Inductive Learning in a Mixed Paradigm Setting*

David B. Skalak and Edwina L. Rissland
Department of Computer and Information Science
University of Massachusetts, Amherst, MA 01003
SKALAK@CS.UMASS.EDU

Abstract

In a *precedent-based* domain one appeals to previous cases to support a solution, decision, explanation, or an argument. Experts typically use care in choosing cases in precedent-based domains, and apply such criteria as case relevance, prototypicality, and importance. In domains where both cases and rules are used, experts use an additional case selection criterion: the generalizations that a particular group of cases support. Domain experts use their knowledge of cases to forge the rules learned from those cases.

In this paper, we explore inductive learning in a "mixed paradigm" setting, where both rule-based and case-based reasoning methods are used. In particular, we consider how the techniques of case-based reasoning in an adversarial, precedent-based domain can be used to aid a decision-tree based classification algorithm for (1) training set selection, (2) branching feature choice, and (3) induction policy preference and deliberate exploitation of inductive bias. We focus on how precedent-based argumentation may inform the selection of training examples used to build classification trees. The resulting decision trees may then be re-expressed as rules and incorporated into the mixed paradigm system. We discuss the heuristic control problems involved in incorporating an inductive learner into CABARET, a mixed paradigm reasoner. Finally, we present an empirical study in a legal domain of the classification trees generated by various training sets constructed by a case-based reasoning module.

Introduction

Precedent-based domains are areas where one appeals to previous cases to support a solution, decision, explanation, or an argument. In such domains, experts typically use care in choosing cases, and apply such criteria as case relevance, prototypicality and importance. In precedent-based domains where both cases and rules are used, experts use an additional selection criterion: the generalizations that a particular group of cases support. Domain experts use their knowledge of cases to forge the rules learned from those cases.

Our focus in this paper is the use of a machine learning technique to induce rules from the case base of a mixed paradigm case-based reasoning/rule-based reasoning ("CBR-RBR") system. Thus, this work is aimed toward a tripartite cooperative "CBR-RBR-ML" system. In this paper, we report on our preliminary steps in this direction.

We discuss several ways that CBR can be used synergistically to aid a cooperating inductive decision-tree based learning algorithm: training set selection, branching feature selection, deliberate bias creation, and specification of induction policy. After a brief introduction to the hybrid architecture of CABARET [Rissland and Skalak, 1989], our mixed paradigm reasoning environment, we investigate the control heuristics that must mediate between the system goals and the learning goals of a mixed paradigm reasoner. We close with a report on our experience in using ID5 [Utgoff, 1988] with the CABARET system. ID5 is an incremental version of the classification algorithm ID3 [Quinlan, 1986], which applies an information-theoretic test to select an attribute upon which to branch at each stage in the creation of a decision tree.

Background: Mixed Paradigm Reasoning Systems

Now that some of the basic aspects of CBR are better understood [Kolodner, 1988; Rissland and King, 1988], some researchers have been investigating how to combine CBR with other reasoning paradigms. In particular, recent projects have combined CBR with reasoning using production rules, approximate rule-based systems, causal models, and utility-based preference analysis [Rissland and Skalak, 1989; Bonissone *et al.*, 1990; Koton, 1988; Oskamp *et al.*, 1989; Goel and Chan-

*This work was supported in part by the National Science Foundation, contract IRI-8908481, the Office of Naval Research under a University Research Initiative Grant, contract N00014-87-K-0238, and a grant from GTE Laboratories, Inc., Waltham, Mass.

drasekaran, 1988; Sycara, 1987]. Our own work has focused on combining precedent-based CBR with rule-based reasoning in adversarial domains. In general, a hybrid approach offers several potential strengths: (1) the ability to overcome blind spots of the individual co-reasoners, (2) the capacity to let the most efficient or somehow most appropriate reasoner handle the tasks for which it is best suited, and (3) the flexibility, power and robustness that arise from representing domain knowledge in a variety of forms.

Domains that have been the subject for such systems include the law and the interpretation of legal rules, corporate mergers and acquisitions, medical and mechanical diagnosis, and labor mediation. Our own research has been primarily in the legal domain, an area we feel presents some of the problems of mixed paradigm systems in a pure and often extreme form:

1. The necessity of understanding the ambiguous terms in which rules are stated.
2. The need to extract the meaning of an individual case in the context of other cases.
3. The modeling of the interaction of a given set of rules and a large mass of cases, each purporting to deal with the same subject matter, each simultaneously limiting and expanding the knowledge of the other.

