. In particular, we point out the main assumptions underlying the method, the appropriateness of some error rate estimations, as well as the behaviour of the method under conditions different from those considered by the proposing author(s). Then, we provide a summary of some empirical results we obtained by using an experimental procedure fairer than that adopted in a previous work by Mingers (1989).

2. A critical review of some pruning methods

This section is devoted to the presentation of six pruning methods we tested in our experiments. In particular, each method is firstly explained by means of a brief description that reflects its original formulation, then it is criticized with the aim of emphasizing properties and limits.

2.1 Reduced error pruning (REP)

This method, proposed by Quinlan (1987), uses an independent set of examples, here called *pruning set*, in order to evaluate the goodness of a subtree of T_{max}. It works as follows: for each internal node *t*, the number of classification errors made on the pruning set, when the subtree T_t rooted in *t* is kept, is compared with the number of misclassifications made when *t* is turned into a leaf and associated with the *best* class. Sometimes, it may happen that the tree obtained by pruning *t* has a better performance than the original tree, that is, the number of errors on the pruning set made by the pruned decision tree is lower than the number of errors made by the original tree. In this latter case the harmful branch is removed. The branch pruning operation is repeated until further pruning increases the classification error rate on the pruning set. It is worthwhile to recall that Quinlan limits the pruning condition given above by imposing a further constraint: a branch T_t can be pruned only if that branch contains no subtree that results in a lower error rate than T_t itself (*bottom-up* method). This contrasts with the description of the same method reported in (Mingers, 1989). In fact, Mingers maintains that, among all the nodes, the one having the largest difference between the number of errors when the subtree is kept and the number of errors when the node is pruned must be chosen. This variant causes the loss of an important property of Quinlan's original method, namely the generation of the smallest subtree with the lowest error rate with respect to the pruning set. As we will see later, such a property can be effectively exploited in the experimental comparison in order to find the optimally pruned tree with respect to the test set.

2.2 Pessimistic error pruning (PEP)

This pruning method, proposed by Quinlan (1987) as the previous one, does not need an independent pruning set. However, since the misclassification rate estimate on the training set is optimistically biased, Quinlan introduces the continuity correction for the binomial distribution that, in his opinion, should provide a more realistic estimate of the classification error rate. Let *e(t)* be the number of training examples erroneously classified at the node $t \in T$. Then, according to the continuity correction, the number of errors made in *t* is given by:

$$n'(t) = e(t) + 1/2$$

while for the subtree T_t it is:

$$n'(T_t) = \Sigma(e(s) + 1/2)$$

where the summation is intended on the set of leaves of T_t.

It should be observed that, if the development of a tree goes on until all its leaves do not make errors on the training set, then $e(s) = 0$ for each leaf s. As a consequence, in that case $n'(T)$ only represents a measure of the tree complexity that associates each leaf with a cost equal to 1/2.

As expected, the subtree T_t makes less errors on the training set than the node t when t is transformed into a leaf, that is, $e(t) > \Sigma\, e(s)$. Nevertheless, it may happen that $n'(t) \leq n'(T_t)$ due to the continuity correction, since the complexity of T_t is also taken into account. In that case, the node t ought to be pruned, but unfortunately, the estimate $n'(T_t)$ of the number of misclassifications made by the subtree remains optimistic. This is the reason for which Quinlan weakens the condition that rules the pruning of a subtree T_t and requires that:

$$n'(t) \leq n'(T_t) + SE(n'(T_t))$$

where

$$SE(n'(T_t)) = [n'(T_t) \cdot (N(t) - n'(T_t)) / N(t)]^{1/2}$$

is the standard error for the subtree T_t under the assumption that errors on the training set are binomially distributed. The algorithm evaluates each node starting from the root of the tree and, if a subtree is chosen for pruning, its internal nodes are not examined. This *top-down* approach gives the pruning technique a high run speed.

