ID2-of-3: Constructive Induction of M-of-N Concepts for Discriminators in Decision Trees

Patrick M. Murphy
Dept. of Info. & Computer Science
University of California, Irvine, CA 92717
pmurphy@ics.uci.edu

Michael J. Pazzani
Dept. of Info. & Computer Science
University of California, Irvine, CA 92717
pazzani@ics.uci.edu

Abstract

We discuss an approach to constructing composite features during the induction of decision trees. The composite features correspond to *m-of-n* concepts. There are three goals of this research. First, we explore a family of greedy methods for building *m-of-n* concepts (one of which, GS, is described in this paper). Second, we show how these concepts can be formed as internal nodes of decision trees, serving as a bias to the learner. Finally, we evaluate the method on several artificially generated and naturally occurring data sets to determine the effects of this bias.

1 INTRODUCTION

In this paper, we discuss an approach to constructing composite features during the induction of decision trees (Quinlan, 1986). Pagallo and Haussler (1990) describe the FRINGE algorithm that analyzes a decision tree and creates new features that are conjunctions of existing features. After the new features are constructed, the induction algorithm is run again and may incorporate the conjunctive features in the test of a single node in the decision tree. These conjunctive tests avoid the need to duplicate subtrees in a decision tree.

Our work on the construction of composite features differs in two ways from FRINGE:

1. New features are constructed while the decision tree is created. This avoids the need to re-run the induction algorithm after the constructive induction process.

2. The new features are *m-of-n* concepts (also known as Boolean threshold functions) rather than conjunctive concepts. We will use the notation *m-of*-$(F1, ..., Fn)$ to indicate an *m-of-n* concept. For example, 2-*of*-(A, B, C, D) is logically equivalent to $AB + AC + AD + BC + BD + CD$.

Although *m-of-n* concepts can be expressed in terms of conjunctions and disjunctions, they are not easily represented as decision trees. Adding conjunctive tests to nodes mitigates the problem of replicated subtrees, but does not completely solve it. For example, for a single 4-*of*-8 concept, there are 70 conjunctive terms.

We are interested in creating composite *m-of-n* features for several reasons. First, *m-of-n* concepts more closely resemble "fuzzy categories" with graded structure (Barsalou, 1985; Smith & Medin, 1981). Second, there is some evidence that this bias helps in the acquisition of naturally occurring concepts (Spackman, 1988). For example, a successful medical expert system makes use of "criteria tables" that are essentially *m-of-n* concepts (Kingsland, 1985).

Our motivation is somewhat similar to that of Utgoff (1988) in developing perceptron trees. In particular, the terms constructed to serve as tests at nodes in the decision tree serve as a representational bias for the learner. However, *m-of-n* concepts are less expressive than perceptrons, and provide further constraints on the learning. Furthermore, the construction of these terms does not require the step of determining that the data is not linearly separable.

Note that we are also interested in efficient means of approximating *m-of-n* concepts when there may be noise in the training data. Pitt and Valiant (1988) show that it is NP-hard to determine whether there exists a *m-of-n* concept that is consistent with a set of examples. Therefore, the algorithm for creating new *m-of-n* features must not rely on such a test. Instead, we merely determine whether a given *m-of-n* concept provides gain according to an information-based evaluation function (Quinlan, 1986). Furthermore, the feature construction algorithm makes use of operators to find a set of *m-of-n* concepts related to a given *m-of-n* concept and performs hill-climbing search to find the *m-of-n* concept in this set whose gain is greatest.

In the next section, we describe two operators for modifying the current *m-of-n* hypothesis and describe an approach to choosing an initial hypothesis. Next, we describe how the *m-of-n* feature creation algorithm is embedded in a decision tree learning program called ID2-*of*-3 and compare its performance with and without the feature creation mechanism on artificially generated data sets. Finally, we give results from running the algorithms on four naturally occurring data sets.

2 DEFINITIONS AND CONVENTIONS

An *m-of-n* concept classifies an example as positive if at least m of the feature-values in the set n are present in the example. Note that typically, examples have a dimensionality d that is greater than n. Therefore, the learner must determine which n of a total d features may be associated with positive examples. Throughout this paper, where unambiguous, n will refer to either the set of n "relevant" feature-values or the cardinality (number of distinct features) of this set. We will call the conjunction of these n features the prototypic region of the *m-of-n* concept. The learner must also determine how far from the prototypic region an example may deviate, while still being classified as positive. We will call this value (i.e., $n - m$) the relative threshold of the *m-of-n* concept. Formally, *m-of-n* concepts are characterized as all examples within hamming distance $n - m$ of some example in the prototypic region.

