Integrating Multiple Classifiers By Finding Their Areas of Expertise

Moshe Koppel **Sean P. Engelson**
Dept. Mathematics and Computer Science
Bar-Ilan University
Ramat Gan 52900, Israel
{koppel,engelson}@bimacs.cs.biu.ac.il
Phone: +972-3-531-8407

Abstract

Using multiple learned classifiers for increasing learning accuracy has attracted much recent interest. A central problem is how to integrate several classifiers to produce a single final classification. We have explored an approach which relies on analyzing the particular *area of expertise* of each classifier. In this paper, we describe a method which uses inductive techniques on a set of training examples to determine the areas of a particular classifier's expertise. By applying this method to each of a set of conflicting classifiers, we can determine the most reliable classifier for each new example and use that classifier to classify the example. In this way, we achieve significantly better classification accuracy than even the best individual expert or an induced combination rule.

Keywords: Combining Classifiers, Inductive Learning, Intermediate Concepts, Reliability Models

Introduction

Using multiple learned models for increasing learning accuracy has attracted much recent interest (Freund 1990; Chan & Stolfo 1995; Ali & Pazzani 1996). A central problem is how to integrate several classifiers to produce a single final classification. One approach is to use the single most reliable classifier (as in CVM (Schaffer 1993; Merz 1995)) or to use a form of weighted voting (eg, SAM (Merz 1995)); another class of methods seek to learn some form of *combination* rule, which computes a final classification based on those given by the competing classifiers (eg, (Chan & Stolfo 1995)). Our more modular approach, also investigated independently by Ortega in his recent dissertation (Ortega 1996), is to learn a *reliability model* for each classifier. We assume that each classifier has a particular subdomain for which it is most reliable, and so we learn a description of this area of expertise. Given such a description, we use the most reliable expert for diagnosis of problems in each subdomain.

We formalize this problem by assuming we are given a set of classifiers, each of which may be partially flawed. Consider, for example, a classifier for diagnosing chronic ulcer disease in which the intermediate concept "intestinal cramps" is too generally defined, but the theory is otherwise correct. Then, patients whose symptoms do not satisfy "intestinal cramps" *as defined in the classifier* will indeed be correctly diagnosed regarding chronic ulcer disease. This illustration suggests that we might use a set of training examples to determine the areas of a particular classifier's 'expertise', represented in terms of the classifier's own ontology.

Determining a Classifier's Expertise

In this section, we will consider how to use a set of classified examples in order to find the strengths and weaknesses of a given classifier. The idea is to determine for what subsets of examples the classifier's result can be taken to be reliable, and to what extent. This procedure constitutes the first stage in the arbitration mechanism for multiple classifiers, to be described in the next section.

Learning reliability trees

To be precise, an *example* E is an assignment of values to a set of *primitive attributes* $\{a_i\}$. The task is to assign each example to one of a set of classes $\{c_i\}$. A *classifier* Γ is a function from examples to classes. Given a training set $\mathcal{E}$ of examples together with their correct classifications, we want to determine Γ's probable accuracy for different subsets of possible new examples.

The basic idea is to use a decision tree induction algorithm (such as C4.5 (Quinlan 1993)) to induce a decision tree for distinguishing between cases where the classifier is correct and those where it is incorrect. Such a decision tree divides the example space up into a number of disjoint subsets, each corresponding to a leaf of the tree. For examples in each subset we can estimate the accuracy of Γ based on the fraction of examples in that subset that Γ classified correctly. In order that these estimates be useful, the decision tree must be pruned sufficiently to ensure that the subsets at the leaves are large enough to give us good estimates of the classifier's reliability. For example, a leaf for which 80 out of 100 training examples are correct for the classifier gives a better estimate of the classifier's reliability than one for which 8 out of 10 are correct. Reliability estimates should therefore take into account the sample size over which they are computed.