This method raises some perplexities. First, the introduction of the continuity correction has no theoretical justification. In statistics, it is used to approximate a binomial distribution with a normal one, but it has never been applied to correct optimistic estimates of error rates. In fact, the continuity correction is useful only to introduce the tree complexity factor. Nonetheless, in the method, this factor is improperly compared to an error rate. It follows that results are not always the expected ones. For instance, let us consider a sample of 1024 training examples, described by 20 binary features randomly generated, that are assigned to two distinct classes with equal probability. In this situation, it would be possible to induce a binary tree T_{max} with 990 leaves that correctly classifies all examples but does not have any predictive capability due to the random generation. A *good* pruning method should reduce the tree just to one leaf. Nevertheless, the PEP method will never prune the root t_0 since $n'(t_0) = 512.5$, $n'(T_{max}) = 495$ and $SE(n'(T_{max}))\approx 16$.

Finally, the top-down approach to tree pruning is not justified when there is no guarantee that all subtrees of a pruned branch T_t should be pruned. Indeed, it is not difficult to find examples in which this method prunes subtrees which contain other subtrees that should not be pruned according to the same criterion (Esposito et al., 1993).

2.3 Minimum error pruning (MEP)

Niblett and Bratko (1986) proposed a *bottom-up* approach seeking for a single tree that minimizes the expected error rate on an independent data set. In the following, we will refer to an improved version of the minimum error pruning reported in (Cestnik & Bratko, 1991).

For a k-class problem, the expected probability that an observation reaching a node t belongs to the i-th class is the following:

$$p_i(t) = [n_i(t) + p_{ai} \cdot m] / [N(t) + m]$$

where:

- $n_i(t)$ is the number of training examples in t classified into the i-th class,
- p_{ai} is the *a priori* probability of the i-th class,
- m is a parameter of the estimate method,
- $N(t)$ is the number of training examples reaching t.

The parameter m determines the contribution of the *a priori* probability of the i-th class to the estimate of the conditional probability of the i-th class in a node t computed as the relative frequency $n_i(t) / N(t)$. For the sake of simplicity, m is assumed to be equal for all the classes. Cestnik and Bratko name $p_i(t)$ as *m-probability estimate*. When a new observation reaching t is classified, the expected error rate is given by:

$$EER(t) = \min_i \{ 1 - p_i(t) \} = \min_i \{ 1 - [n_i(t) + p_{ai} \cdot m] / [N(t) + m] \} = \min_i \{ [N(t) - n_i(t) + (1 - p_{ai}) \cdot m] / [N(t) + m] \}$$

In the minimum error pruning method, the expected error rate for each internal node t is computed. It is called *static error*, $STE(t)$. Then, the expected error rate when t is not pruned, called *dynamic* (or *backed-up*) *error*, $DYE(t)$, is computed. It is given by a weighted sum of the expected error rates of the children, where the weights are the probabilities that an observation will reach the corresponding child. In the following, the weights p_i will be estimated by the proportion of training examples reaching the i-th child, as originally proposed by Niblett and Bratko.

Having obtained several subtrees for distinct values of *m*, we are faced by the problem of choosing the best one. Cestnik and Bratko suggest the intervention of a domain expert who can choose the right value of *m* according to the level of noise in the data or even study the selection of produced trees. Since we do not dispose of an expert, in our experiments we adopted a different criterion. The classification accuracy of the trees has been evaluated on an independent pruning set and the smallest tree with the lowest error rate on the pruning set has been selected.

The newer version of the minimum error pruning seems to overcome two problems that affect the original proposal by Niblett and Bratko: optimistic bias (Watkins, 1987) and dependence of the expected error rate on the number of classes (Mingers, 1989).