We will define a number of operators that either generalize or specialize a given *m-of-n* concept. For example, such a concept may be generalized by including an additional feature-value in the set n (and not increasing m). Adding a feature-value to n may include adding another value to a feature already present in n. One way to specialize an *m-of-n* hypothesis is to increase m.

Decision trees constructed by ID-2-*of*-3 are like those built by ID3 (Quinlan, 1986) except that they contain *m-of-n* hypotheses as discriminators at internal nodes.

3 CONSTRUCTING *M-of-N* CONCEPTS

An *m-of-n* concept construction algorithm, GS, is described in this section that applies operators during hill-climbing search to successively refine partial *m-of-n* hypotheses (Table 1). GS, chooses as its initial hypothesis the feature-value that best splits the data, according to the information based heuristic entropy function from ID3. The chosen feature-value, F, becomes the 1-*of*-(F) initial hypothesis that operators are then applied to. For a description and comparison of other approaches for constructing *m-of-n* hypotheses see (Murphy & Pazzani, 1991).

Table 1. *M-of-N* Constructive Induction Algorithm

```
function generate_m-of-n(Examples)
    Best_Concept := initial-concept(Examples);
    Best_Cost := eval-concept(Examples,Best_Concept);
    repeat
        Concept := Best_Concept;
        Cost := Best_Cost;
        for all Op_Instantiation(Concept)
            Temp_Concept := Op_Instantiation(Concept);
            Temp_Cost := eval-cost(Examples,Temp_Concept);
            if Temp_Cost < Best_Cost then
                Best_Cost := Temp_Cost;
                Best_Concept := Temp_Concept;
    until BestCost ≥ Cost
    return Concept;
```

3.1 GS

GS conceptually builds an *m-of-n* concept from the simplest to the more complex. It starts with a simple *m-of-n* concept, where the majority of the features in the feature space are considered irrelevant, and applies operators that add relevant feature-values to the prototypic region and optionally increment the threshold. Specifically, the two operators used in GS are:

- *m-of-n* + 1 : Keep threshold constant, add feature-value.
  (e.g., 2-*of*-($\overline{A}, B, D$) $\Rightarrow$ 2-*of*-($\overline{A}, B, D, \overline{C}$))
- $m+1$-*of*-$n+1$: Increment threshold, add feature-value.
  (e.g., 2-*of*-($\overline{A}, B, D$) $\Rightarrow$ 3-*of*-($\overline{A}, B, D, \overline{C}$))

Operator *m-of-n* $+ 1$ generalizes the hypothesis by either increasing the relative threshold or by generalizing the prototypic region. Before application of the operator, the hypothesis covered all points within a hamming distance of $n - m$ of the prototypic region. After the operator application, if the feature added is already present in n, the prototypic region is generalized and n remains constant, otherwise the prototypic region is specialized as n is incremented, and the maximum hamming distance covered increases by one.

Operator $m + 1$-*of*-$n + 1$ specializes the hypothesis. If the feature added is already present in n, the prototypic region is generalized and the relative threshold is decreased. Otherwise the prototypic region is specialized with the relative threshold remaining constant.

Note, if the feature-value added, $f = v$, by either of these operators adds the only remaining value, v, for f not already in n, the feature f becomes irrelevant to the concept.

GS has the potential of generating any *m-of-n* concept in the hypothesis space. The number of possible hypotheses considered by GS is $O(kd^2)$, where k

is an upper bound on the number of values for any feature and d is the dimensionality of the hypothesis space. Since there are $O(d3^d)$ concepts in the hypothesis space of Boolean valued features (Hampson & Volper, 1986), each operator application, in such a space, discards $O(3^d/d)$ possible hypotheses. Search depth is no greater than kd. The time required to generate a single *m-of-n* concept is $O(ekd^3)$, where e is the number of training examples.

4 EMBEDDING *M-of-N* HYPOTHESIS

M-of-N hypotheses by themselves suffer from the same representational limits as linear threshold units – they can only represent linearly separable concepts. To overcome this weakness, *m-of-n* hypotheses generated by the previous algorithms are used to create new terms that serve as nodes in decision trees (see Table 2).