Applying different knowledge representations and reasoning modes in a hybrid architecture requires sophisticated control mechanisms. One general control approach is to apply knowledge handled initially in one form (*e.g.*, cases) in another form (*e.g.*, rules). This shift in perspective can aid the reasoners' cooperative efforts and afford a common basis to compare differences in the knowledge of diverse knowledge sources. Thus, one of the advantages of a hybrid approach is that domain knowledge can be represented in a variety of ways. A knowledge engineer working in a mixed paradigm environment strives for an appropriate initial mix. Of course, no unique or optimal mix exists and, more importantly, the mix should be amenable to change as the total system evolves. For instance, certain clusters of cases might be inductively generalized into learned rules. Coalescing cases into generalized rules typically occurs during the initial knowledge engineering. (In fact, in pure RBR systems, this process is carried to an extreme in that all case knowledge must be represented in rule form.) However, in most domains, such re-representation of cases as rules (and sometimes *vice versa*) is a continuing process. Thus, concerns about learning — whether done initially by the knowledge engineer or over time by a learning mechanism — are paramount in mixed paradigm systems.

What distinguishes a mixed paradigm CBR-RBR system from a pure RBR system, for instance, is the continuing availability of the cases for learning. This feature benefits the application of learning techniques: inductive learning methods can be applied at any time and training instances need not be manufactured. However, one usually does not want to have the system learn at every turn, and, furthermore, one does not usually feel every case is equally important to the learning. Control is the issue, both in the basis and the timing of learning.

Using CBR in the Service of Inductive Learning

In this section we address several areas in which CBR can be used to aid inductive learning:

1. Training set selection
2. Feature selection
3. Creation of deliberate bias and selection of induction policy.

In a final section, we discuss our empirical results as to the first of these, training set selection.

Using CBR for Training Set Selection

In many domains it may be inefficient, impracticable or inadvisable to include all the members of a case base in the training set to be put to an inductive learning algorithm. The sheer number of available cases can be prohibitive. The incremental benefit of including many cases of the same ilk may be small. Human experts themselves are selective in the use of cases. These observations are borne out in our domain, the law, where the case base is large, many cases are similar to other cases, and neither judges nor attorneys use all the cases available to them to formulate rules. These observations point to a sampling problem: how to select a proper subset of a case base that accurately reflects the entirety. In our implemented systems, statistical sampling has not been possible in view of the "small" number of cases represented. But the question remains as to the means of selecting a set of cases that can serve as a surrogate for the entire case base.

The various classifications of cases recognized in CBR suggest an attack on this sampling problem: use CBR-generated classes to construct surrogate training sets. In CBR, cases may be ranked by their similarity to a given cases (*e.g.*, most-on-point), their usefulness in a given situation (*e.g.*, most salient), their precedential capacity (*e.g.*, "best case to cite", counterexample), and on finer distinctions (*e.g.*, counterexamples may be anomalous, extreme, or "incursive"). Recognition of case taxonomies of these sorts contrasts with the usual treatment of instances in inductive learning, where are all examples are treated equally, without regard to the various roles they play, aside from their classification, say as positive or negative instances. Notable exceptions to this democratic treatment include examples that have the power to drastically reduce a search space, such as "near misses," [Winston, 1975; Buchanan *et al.*, 1990] and, under the Candidate Elimination Algorithm [Mitchell, 1978], examples that can halve a version space.

If the thesis is borne out that certain subclasses can stand as surrogates for the entire case base, then the specially selected training sets may improve (1) the efficiency of the learning (*e.g.*, a given level of performance can be achieved with fewer cases) and (2) its quality (*e.g.*, a given number of special cases can lead to better performance than an equal number of non-special cases). In a mixed paradigm setting, where such categorization of cases is done by the CBR module during the normal course of problem solving, little or no additional overhead is spent in generating the training sets we propose for consideration. It remains for further experiments to determine whether outside a mixed paradigm environment any benefits from using CBR to select a useful subset of training instances outweigh the cost of the CBR analysis.