A further point is the choice of features for induction. It is not always the case that those regions of the example space where Γ is reliable can be simply described in terms of primitive attributes; more complicated derived attributes may need to be used to attain efficient and effective induction. Hence we use of intermediate subconcepts in Γ itself as features for induction. In a logical theory, for example, these subconcepts are the various propositions and clauses in the theory, in a decision tree these subconcepts are the conditions at the various decision nodes in the tree, and in a neural net these might be the activation values of internal nodes. These features are computed for each training example and added to its description when performing the induction.

The algorithm for learning a reliability tree is therefore as follows:

ReliabilityLearn(Γ,$\mathcal{E}$):

1. Partition $\mathcal{E}$ into $\mathcal{E}_C$ and $\mathcal{E}_I$, the subsets of $\mathcal{E}$ correctly and incorrectly classified by Γ, respectively;
2. Select a set of features as a basis for concept induction including computed features (these should always include the primitive attributes defining the examples and the example's class in Γ);
3. Build a pruned decision tree T (we use C4.5 for this purpose);
4. Compute the *reliability* for each leaf L in T based on C and I, the numbers of examples in $\mathcal{E}_C$ and in $\mathcal{E}_I$, respectively, classified to L; we use the formula

$$\frac{\max(C, I)}{C + I + \frac{1}{2}}$$

5. If $C > I$ then L's *correctness* is 'correct', otherwise it is 'incorrect'.

`illegal`	`:-`	`same-loc-ab-cd`, **adj-bf**
`illegal`	`:-`	`same-loc-ab-ef`
`illegal`	`:-`	`same-loc-cd-ef`
`illegal`	`:-`	`king-attack-king`
`illegal`	`:-`	`rook-attack-king`
`king-attack-king`	`:-`	`adj-ae, adj-bf`
`king-attack-king`	`:-`	`adj-ae, b=f`
`king-attack-king`	`:-`	`a=e`, *adj-bf*
king-attack-king	**:-**	**knight-move-ab-ef**
rook-attack-king	*:-*	*c=e, king-not-b-file*
`rook-attack-king`	`:-`	`d=f, king-not-b-rank`
`king-not-b-rank`	`:-`	`¬b=d`
`king-not-b-rank`	`:-`	`b=d, ¬between-cae`

Figure 1: A flawed version of the chess endgame theory, shown in propositional form. For simplicity we do not show the computation of low-level nodes such as `between-cae`. Added antecedents and added clauses are shown in boldface, while deleted components (not in the flawed theory) are shown in italics.

In other words, we use induction to distinguish between those examples the classifier classifies correctly and those it classifies incorrectly, also estimating the decisiveness of such determinations.

Example: Chess endgame

To illustrate the operation of **ReliabilityLearn**, consider the flawed chess endgame theory depicted in Figure 1. It should be noted that examples are of the form $(a = 1, b = 4, \ldots)$, and that propositions such as `a=b` are defined explicitly in the actual theory, (although the definitions are not shown).

The errors are that a new clause is added for `king-attack-king`, that the antecedent `adj-bf` was moved from `king-attack-king` to the first `illegal` clause, and one of the clauses for `rook-attack-king` is deleted. The critical point is that precisely when these mistakes are not used in classifying an example, its classification by the theory will be reliable.

To evaluate **ReliabilityLearn**, we used C4.5 with pruning, using internal features of the theory for performing the induction. We used a set of 1000 examples, 315 of which were positive and 685 of which were negative, where 749 were correctly classified by the theory.

We first ran **ReliabilityLearn** on all 1000 examples in the training set; Figure 2 shows the reliability tree thus constructed. We will consider in detail two of the leaves of the tree.