Since the space of *m-of-n* concepts is a superset of the space of single attribute discriminations, decision trees constructed in this manner are complete (Utgoff, 1988) in that they can represent any subset of the instance space. Since decision trees also have this property, the intent is not to make decision trees more expressive, but rather to bias decision trees to make polythetic (Fisher, 1987) discriminations. Furthermore, since each node of an *m-of-n* decision tree has more representation power than that of an ID3 decision tree node, *m-of-n* decision trees have the potential of being much smaller than ID3's trees when concepts conform to combinations of *m-of-n* decisions.

Table 2. ID2-*of*-3 Decision Tree Algorithm

```
function generate-tree(Examples)
    if examples-discriminated(Examples) then
        return example-class(Examples);
    else
        Discriminator := generate_m-of-n(Examples);
        (Positive_Examples,Negative_Examples) :=
              split-examples(Discriminator,Examples);
        return tree(Discriminator,
                      generate-tree(Positive_Examples),
                      generate-tree(Negative_Examples));
```

5 EXPERIMENTS

To evaluate the effects of constructing *m-of-n* terms, the accuracy of the decision trees produced by the above algorithms were compared for both artificial and real domains. We also ran a version of ID3 (provided by Ray Mooney) on the same data. Since we have not yet implemented a pruning algorithm for ID2-*of*-3, we did not use any of the pruning algorithms for ID3.

5.1 ARTIFICIAL DATA

We ran an exhaustive series of experiments comparing ID3, and ID2-*of*-3 augmented with GS. The goal of the experiments was to gain an understanding of the properties of GS in constructing *m-of-n* terms.

The artificial concepts were formed from a Boolean feature space with two classes (positive and negative), distinguished by some form of *m-of-n* target concept. For each concept, we recorded the accuracy as a function of the training set size. Target concepts ranged from simple *m-of-n* concepts, consisting of a singleton *m-of-n* concept, to more complex *m-of-n* concepts, e.g., conjuncts and disjuncts of singleton *m-of-n* concepts. The singleton *m-of-n* concepts ranged from conjuncts, (*n-of-n*), to disjuncts (1-*of-n*). We also tested the effects of irrelevant features on these concepts. A listing of some of the experiments is provided below.

- Irrelevant Features Domains
  - 3-*of*-5/0 (0 irrelevant features)
  - 3-*of*-5/3 (3 irrelevant features)
  - 3-*of*-5/5 (5 irrelevant features)
- Disjunction versus Conjunction
  - *m-of*-5/3, $1 \leq m \leq 5$
- Disjunctive *m-of-n*
  - 3-*of*-5 $\vee$ 2-*of*-3/2 (disjoint Ns)
  - 3-*of*-5 $\vee$ 3-*of*-5/2 (non-disjoint Ns)
- Conjunctive *m-of-n*
  - 3-*of*-5 $\wedge$ 2-*of*-3/2 (disjoint Ns)
  - 3-*of*-5 $\wedge$ 3-*of*-5/2 (non-disjoint Ns)

In the next section, we will summarize the results of these experiments and graph some of the results. Training and testing sets were generated by randomly choosing examples, with replacement, from the feature space. The accuracy for a particular number of examples were averaged over twenty runs on varying training and testing sets.

5.1.1 The Utility of Creating *M-of-N* Terms

Using GS for creating new terms when learning artificial *m-of-n* target concepts, resulted in accuracy at least as good as and usually better than single attribute discriminations. Performance results tended to converge for target concepts where m was nearly equal to n (e.g., *n-of-n*). On the more complex disjunctive and conjunctive *m-of-n* target concepts, creating new terms was always very beneficial. Differences for these domains were more pronounced when the prototypic regions of the simple *m-of-n* concepts that composed the disjuncts or conjuncts were disjoint with respect to feature relevance. Being disjoint, the target concepts were more prototypic in nature and more closely oriented to the bias of the *m-of-n* algorithms. Figure 1 graphs the accuracy of the conjunction of two *m-of-n* concepts – 3-*of*-(a,b,c,d,e) $\wedge$ 2-*of*-(f,g,h) when there are a total of 10 features. The two algorithms compared are the decision tree learner without constructive induction and the decision tree learner augmented with GS.