To test our hypothesis, we have performed a series of experiments on rule generation from sets of cases generated by a case-based reasoner. The categories we have used to form training sets include:

(1) **"Most-On-Point" cases**;
(2) **"Best" cases**;
(3) **"Near Miss" cases**;
(4) **"Trumping" cases**; and
(5) **"Conflict" cases.**

Empirical results using these five training sets are presented below, after a description of the construction of these case sets.

Most-On-Point Cases Training Set

To construct the most-on-point training set, we first performed a CBR analysis of each case in the case base, regarding each case as a problem situation presented *de novo* to the system. The most-on-point cases for a problem case are defined as those maximal in a partial ordering of cases according to similarity with the problem case. To form a training set composed of most-on-point cases, we take the union (over all the cases in the case base) of the cases that have appeared as a most similar (most-on-point) case for any problem case in the case base.

Specifically, the partial ordering by similarity used in these experiments is created as follows. The universe of factors or "dimensions" used to determine similarity is first restricted to those dimensions that are applicable to the problem case. The similarity relation on cases is then defined in terms only of the dimensions from this new, restricted universe. Specifically, a case B is more similar to the problem case than Case A if case A's applicable dimensions (taken from the restricted universe) are a subset of case B's applicable dimensions. This similarity relation defines a partial order on the case base. It can be represented as a rooted directed graph, which for historical reasons is called a "claim lattice" [Ashley, 1990]. The most-on-point cases are the maximal elements in this rooted graph, and are found in the nodes closest to the root node, which contains the problem case.

Best Cases Training Set

To form the training set consisting of "best" cases, we take the union (over all the cases in the case base) of the cases that have appeared as a best case to cite in support of one side or the other in a problem case.

Given a problem case and a point of view (say, plaintiff's or defendant's), the best cases to cite are defined as most-on-point cases that have been decided for that viewpoint and that share strength with the problem situation along at least one dimension.

Near Miss Cases Training Set

To form the training set consisting of near miss cases, we take the union (over all the cases in the case base) of the cases that have appeared as a "near miss" case for another case. For the purpose of these experiments, a near miss case was defined as a case that appeared as a most-on-point case in an "extended" claim lattice but not as a most-on-point case in the usual claim lattice.

The extended claim lattice is constructed analogously to the usual lattice, but instead of restricting the relevant universe of dimensions to those applicable to the problem case, the universe of dimensions is taken as the union of the applicable <u>and</u> the "near miss" dimensions. A near miss dimension is a dimension all of whose prerequisites are satisfied, except a single, previously designated prerequisite. Dimensional prerequisites ensure that enough background information is available in the case for it to make sense to analyze a case with respect to that factor.

Thus a near miss case is a case that would be compelling if some additional factor(s) were present in the problem case. Near misses are often probative because they embody situations that are nearly most-on-point but nonetheless exhibit well-focused differences with a problem situation.

Trumping Cases Training Set

To form this training set, we take the union (over all the cases in the case base) of pairs of cases that have appeared as a "trumping" and a "trumped" case for a problem case, where the trumping and trumped cases are the best cases that can be cited for their respective viewpoints.

One case is said to "trump" another if (1) it is strictly more-on-point than the other: that is, the trumped case applicable factors are a proper subset of those of the trumping case, and (2) it has been decided oppositely.

Conflict Cases Training Set

To form this training set, we take the union (over all the cases in the case base) of groups of cases that have appeared in "conflict nodes" in the claim lattice. They are cases of equal similarity but of opposite classification.

A conflict node is a node containing cases that have been classified differently (*e.g.*, held for plaintiff, held for defendant) despite the fact that each is equally sim-

ilar to a given case. Such cases share the same applicable factors at the dimensional level (at least where the universe of applicable dimensions is restricted to those applicable to a given case.)

The choice of these sets as candidates for experimentation is supported by various intuitions. To find an effective training set requires a selection of the most representative examples or those packing the most punch. If one thinks of the cases as embedded in a many-dimensioned space, where the dimensions are the relevant domain factors or other features used in the case representation, relevant cases cluster in the subspace occupied by the current problem case. The rationale for using training sets of most-on-point and best cases is that by selecting only these cases one selects the clusters for training, and avoids some of the outliers, cases that exist in a different subspace from the others.