Consider first the leaf marked (1), where `knight-move-ab-ef` is false, `a=e` is false, and `same-loc-ab-cd`

```
knight_move_ab_ef = T:
 rook_attack_king = T: CORRECT (R = .96)
 rook_attack_king = F:
  same_loc_ab_cd = T: CORRECT (R = .80)
  same_loc_ab_cd = F: INCORRECT (R = .92)
knight_move_ab_ef = F:
 a=e = T:
  adj_bf = T: CORRECT (R = .98)
  adj_bf = F:
   same_loc_ab_ef = T: CORRECT (R = .97)
   same_loc_ab_ef = F:
    rook_attack_king = T: CORRECT (R = .95)
    rook_attack_king = F: INCORRECT (R = .92) (2)
 a=e = F:
 same_loc_ab_cd = F: CORRECT (R = .91)        (1)
  same_loc_ab_cd = T:
   illegal = T: CORRECT (R = .8)
   illegal = F: INCORRECT (R = .93)
```

Figure 2: Reliability tree for the flawed theory shown in Figure 1, learned from 1000 training examples.

is false, to which 766 of the examples are classified. For examples classified to this leaf, three of the theory's flaws have no effect: the clauses affected by the change of `adj-bf` are false regardless of the value of `adj-bf`, and when `knight-move-ab-ef` is false. The theory thus classifies examples at this leaf correctly with high reliability.

Consider now the leaf marked (2), where `knight-move-ab-ef` is true, but `rook-attack-king` and `same-loc-ab-cd` are false. For examples classified to this leaf, the theory is considered *incorrect* with high reliability. This demonstrates how inductive methods can find useful information about a theory; we can discover both when the theory is reliably correct, and when it is reliably incorrect. In the latter case, for theories such as this one where the root is two-valued, the right thing to do is to classify opposite to the theory (Engelson & Koppel 1995).

Figure 3 shows the results of using the theory to classify only those examples whose reliability score is above a certain threshold, marking all others as "unknown" (assigning the class opposite to that given by the theory when it is reliably incorrect for an example). We used 10-fold cross-validation, withholding varying amounts of nested training data in order to create a learning curve.

As can be seen, the number of examples assigned a class by this procedure tends to increase with the amount of training, as the leaves become more decisive. At the same time, classification accuracy increases, and

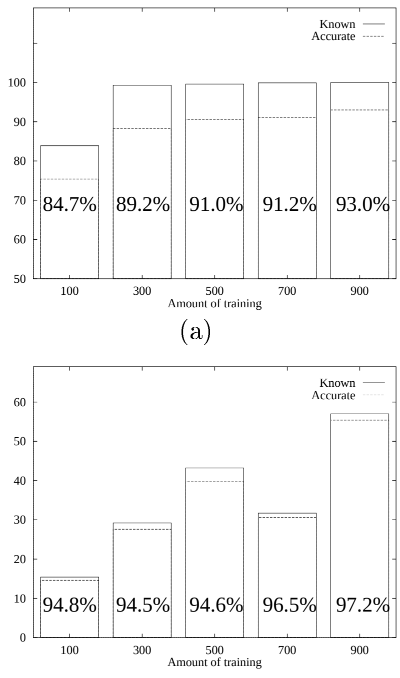

Figure 3: Using the reliability tree for selective classification with the chess endgame theory for reliability thresholds of (a) 0.8 and (b) 0.9. Plotted are average fraction of known examples (those assigned a class) and classification accuracy for known examples, over ten disjoint 100-example test sets.

becomes considerably higher than classification by the theory alone (74.9%). This is due to reversing reliably incorrect classifications by the theory, as well as rejecting unreliably classified examples.

Discussion

As the example above illustrates, much of the power of the method comes from the use of internal propositions of a flawed theory as features for induction. The use of such internal propositions as features for use in induction is not new, going back to the work of Drastal and Raatz (1989), and more recently to the work of Ortega and Fisher (1995) and Donoho and Rendell (1995). Our approach differs, however, in that rather than learn a decision tree directly for the target concept, we learn a decision tree which distinguishes when the given theory is correct and when it is incorrect. Due to the greedy nature of C4.5, the trees learned for the correct/incorrect concept may differ greatly from those learned directly for the target concept. In particular, when flaws are focused in one subconcept of the theory, the tree constructed for the correct/incorrect concept may be more natural, in that it will simply rule out classifying on the basis of that subconcept.