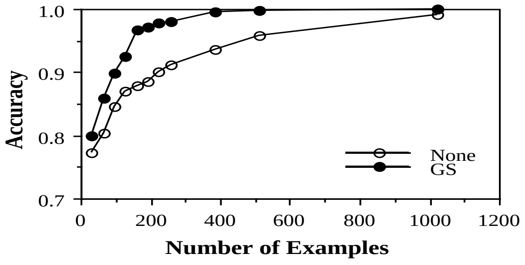

Figure 1: Conjunction of 2 *m-of-n* Concept

5.2 NATURALLY OCCURRING DATA

We have tested ID2-*of*-3 on four domains from the UCI Repository of Machine Learning Databases & Domain Theories. Currently, we are restricted to not using data with missing values. The databases selected are the lymphography, lenses, shuttle landing and mushroom databases (the feature *stalk-root* was removed from mushrooms because it had missing values). The lymphography domain was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. Thanks go to M. Zwitter and M. Soklic for providing the data.

The design of each experiment follows a 2 algorithm (learning decision trees with and without *m-of-n* terms) $\times$ n training set size design. Each experiment used different values for the training set size. We measured the accuracy of each algorithm and the number of internal nodes in the decision tree at each training set size.

On two of the domains, we found a statistically significant increase in performance when creating *m-of-n* terms during learning. Adding *m-of-n* terms had the most positive impact in the lymphography domain. Figure 2 graphs each algorithm, averaged over 20 runs. An analysis of variance indicated that the algorithm had a significant effect on the accuracy in this domain ($F(1, 342) = 26.6, p < .0001$). Similarly, using GS improved the accuracy on the shuttle landing databases ($F(1, 1782) = 14.4, p < .0001$). Figure 2 graphs the accuracy on this domain, averaged over 100 runs. On the lenses database, no significant difference in accuracy was noticed when averaged over 40 runs. ($F(1, 858) < 1$).

On the mushroom database, creating new terms interfered with the ability of the learner to converge on a more accurate concept. In this case, not creating new terms resulted in significantly higher accuracy ($F(1, 266) = 11.758, p < .001$). Figure 2 shows this data, which is averaged over 20 runs. In the mushroom domain, many of the attributes, including one highly predictive feature, have multiple values, and ID3 creates a multi-valued discrimination at each node. GS discriminations are binary (although multiple values for the same features can appear in a single node– 1-*of*-($odor = A, odor = L, odor = N$)) Using a single feature with 9 values allowed ID3 to generate highly accurate (> 95%), decision trees. Since GS makes binary decisions at its nodes, it typically required several nodes to simulate this multi-valued branch. We hypothesize that the highly predictive multi-valued attributes in this domain are responsible for the decrease in accuracy. To test this hypothesis, we compared the accuracy of the decision tree learner augmented with GS to a variant of the decision tree learn that made binary trees. In this variation, the a nodes corresponded to a boolean decision (e.g., $odor = L$), rather than a separate branch for each value of an attribute. There was not a significant difference in the accuracy of the binary decision tree learner or the decision tree learner augmented with constructive induction of *m-of-n* terms ($F(1, 266) < 1$).