The examination of a training set incorporating near misses relies on previous work that recognizes the capacity of the near miss to decrease the size of a search space [Winston, 1975; Buchanan *et al.*, 1990]. The intuition here is that in order to delimit a concept, examples immediately inside ("near hits") and immediately outside (near misses) should be included. The near-miss training set incorporates cases that would be among the most relevant, if at least one more feature were known about the case. One of the benefits of a near-miss case is that it isolates one or a few features and "controls" for their presence.

The trumping and conflict case training sets reflect the intuition that induction may be enhanced by including groups of quite similar cases of positive and negative classification. Further, examples that highlight problems in classification can be telling. In a trumping/trumped pair, the problem is that the presence of one or more additional factors can toggle the outcome of a case (cf. [Subramanian and Feigenbaum, 1986]). Including conflict cases exploits the similar intuitive idea that some differences in magnitude of the applicable factors must be responsible for distinctions in classification. These training sets are particularly relevant to the credit assignment problem as it applies to factor analysis: finding differences that make a difference.

Of course, alternative training sets exist, and there are reasons why the suggested training sets may be inadequate. For instance, the most-on-point training set has been described as eliminating outlying cases, but these outliers can provide important exceptions to a general rule. A couple of responses to this particular observation are possible. One possible response is that in a mixed paradigm setting, other CBR methods, such as "distinguishing," can be invoked to explain these outlying cases. Distinguishing identifies, highlights and manipulates relevant differences between cases. Distinguishing is a useful technique, both in the law, and in CBR, where difference links are frequently maintained to index a case base [Ashley, 1987; Bareiss *et al.*, 1987; Falkenhainer, 1988]. A case that can be distinguished easily from a group of other cases need not be used to support rule induction. Our task here is merely to come up with domain rules of thumb that reflect the classification knowledge inherent in a case base. From this perspective, exceptions are not as bothersome, since the usual functions inherent in a CBR module can compensate for shortcomings in heuristic rules.

Using CBR for Feature Selection

One of the major drawbacks of a similarity-based inductive learning system is the lack of domain knowledge available to the learner. Frequently, features are treated syntactically, without supporting semantic knowledge. In ID5 for instance, the attributes chosen for branching depend ultimately on the frequency of their associated values, without regard to the importance of the attribute to that domain. CBR methods and the domain knowledge implicit in the cases can be used to overcome this drawback.

The basic idea behind using CBR to aid the selection of features for decision tree branching is that CBR elicits domain knowledge inherent in the cases that is ignored by the fundamentally statistical classification scheme used by ID3 family of algorithms. CBR analysis can provide some of the domain theory through its identification of important factors. For instance, features that are used as distinguishing features in CBR may be good candidates for branching. As we have mentioned, distinguishing is used in precedent-based argumentation to show why a distinction makes a difference in the outcome of a case. Typically one shows that factors that have appeared in one case but not in another account for the difference in their outcomes. An advocate distinguishes cases cited by an opponent (or anticipated by the advocate that the opponent will cite), to show that the result the opponent seeks is dependent on additional factors not present in the current case, or on features that the current case possesses that are absent in the cited case. Distinguishing factors are exactly the sort of features one should consider at branches in a decision tree.

The use of a component that takes advantage of CBR-provided domain knowledge as well as of information-theoretic methods may provide a basis for enhanced classification techniques. Mingres [1989] has shown that using a variety information-theoretic tests alone to select branching features results in no more accurate classification than does a random selection of branching attributes. Mingres's results suggest a need to bring domain knowledge into the choice of branching attributes. CBR can elicit such knowledge inherent in the training cases themselves.

Using CBR for Other Learning Tasks

CBR has additional potential to aid inductive learning. First, CBR can be used to take advantage of the bias inherent in inductive generalization. In an adversarial domain such as the law, a case-based reasoner can deliberately use bias advantageously by selecting (only) those cases that are most favorable or by describing cases in terms that are most favorable. Favorable precedents can then be used in the training set to create a rule favorable to that party. In fact, any domain can be considered "adversarial" where one can assume differing viewpoints or hypothetically assess the strength of the argument for one outcome or another.

Second, CBR can be used to formulate induction policy. Every case that would alter an induced rule need not drive incremental generalization or specialization. CBR's capacity to tag examples by subcategory, such as "anomalous" or "distinguishable," can be used by an inductive learner to avoid generalization on the basis of an unrepresentative example.