As to the first problem, let us consider a set of high-dimensional binary feature vectors. Each feature vector is an example of one of two classes: the class is assigned randomly to each example. If we induce a decision tree from this set, then the *true* error rate of the tree would be 50% since the tree has no predictive power. Let us suppose that each leaf of the tree covers a single training example (this hypothesis seems quite realistic since no binary attribute has a predictive power for the problem, thus, an exact classification of an example can only be obtained by looking at its feature vector as a whole). In the original version of the minimum error pruning the static error of the root equals the true error rate, while the dynamic error of the root itself is at most 33.33%. Therefore, the tree can never be pruned in the root, even if this is desirable. In the improved version of the method we have:

$$STE(root) = [1024-512+(1-1/2)\cdot m]/(1024+m) = 1/2 \qquad DYE(root) = [0+(1-1/2)\cdot m]/(1+m) = m/[2\cdot(m+1)]$$

therefore the root will be pruned when we recognize that the degree of noise is very high and we set *m* to infinity.

As to the second problem, we observe that, in the improved version, the expected error rate is not affected by the number of classes but by a free parameter, *m*, whose choice is committed to an expert or based on an independent data set.

2.4 Critical value pruning (CVP)

This bottom-up method, proposed by Mingers (1987), operates as follows. A threshold, called *critical value*, for the node selection measure is initially set. Then, an internal node of the tree is pruned if the value taken by the selection measure in that node does not exceed the critical value. Nevertheless, it may happen that the pruning condition is met by a node *t* but not by all its children. In that case, the branch T_t is kept because it contains relevant nodes. Obviously, the higher the critical value the more drastic the pruning.

The method proposed by Mingers consists of two main steps:

1) prune T_{max} for increasing critical values
2) choose the best tree among the sequence of the pruned trees by measuring both the *significance* of the tree as a whole and its *predictive ability*.

As to the first phase, if the evaluation measure used for developing the tree is the *gain ratio* (Quinlan, 1993), and the tree is grown until all examples in each node belong to the same class, then all nodes that are parent of leaves will have a gain ratio equal to 1.0 while the others will have a lower value. This means that, with the gain ratio as selection measure, we can choose only between two trees: T_{max} and the root tree. Thus, the method does not seem sufficiently general to be applied to trees built by using any criterion for attribute selection. Nonetheless, this is in contrast with the experimental results provided by Mingers (1989, Table 1).

For the second phase, the author proposes to measure the significance of the tree by means of the G statistics. It is computed for a large contingency table having a row for each leaf in the tree and a column for each class. It evaluates the degree of interdependence between the leaves of a tree and the classes of the problem, so that it is high for fully expanded trees that correctly classify the whole set of examples. Mingers presents also a test on the significance measure, but unfortunately, such a test is only able to establish whether the predictive ability of a tree is meaningful, but it cannot be used to choose among trees that pass the test. Indeed, it is not enough to compare the G values taken by two contingency tables, since such values depend on the degrees of freedom of the G statistics, so that they are usually higher in unpruned trees, because of the higher number of rows in the contingency table (obviously the number of columns does not change).

For what concerns the evaluation of the predictive accuracy, two alternative solutions are proposed:

- to estimate the probability of a correct prediction *p* and then to estimate the probability that the number of correct predictions equals the number of observations correctly classified by the pruned tree,
- to compute the error rate by using an independent test set.

The formulation provided by Mingers for the first solution is rather weak. Indeed, he estimates the probability that class C_i is correctly predicted as follows:

$P(x \in C_i \wedge x$ is classified as an instance of $C_i) = P(x \in C_i) \cdot P(x$ is classified as an instance of $C_i / x \in C_i) = n_i/n \cdot n_i/n = n_i^2/n^2$

While approximating $P(x \in C_i)$ by means of the relative frequency of the examples of class C_i is plausible, using such a frequency to approximate the *a posteriori* probability is untenable.

Finally, we observe that Mingers also suggested to estimate the error rate on an independent pruning set and to use this measure to choose the best subtree in the resulting sequence of trees. Nevertheless, the sequence detected in the first step of this method might not contain the best tree with respect to the pruning set; therefore we have no guarantee of finding the smallest and most accurate subtree as for the reduced error pruning.