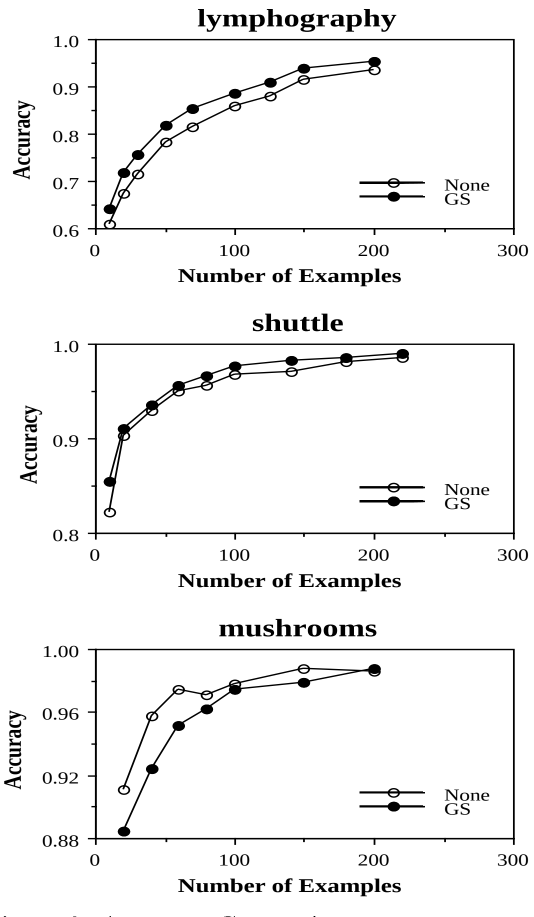

Figure 2: Accuracy Comparisons

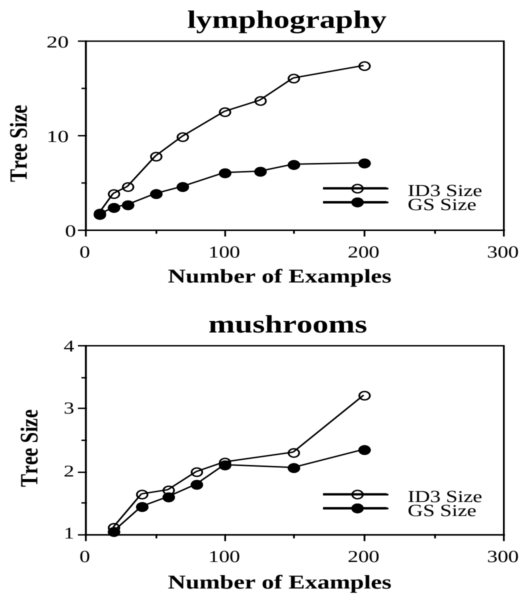

Figure 3: Tree Size Comparisons

The size of the tree (i.e., number of internal nodes) created by each algorithm are graphed as a function of the training set size for the lymphography and mushroom databases in Figure 3. A ratio of ID3's tree sizes to GS's tree sizes as a function of the training set size number is shown in Figure 4 for various domains. In general, we have observed that the greatest improvement in accuracy occurs when constructive induction results in the greatest reduction of trees sizes.

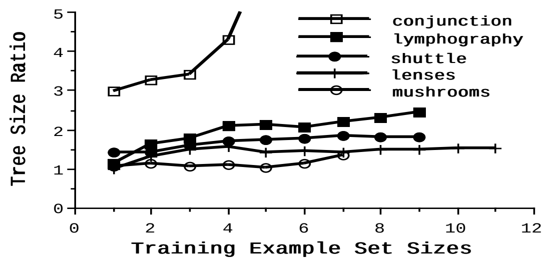

Figure 4: Tree Size Ratios

6 CONCLUSION

We investigated an approach to creating new composite *m-of-n* terms during induction of decision trees. An algorithm was explored and experimental results indicate that starting with simple *m-of-n* hypotheses and then using operators to make them more complex efficiently generates discriminators for decision trees. On artificial data that conforms to the bias of the *m-of-n* learner, creating composite terms substantially improves the performance of the learner. The algorithm also showed some statistically significant improvement on naturally occurring datasets when the tree generated were much smaller than trees generated using only feature-value pairs as discriminators. More experimentation and analysis is needed to understand which biases are appropriate for various types of naturally occurring data.

References

Barsalou, L. (1985). Ideals, central tendency, and frequency of instantiation as determinates of graded structure in categories. *Journal of Experimental Psychology: Learning, Memory and Cognition, 11*, 629–654.

Fisher, D. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning, 2*, 139–172.

Hampson, S. E. & Volper, D. J. (1986). Linear Function Neurons: Structure and Training. *Biological Cybernetics, 53*, 203–217.

Kingsland, L. C. III (1985). The evaluation of medical expert systems: Experience with the AI/RHEUM knowledge-based consultant in rheumatology. *Proceedings of the Ninth Annual Symposium on Computer Applications in Medical Care.* Washington, DC: IEEE Computer Society Press.

Murphy, P. M & Pazzani, M. J. (1991). *ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees* (Technical Report 91-37). Irvine: University of California, Department of Information and Computer Science.

Pagallo, G., & Haussler, D. (1990). Boolean feature discovery in empirical learning. *Machine Learning, 5*, 71–100.

Pitt, L. & Valiant, L. G. (1988). Computational limitations on learning from examples. *JACM, 35*, 965–984.

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

Smith, E. E., & Medin, D. L. (1981). *Categories and concepts.* Cambridge, MA: Harvard University Press.

Spackman, K. (1988). Learning categorical decision criteria in biomedical domains. *Proceedings of the Fifth International Workshop on Machine Learning* (pp. 36–46). Ithaca, NY: Morgan Kaufmann.

Utgoff, P. (1988). Perceptron trees: A case study in hybrid concept representations. *Proceedings of the Seventh National Conference on Artificial Intelligence* (pp. 601–606). Boston, MA: Morgan Kaufmann.

Appeared in: Machine Learning: Proceedings of the Eighth International Workshop (ML91).