Inductive Learning and CABARET

In this section, we describe our preliminary investigation of the addition of a similarity-based learning component to an existing hybrid architecture.

A Mixed Paradigm Reasoning Program

CABARET ("CAse-BAsed REasoning Tool") provides an environment to explore mixed paradigm CBR-RBR reasoning [Rissland and Skalak, 1989]. The current architecture includes six major knowledge sources: (1) a case-based reasoner modelled after the architecture of HYPO [Ashley, 1990]; (2) a production system, permitting forward and backward chaining; (3) an agenda-based control module, which, on the basis on the current state of problem solving, places prioritized tasks on an agenda to be performed by either the CBR or RBR module; (4) a CBR monitor, which monitors the progress of the case-based reasoner and extracts information relevant to the control module; (5) an RBR monitor, which performs the analogous task for the rule-based reasoner; and (6) a corpus of domain knowledge, in such form as hierarchical nets, which is profitably expressed independently of the other knowledge sources because it cannot be expressed most suitably as rules or cases or simply because both RBR and CBR components make use of it.

CABARET's CBR component is somewhat different from a stand-alone CBR reasoner like HYPO [Ashley, 1990], in that it is augmented by the ability to distinguish and analogize cases on the basis of their behavior under a rule-set as well as on the basis of "dimensions" internal to the CBR module.

Heuristics to Invoke Learning

Ongoing experiments in CABARET involve the addition of one more knowledge source: the ID5 learning algorithm [Utgoff, 1988]. To trigger this additional knowledge source, several control heuristics based on broadening and narrowing of rules have been developed. Rule "broadening" is applied when a domain rule with a given conclusion fails to fire, but the user has assumed a point of view that wants to establish the rule consequent. Narrowing is used when one wants to prevent a rule from firing.

In CBR, broadening is accomplished in a variety of ways. For instance, one can show that a rule-antecedent is not necessary, expand the scope of a polymorphic ("open-textured") antecedent, or construct analogies with cases where the rule has in fact fired, and so forth.

Inductive learning can be used to support such methods, such as expanding the scope of important rule terms. Since in the law, as in some other domains, cases record whether certain terms in the statute are satisfied by the case at hand, these determinations can classify cases for use as training examples to induce the meaning of the term. In our experiments, we have used a court's determinations about particular terms in a legal statute (governing the home office deduction under Federal income tax law), to have CABARET and ID5 induce rules regarding these important statutory predicates.

One can also use inductive techniques to create a rule that competes with existing rules. Then, if the current problem case satisfies the induced rule, one may argue that the consequent is established on the basis of precedent. In the law, this sort of use of rules induced from common law cases (a part of the "blackletter" law) plays a major role in appellate argument.

The bias inherent in an adversarial domain also permits creative use of an inductive learning algorithm to manipulate bias deliberately through selection of the cases that go into the training set, as noted above. In addition, pointers between cases provided by the cases themselves can be used in some domains in conjunction with CBR to select "biased" training sets. Legal cases often cite a variety of other cases. To broaden a rule upon the failure of other approaches to rule-expansion, CABARET may include heuristics for the studied selection of a training set that favors one side or another.

Similar techniques may be used for narrowing, in which one discredits a rule or limits its scope. CABARET attempts to narrow a rule when it has fired but the vantage point of the user is against its application.

An Experiment in Federal Taxation

In order to gain some experience with inductive learning in a mixed paradigm setting, to test some of the proposed uses of CBR for inductive learning and to evaluate the domain rules induced by ID5, we have performed a series of experiments with a case base of actual Federal income tax cases dealing with the so-

called "home office deduction." Under certain stringent conditions set forth in §280A of the Internal Revenue Code, relating to the regular and exclusive use of the space as an office and other requirements, a taxpayer may take a deduction against gross income for expenses incurred in connection with the home office, such as a proportion of the rent, utilities, and equipment[1]. The case base used for these experiments consisted of 25 litigated cases drawn mostly from Federal Tax Court and the Federal Courts of Appeal.