2.5 Cost-complexity Pruning (CCP)

This method, proposed by Breiman et al. (1984) and called *cost-complexity pruning*, is characterized by two phases:

1) selection of a family of subtrees of T_{max}, $\{T_{max} \equiv T_0, T_1, T_2, \ldots, T_L\}$, according to some heuristics,
2) choice of the best tree T_i according to an estimate of the true error rates of the trees in the family.

In the first phase, each T_{i+1} is obtained from T_i by pruning those branches that show the lowest increase in apparent error rate per pruned leaf. In other words, given T_i, the following ratio is computed for each internal node *t*:

$$\alpha = (r(t) - r(T_t))/(L_t - 1)$$

where $r(t)$ and $r(T_t)$ are apparent error rates of the node t and the branch T_t respectively, while L_t is the number of leaves in the branch T_t. Then, T_i is pruned in all nodes with the lowest value of α, and a new tree T_{i+1} of the family is obtained. The process stops when the root tree is obtained. It is possible to prove that each tree T_i is characterized by a distinct value α_i, such that $\alpha_i \leq \alpha_{i+1}$. Therefore, $\{T_0, T_1, T_2, \ldots, T_L\}$ is actually a *parametric family* of trees, denoted as $T_{max}(\alpha)$.

The second phase chooses the best tree in $T_{max}(\alpha)$ with respect to predictive accuracy. Breiman et al. propose two distinct ways of estimating the true error rate of each tree in the family, one based on the use of an independent pruning set as for the reduced-error pruning, while the other based on cross-validation sets. In the first case, however, the cost-complexity pruning method is theoretically under a disadvantage with respect to the reduced-error pruning because it can only choose a tree in the set $\{T_0, T_1, T_2, \ldots, T_L\}$ instead of the set of all possible subtrees of T_{max}, with the consequence that, if the most accurate subtree with respect to the pruning set is not in $\{T_0, T_1, T_2, \ldots, T_L\}$, it cannot be selected. On the other hand, when cross-validation sets are used, the tree selection process is based on a strong assumption whose amount of bias is unpredictable (Malerba et al., 1994).

There is another aspect of the pruning method adopted in CART that deserves special attention. Breiman et al. decide to choose the smallest tree in the set $\{T_0, T_1, T_2, \ldots, T_L\}$ such that its error rate is not greater than one standard-error with respect to the lowest error observed for trees T_i. As we will see later, the effect of such a rule of thumb, called 1SE, is a strong tendency to overprune.

2.6 Error-Based Pruning (EBP)

This pruning method has been implemented in C4.5 (Quinlan, 1993) and presents a novelty with respect to the others seen so far, since it allows the replacement of a subtree by one of its branches. In this way it is possible to overcome a limit of the bottom-up strategy that does not prune ancestors of nodes which appear to be good. Indeed, if *t* is a node that should be pruned according to some criterion, while *t'* is a child of *t* that should not be pruned according to the same criterion, bottom-up pruning methods generally avoid pruning *t* since T_t contains sub-branches with significant classification performance. On the contrary, the error-based pruning acts by *grafting* $T_{t'}$ onto the place of *t*, so saving the good sub-branch and deleting the useless node *t*.

Unfortunately, the main weakness of such a method is the criterion used to predict the error rate of nodes and branches. Indeed, if a node *t* makes e(t) errors on n(t) training cases that reach it, Quinlan estimates the upper confidence limit of the expected error probability of the node by means of e(t) and n(t) themselves. The underlying assumption is that the same sample used to build the classifier can be considered as a random sample on which to test the tree. As the same author admits, such an assumption is untenable, but, strangely enough, Quinlan himself claims that C4.5 employs a far more pessimistic estimate of errors than the pessimistic error pruning. As we will show in the next section, from our experimental results we draw the very opposite conclusion.

3. Experimental results

The need of making some experiments on pruning methods arises from the fact that the experimental procedure designed by Mingers (1989) for his empirical study is not very fair (see (Esposito et al., 1993) for a detailed discussion). Moreover, since we are conscious that different pruning methods represent different biases, we will avoid to draw conclusions on which is the "best" method, but, following the main stream of the analytical comparison between methods pursued in the paper, we will try to understand which are the biases that affect them.