The fact that we learn decision trees for the 'theory correct/incorrect' concept also sheds some light on the utility of using internal nodes. As the example above showed, internal nodes are particularly useful in diagnosing examples for which the theory is unreliable. This is because it is precisely truth or falsehood of internal nodes in the *flawed* theory which will predict well whether particular flaws affect the classification of the given example, and so will give a better estimate of the theory's reliability. Hence, we expect the internal features of a theory, in combination with the root of that theory, to be better features for induction (whether for the target class or for correct/incorrect) than either the primitive attributes or other kinds of synthetic features.

Combining Multiple Classifiers

Now consider the case where we are given several possibly flawed classifiers for a particular domain. Rather than try to combine the results of the different classifiers, we seek to choose the *right* classifier to use to classify each new input example. By using each classifier just on that subdomain for which it is most reliable, our overall results should be considerably better than those of the best individual classifier. If different classifiers are experts in different subdomains, then the more experts we have, the greater the area that is covered reliably (Ali & Pazzani 1996).

Given a set of classifiers $\Gamma_1,\ldots,\Gamma_n$ for a binary concept and a set of classified examples $\mathcal{E}$, we use the following arbitration algorithm to classify new input examples by using the most reliable expert for each one.

Arbitrate($\Gamma_1,\ldots,\Gamma_n,\mathcal{E}$):

1. Construct a reliability decision tree for each input classifier Γ_i, $T_i =$ **ReliabilityLearn**($\Gamma_i, \mathcal{E}$);
2. For each new input example E:
   (a) Evaluate the correctness c_i and reliability r_i of each classifier for E, using T_i;
   (b) Let Γ_{i_*} be the classifier with highest reliability r_i;
   (c) If $c_{i_*} =$'correct', assign E the class $\Gamma_{i_*}(E)$;
   (d) Else if $c_{i_*} =$'incorrect', assign E the class $\neg\Gamma_{i_*}(E)$;

Put simply, the algorithm classifies each new example E according to the classifier that has proven most reliable in classifying examples in the training set similar to E. Of course, this classification may be the opposite of that given by one of the given classifiers, if that classifier is incorrect with higher reliability than that of any other classifier being correct (or incorrect). (If the concept being learned is not binary, we may simply use the most reliably correct classifier on each example.)

Results

We first evaluated the use of arbitration between multiple classifiers by using three flawed versions of the chess endgame theory, each with six random flaws (each the insertion or deletion of an antecedent or a clause). We used a set of 1000 preclassified examples in 10-fold cross-validation experiments.

The accuracies of the three theories over all 1000 examples were all near 70%, as shown in Figure 4. Using C4.5 to learn the target concept directly on primitive attributes gives similarly low accuracy. By including the classifications given by all three theories as a feature for building C4.5's decision tree, however, accuracy improves noticeably. (This strategy is equivalent to Chan and Stolfo's *class-attribute-combiner* (Chan & Stolfo 1995).) The best results shown are obtained by using the **Arbitrate** algorithm, with an accuracy of 93.4% using 10-fold cross-validation. This shows how by properly combining the information in the theories with the information in the training set, high classification accuracy can be obtained. Preliminary results for C4.5 using all internal nodes as features for standard induction show it also to achieve good accuracy. In general, however, using all the internal features of

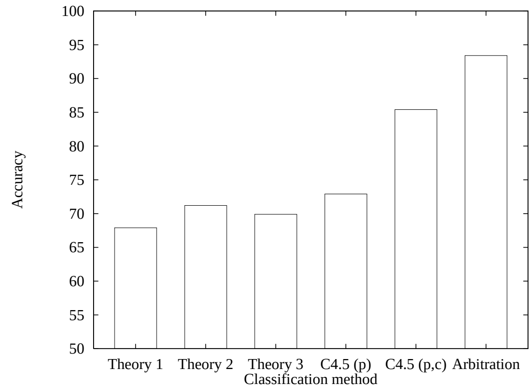

Figure 4: The accuracy of the three flawed chess endgame theories on 1000 training examples; the accuracy of C4.5 with primitive attributes (C4.5 (p)), C4.5 on primitive attributes and theory classifications (C4.5 (p,c)), and arbitration between the theories, using 10-fold cross-validation.