In order to test the efficacy of each of the training sets discussed above, we have run ID5 at each of two levels of representation produced by the case-based analysis of a case:

1. the *factual-predicate* ("F.P.", also called the *interpretation frame* level), of features and values derived by the CBR component of CABARET from the base-level features (there are 20 factual predicates);

2. the *dimension* ("Dim.") level, with values t or nil, according to whether or not the dimension was applicable in the case-based analysis of that case (14 dimensions).

Cases at both levels of representation were classified as + or −, according to whether the taxpayer received the home office deduction in the actual legal case. Since no acknowledged "correct" theory of the home office deduction domain is available, the trees derived from each training set were compared to the classification tree derived from the entire case base (the "All Cases" training set). The tree derived from the entire case base served as a surrogate for a target concept definition. All the cases were then classified by the resulting classification trees to test each tree's classification accuracy. If a case was classified incorrectly by the tree, the case was counted as an error below. If a case was unable to be classified by a decision tree, it was recorded as "No Class" in the tables following. To serve as a control for each CBR training set of a given size, a collection of training sets of that size was generated, containing a random selection of cases from the case base[2]. The average number of errors and unclassified cases were then computed for these random collections. For the random sets of size 19 and 8, 20 trees generated from random cases were constructed; for the random sets of size 3, 100 trees were used to compute the average errors[3]. We present empirical results and summarize the performance of each training set.

[1] Nothing herein may be construed as legal advice, for which the reader should consult his own tax practitioner.

[2] Except that no collection of precisely 18 random cases was generated.

[3] More variation was observed in the classification accuracy of the training sets of size 3; hence the larger sample.

All Cases, Most-on-point & Best Cases Training Sets				
Training Set	*Level*	*Cases*	*Errors*	*No Class*
All Cases	Dim.	25	0	5
Most-on-point	Dim.	19	0	5
Best Cases	Dim.	18	1	5
Random	Dim.	19	1.25	3.05
All Cases	F.P.	25	0	0
Most-on-point	F.P.	19	1	2
Best Cases	F.P.	18	2	2
Random	F.P.	19	0.85	1.85

Contrary to intuition, the random training sets performed as well as or better than the most-on-point and best cases training sets. On the other hand, the size of the most-on-point and best cases sets was nearly that of the entire case base, and one might expect similar performance from any training collection of this relative size. It remains for experiments with other case bases to determine if the most-on-point and best cases training sets (as we have constructed them) are often almost as large as the case base itself.

We note that the dimensions did not completely discriminate the case base. This observation underlines the necessity of comparing the magnitudes of relevant factors, and not just their presence or absence. The comparison of values along dimensions is an integral feature of the argument generation module of HYPO [Ashley, 1990].

Near Miss Cases Training Sets				
Training Set	*Level*	*Cases*	*Errors*	*No Class*
Near Misses	Dim.	8	4	0
Random	Dim.	8	5.25	0.80
Near Misses	F.P.	8	3	3
Random	F.P.	8	3.75	3.55

At the dimension level, near misses yielded slightly fewer errors and unclassified cases than random case sets. These results may tend to confirm the efficiency of including near misses in a training set. At the factual predicate level, performance was approximately equivalent.

Trumping Cases Training Sets				
Training Set	*Level*	*Cases*	*Errors*	*No Class*
Trumping	Dim.	3	4	0
Random	Dim.	3	8.27	0.35
Trumping	F.P.	3	0	19
Random	F.P.	3	7.70	3.36

In this domain, it is rare that one side can cite a case that is strictly better than its opponent's best case. Only three cases that were members of a trumping-trumped pair of cases were found. Nevertheless, even using this minuscule training set, only 4 errors were produced at the dimension level. Random collections of three cases generated trees that yielded over 8 errors.

In part, the superior performance of the trumping cases may be attributed to their providing pairs of cases of opposing classification that possess highlighted differences. Trumping case pairs can therefore illuminate the credit assignment problem. They implicitly isolate factors whose presence or absence makes a critical difference, and thus can be assigned the blame or credit for a classification.

Conflict Cases Training Sets				
Training Set	*Level*	*Cases*	*Errors*	*No Class*
Conflicts	Dim.	4.8	3.73	3.73
Random	Dim.	4.8	6.76	0.41
Conflicts	F.P.	4.8	6.00	8.30
Random	F.P.	4.8	4.68	4.43

Ten sets of training examples, each containing only cases derived from a conflict node were generated for this experiment. All conflict nodes for all cases were generated, and were then filtered to eliminate conflict nodes that contained too few cases to present two sufficiently strong opposing viewpoints, or too many to provide a focused conflict[4]. Average figures are used for the number of cases, errors and unclassified cases in the table above.