In our experimentation, each data set is still randomly divided into three subsets, according to the following criterion: *growing* set (49%), *pruning* set (21%) and test set (30%). The union of the growing and pruning set is called *training* set, and its size is just 70% of the whole data set. The growing set contains the 70% of cases of the training set, while the pruning set the remaining 30%. Both the growing set and the training set are used to learn decision trees, that, for simplicity of naming, we will call *grown* tree and *trained* tree, respectively. The former is used by those methods that need an independent pruning set in order to prune a decision tree, namely REP, MEP, CVP, and the ECP based on an independent test set and adopting the 1SE rule (1SE) or not (0SE). The latter is used by those methods that exploit the training set only, such as PEP, EBP, and the CCP based on cross-validation sets and adopting the 1SE rule (CV-1SE) or not (CV-0SE). The evaluation of the error rate is always made on the test set.

For each data set considered, 25 experiments are made by randomly partitioning the data set into three subsets. Moreover, in each experiment two statistics are recorded for pruned, grown and trained trees: the number of leaves (*size*) of the resultant tree and the error rate (*e.r.*) of the tree on the test set. A two-tailed paired t-test can be used to evaluate the significance of the error rate difference between each pruned tree and the corresponding trained tree.

As to the MEP, the following *m* values have been chosen: 0.5, 1, 2, 3, 4, 8, 12, 16, 32, 64, 128, 512 and 1024. Experiments on the CVP were made by setting a maximum critical value equal to 1.0 and a step equal to 0.01. The only selection measure considered is the gain ratio.

In Table 1, the main characteristics of the databases considered in our experiments are reported. They are available at the UCI Machine Learning Repository (Murphy & Aha, 1994) and some of them have even been used to compare different pruning methods (Quinlan, 1987; Mingers, 1989). The data base Heart-Disease is actually the join of four data sets with the same number of attributes but collected in four distinct places (Hungary, Switzerland, Cleveland and Long Beach). Of the 76 original attributes, only 14 have been considered, since they are the only ones to be considered useful for the classification. Moreover, examples have been assigned to two distinct classes: *no presence* (value 0) and *presence* of heart diseases (values 1,2,3,4).

Table 1. Main characteristics of the databases used for the experimentation.

database	No. Cases	No. Classes	No. Attributes	Continuous attributes	Multi-valued attributes	Null values	% Base Error	Noise level	Uniform distrib.
Iris	150	3	4	4	0	no	66.67	low	yes
Glass	214	7	9	9	0	no	64.49	low	no
Led	1000	10	7	0	0	no	90	10%	yes
Hypo	3772	4	29	7	1	yes	7.7	no	no
P.-gene	106	2	57	0	57	no	50	no	yes
Hepat.	155	2	19	6	0	yes	20.65	no	no
Cleveland	303	2	14	5	5	yes	45.21	low	yes
Hungary	294	2	14	5	5	yes	36.05	low	no
Switzerland	123	2	14	5	5	yes	6.5	low	no
Long Beach	200	2	14	5	5	yes	25.5	low	no
Heart	920	2	14	5	5	yes	44.67	low	yes

Table 2. Error rate variations for different pruning methods (significance level: 0.10)

database	REP	MEP	CVP	OSE	1SE	PEP	CV 0SE	CV 1SE	EBP
Iris	0	0	0	0	0	0	0	-	0
Glass	-	0	0	0	-	0	-	-	0
Led	-	-	-	0	-	0	0	-	0
Hypo	+	+	0	+	0	+	+	-	+
P.gene	0	0	0	0	0	0	0	0	0
Hepatitis	0	0	0	0	0	0	0	0	0
Cleveland	0	0	0	0	0	0	0	-	0
Hungary	+	+	0	+	+	+	+	+	+
Switzerland	+	0	+	+	+	+	+	+	+
Long Beach	+	+	+	+	+	+	+	+	+
Heart	0	0	0	0	0	+	0	-	+

Table 3. Tree size variations for different pruning methods (significance level: 0.10)

database	REP	MEP	CVP	OSE	1SE	PEP	CV 0SE	CV 1SE	EBP
Iris	-	-	u	-	o	-	-	o	u
Glass	-	u	u	-	o	u	-	o	u
Led	-	u	u	u	o	o	u	o	u
Hypo	o	u	u	-	o	-	-	o	u
P-gene	o	u	u	-	o	o	-	o	u
Hepatitis	-	u	-	-	o	-	o	o	u
Cleveland	-	-	u	-	o	u	o	o	u
Hungary	o	-	u	o	o	-	-	o	u
Switzerland	-	u	-	-	-	-	-	-	-
Long Beach	-	u	-	-	o	-	-	o	u
Heart	-	-	u	-	o	o	o	o	-

In order to study the effect of pruning on predictive accuracy of decision trees, we compare the error rate of each pruned tree with that of the corresponding (unpruned) trained tree. Tables 2 reports the result of a two-tailed paired t-test with a 0.1 confidence level. A (+) means that the application of the pruning method actually improves, on average, the predictive accuracy of the decision tree, while a (-) indicates a significant decrease in predictive accuracy. When the effect of pruning is neither good nor bad, a 0 is reported. It is easy to see that pruning does not generally decrease predictive accuracy. The only exception is represented by the application of the 1SE rule with both an independent pruning set and cross-validation sets. Moreover, there is no indication that methods exploiting an independent pruning set perform definitely better than the others, as claimed by Mingers (1989).

Another interesting characteristic of pruning methods is their tendency to overprune decision trees. In order to study such a problem, we produced two decision trees for each experiment, called *optimally pruned grown-tree* (OPGT) and *optimally pruned trained-tree* (OPTT), respectively. The former is a grown tree that has been pruned by using the reduced error pruning on the test set. Thus, it is the best pruned tree we could produce from the grown tree because of the property of the reduced error pruning we mentioned in Section 2.1. Similarly, the OPTT is the best tree we could obtain by pruning some branches of the trained tree. Obviously, OPGTs are suitable to compare trees obtained with pruning methods that *do* use an independent pruning set, while OPTTs are more appropriate to compare results of pruning methods that *do not* need a pruning set. Therefore, by comparing the size of trees produced by a pruning method with the size of the corresponding optimal tree, we can have an indication of the tendency of each method. In Table 3, a summary of the two-tailed paired t-tests at a significance level 0.1 is shown. Here, (u) stands for significant underpruning, (o) for significant overpruning, while (-) stands for no significant difference. At a glance, we can immediately conclude that MEP, CVP and EBP tend to underprune, while REP, 1SE and CV-1SE tend to overprune. We would be tempted to conclude that the predictive accuracy is improved whenever a pruning method does not produce trees with significant difference in size from the corresponding optimally pruned tree. However, this is not true for two reasons. First of all, it is not always true that an optimally pruned tree is more accurate than the corresponding grown/trained tree. In other words, pruning may help to simplify trees without improving its predictive accuracy. Secondly, tree size is a global feature that can provide us with an idea of what is happening, but it is not detailed enough to guarantee that only over or underpruning occurred. For instance, if a method overprunes a branch but underprunes another one, then it is actually increasing the error rate with respect to the optimal tree, but not necessarily the size. This problem can be observed with the database Glass and the method CV-0SE. Indeed, in this case there is a decrease in accuracy (see Table 2) but the size of pruned trees is close to the optimal value (see Table 3).

By ideally superimposing Tables 2 and 3 it is also possible to draw some other interesting conclusion. For instance, in some databases, such as Hungary and Heart, overpruning produces better trees than underpruning. This latter surprising result confirms Holte's observation that even simple rules perform well on most commonly used data sets in the machine learning community (1993). In any case, we have also indications that overpruning may have undesirable effects when too extremist, as in the case of the application of the rule 1SE.