At the factual predicate level, random training sets performed better than the conflicts training sets. However, at the dimension level, while the total number of errors and unclassified cases were the same for the conflicts and random sets, the distribution of these errors was notably different. Random training yielded a number of mistakes in classification, but almost no cases that could not be classified. The conflicts cases resulted in a tree that recognized that alternative arguments were available: a relatively large number of cases were unclassifiable, but a more reliable classification resulted when made.

Branching Feature Selection

While we have not yet performed a formal study of the utility of the home office classification trees, several informal observations may be noteworthy. We noted, for example, significant agreement in the attributes that appear in the top several levels of many of the decision trees generated in these experiments. These factors include the relative time that the taxpayer spent in the home office and any other office, whether the home office was in a separate structure, whether the home office was necessary to perform the taxpayer's assigned responsibilities, whether it was used occasionally or regularly, and how many hours per week the home office was used. As discussed in more detail elsewhere [Rissland and Skalak, 1990; Skalak, 1989], these factors are arguably among the most crucial to one's home office deduction claim.

[4]Specifically, nodes containing fewer than 4 or more than 8 cases were eliminated.

Finally, the shape of the following tree based on a most-on-point training set is representative of the unbalanced trees observed throughout these experiments.

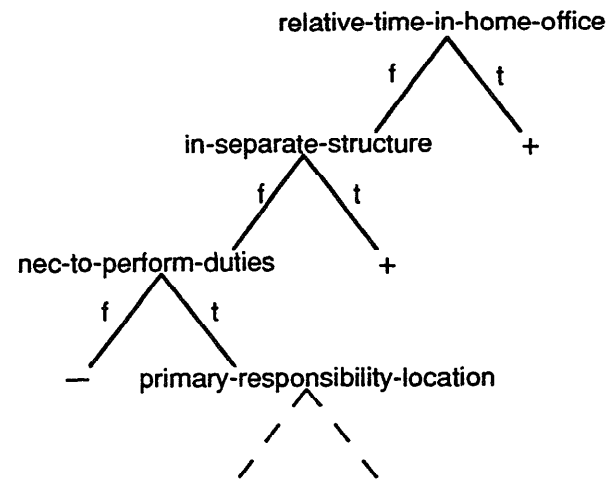

Figure 1: Classification tree generated by ID5 using Most-On-Point Training Set, Dimension level (pruned to show first several branching features)

Conclusions

Just as experts can use cases to create a new rule or refine an old one, cases can be so used by a mixed paradigm system with a learning component. We have shown how the incorporation of an inductive learning component into a mixed paradigm CBR-RBR system provides many opportunities for fruitful synergy. Generally, a learning capability can help the system evolve. Specifically, CBR can aid an inductive learning algorithm through the informed selection of training instances and branching attributes, the control over induction policy, and the deliberate exploitation of inductive bias.

Empirical results are mixed as to the utility of CBR to select training sets. Relatively large training sets suggested by CBR (most-on-point, best cases) did no better than a random selection of cases at each of two representation levels. Smaller training sets that explicitly incorporated similar training instances of opposite classifications (trumping, conflict, near miss) yielded fewer classification errors at a domain factor level.

Acknowledgments

We are grateful to Paul Utgoff for supplying the code for ID5 and to Dan Suthers for providing many of the software tools underlying the system. We thank also Elizabeth Gene and Tony Reish for their assistance in coding CABARET.

References

[Ashley, 1987] Kevin D. Ashley. Distinguishing – A Reasoner's Wedge. In *Ninth Annual Cognitive Science Society Conference*, University of Washington, Seattle, 1987.

[Ashley, 1990] Kevin D. Ashley. *Modelling Legal Argument: Reasoning with Cases and Hypotheticals.* M.I.T. Press, Cambridge, 1990. In press.

[Bareiss *et al.*, 1987] E. Ray Bareiss, Bruce W. Porter, and Craig C. Wier. Protos: An Exemplar-Based Learning Apprentice. In *Proceedings Fourth International Workshop on Machine Learning*, pages 12–23, University of California at Irvine, June 1987.