4. Conclusions

In this paper, a comparative study of six well-known post-pruning methods has been presented. Each method is briefly described and its weaknesses and strengths are pointed out. Moreover, some results concerning a redesigned experimentation on pruning methods have been reported. By comparing error rates of pruned trees with the corresponding grown/trained trees we have found no clue supporting Mingers' claim that pruning methods exploiting an independent pruning set perform better than the others. On the contrary, our results confirm Holte's observation that even simple rules perform well on most commonly used data sets in the machine learning community. Indeed, in several cases we noticed that a moderate overpruning is better than not pruning at all. Future work should consider a more extensive experimentation in which more databases as well as several selection measures are considered.

Pruning methods have been implemented as an extension of C4.5, a system distributed by Morgan Kaufmann. Only additional source files developed at the Department of Informatics of the University of Bari are available upon request by sending an e-mail at malerbad@vm.csata.it.

References

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*, Belmont, CA: Wadsworth International.

Cestnik, B., & Bratko, I. (1991). On estimating probabilities in tree pruning. In Y. Kodratoff (Ed.), *Machine Learning - EWSL-91*, Lecture Notes in Artificial Intelligence, Berlin: Springer-Verlag.

Cestnik, B., Kononenko, I., & Bratko, I. (1987). ASSISTANT 86: A knowledge-elicitation tool for sophisticated users. In I. Bratko & N. Lavrac (Eds.), *Progress in Machine Learning*. Wilmslow: Sigma Press.

Esposito, F., Malerba, D., & Semeraro, G. (1993). Decision tree pruning as a search in the state space. In P. Brazdil (Ed.), *Machine Learning: ECML-93* , Lecture Notes in Artificial Intelligence, Berlin:Springer-Verlag.

Fisher, D.H., & Chan, P.K. (1990). Statistical guidance in symbolic learning. *Annals of Mathematics and Artificial Intelligence*, 2, pp. 135-148.

Holte R.C. (1993). Very simple classification rules perform well on most commonly used data sets. *Machine Learning*, 11, 1, 63-90.

Hyafil, L., & Rivest, R.L. (1976). Constructing optimal binary decision trees is NP-complete. Information Processing Letters, 5(1), 15-17.

Malerba, D., Semeraro, G., & Esposito, F. (1994). Choosing the best pruned decision tree: A matter of bias. *Proceedings of the Fifth Italian Workshop on Machine Learning*, Parma, Italy, 26-28 September.

Mingers, J. (1987). Expert systems - rule induction with statistical data. *Journal of the Operational Research Society, 38*, 39-47.

Mingers, J. (1989). An empirical comparison of pruning methods for decision tree induction. *Machine Learning, 4*, 227 - 243.

Murphy, P., & Aha, D.W. (1994). UCI repository of machine learning databases [Machine-readable data repository]. Irvine, CA: University of California, Department of Information and Computer Science.

Niblett, T. (1987). Constructing decision trees in noisy domains. In I. Bratko & N. Lavrac (Eds.), *Progress in Machine Learning*. Wilmslow: Sigma Press.

Niblett, T., & Bratko, I. (1986). Learning decision rules in noisy domains. *Proceedings of Expert Systems 86*, Cambridge: Cambridge University Press.

Quinlan, J.R. (1987). Simplifying decision trees. *International Journal of Man-Machine Studies, 27*, 221-234 (also appeared in B. R. Gaines & J. H. Boose (Eds.), *Knowledge Acquisition for Knowledge-Based Systems*, Academic Press, 1988).

Quinlan, J.R. (1993). *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufmann.

Quinlan, J.R., & Rivest, L.R. (1989). Inferring decision trees using the minimum description length principle. *Information and Computation*, 80, 227-248.

Schaffer, C. (1993). Overfitting avoidance as bias. *Machine Learning, 10*, 153-178.

Watkins, C.J.C.H. (1987). Combining cross-validation and search. In I. Bratko & N. Lavrac (Eds.), *Progress in Machine Learning*, Wilmslow: Sigma Press.