[Bonissone *et al.*, 1990] Piero Bonissone, Lauren Blau, and Saad Ayub. Leveraging the Integration of Approximate Reasoning Systems. In *Proceedings of the AAAI Symposium on Case-Based Reasoning–1990*, Palo Alto, March 1990.

[Buchanan *et al.*, 1990] B. G. Buchanan, E.L. Rissland, P. R. Rosenbloom, H. T. Ng, and J. Sullivan. Intelligent Example Selection for Learning Systems, 1990. In press.

[Falkenhainer, 1988] Brian Falkenhainer. The Utility of Difference-Based Reasoning. In *Proceedings AAAI–88*, St. Paul, August 1988. American Association for Artificial Intelligence.

[Goel and Chandrasekaran, 1988] Ashok Goel and B. Chandrasekaran. Integrating Model-Based Reasoning and Case-Based Reasoning. In *Proceedings of the AAAI Workshop on AI in Design.* American Association for Artificial Intelligence, August 1988. St. Paul.

[Kolodner, 1988] Janet L. Kolodner, editor. *Proceedings of the DARPA Case-Based Reasoning Workshop–1988*, Clearwater Beach, FL, May 1988.

[Koton, 1988] Phyllis A. Koton. *Using Experience in Learning and Problem Solving.* PhD thesis, Department of Electrical Engineering and Computer Science, MIT, 1988.

[Mingres, 1989] John Mingres. An Empirical Comparison of Selection Measures for Decision-Tree Algorithms. *Machine Learning Journal*, 3:319–342, 1989.

[Mitchell, 1978] Tom Mitchell. *Version Spaces: An Approach to Concept Learning.* PhD thesis, Computer Science Dept., Stanford University, 1978. Rep. No. STAN-CS-78-711.

[Oskamp *et al.*, 1989] A. Oskamp, R.F. Walker, J.A. Schrickx, and P.H. van den Berg. PROLEX, Divide and Rule: A Legal Application. In *Proceedings of the Second International Conference on Artificial Intelligence and Law*, Vancouver, B.C., June 1989.

[Quinlan, 1986] J.R. Quinlan. Induction of Decision Trees. In *Machine Learning 1(1)*, pages 81 – 106. Kluwer Academic, Boston, 1986.

[Rissland and King, 1988] Edwina L. Rissland and James A. King, editors. *Proceedings of the AAAI–88 Case-Based Reasoning Workshop*, St. Paul, August 1988. American Association for Artificial Intelligence.

[Rissland and Skalak, 1989] Edwina L. Rissland and David B. Skalak. Combining Case-Based and Rule-Based Reasoning: A Heuristic Approach. In *Proceedings IJCAI–89*, Detroit, August 1989. International Joint Conferences on Artificial Intelligence, Inc.

[Rissland and Skalak, 1990] Edwina L. Rissland and David B. Skalak. Opportunities for Learning With and Within CBR. In *Proceedings of the AAAI Symposium on Case-Based Reasoning–1990*, Palo Alto, March 1990.

[Skalak, 1989] David B. Skalak. An Experiment with Decision Trees and Legal Cases. Internal Memorandum, Department of Computer and Information Science, University of Massachusetts, Amherst, MA, 1989.

[Subramanian and Feigenbaum, 1986] Devika Subramanian and Joan Feigenbaum. Factorization in Experiment Generation. In *Proceedings AAAI–86*, Philadelphia, August 1986. American Association for Artificial Intelligence.

[Sycara, 1987] Katia Sycara. *Resolving Adversarial Conflicts: An Approach Integrating Case-Based and Analytic Methods.* PhD thesis, School of Information and Computer Science, Georgia Institute of Technology, 1987.

[Utgoff, 1988] Paul E. Utgoff. ID5: An Incremental ID3. In *Proceedings of the Fifth International Conference on Machine Learning*, Ann Arbor, June 1988.

[Winston, 1975] Patrick H. Winston. Learning Structural Descriptions from Examples. In Winston, P.H., editor, *The Psychology of Computer Vision*, pages 157 – 209. McGraw-Hill, New York, 1975. Reprint of Rep. No. TR-231, AI Laboratory, MIT.