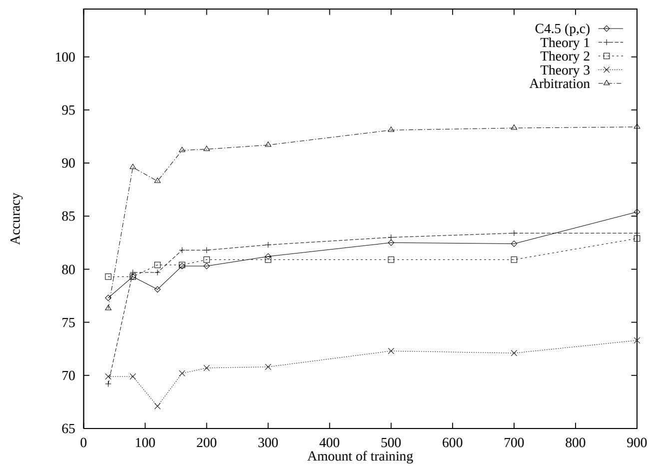

Figure 5: Learning curves for classification accuracy of: C4.5 with primitives and theory classifications, three flawed theories combined with their reliability trees (using internal nodes), and **Arbitrate** for the three theories. Each curve was generated using 10-fold cross-validation on 1000 examples, by withholding different numbers of examples from each 900-example training set to create a learning curve.

a number of theories for induction can lead to overfitting. In addition, the arbitration approach is modular, allowing a reliability model to be constructed independently for each component theory.

It is still possible, however, that the gain in accuracy actually arises from the building of the reliability trees for each theory, and that arbitration itself contributes little. To test the hypothesis that selecting the correct theory is important, we generated the learning curves shown in Figure 5. Five learning curves are compared: C4.5 using primitives and theories' classifications (as above), for each of the three theories, classifying according to the theory's reliability tree (either using or reversing the theory's classification), and arbitrating between the three theories. As is clear from the figure, arbitration performs substantially better than each of the three individual theories, even with their associated decision trees. This reflects the fact that **Arbitrate** uses different theories for different types of examples.

We now give results for a more extensive experiment testing this hypothesis. For each trial we generated five flawed theories, each by inserting twelve random errors into the base theory. We randomly ordered the theories and recorded the average accuracy produced by running **Arbitrate** on the first theory, the first two theories, and so forth, doing 10-fold cross-validation on 1000 examples. We ran 8 such trials and averaged the results for each number of theories used in arbitration.

The results of this experiment are summarized in

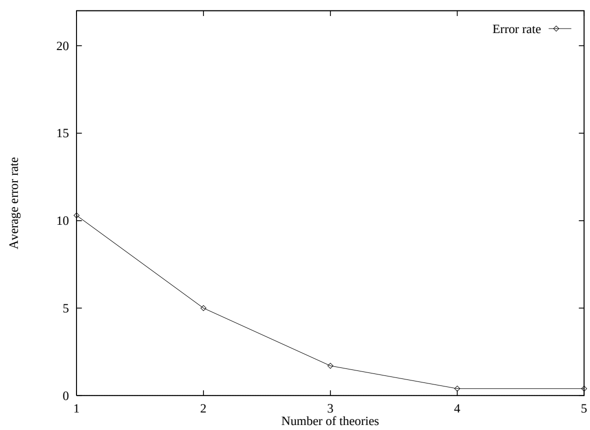

Figure 6: Average error rate in 10-fold cross-validation as the number of theories arbitrated between increases. See text for full explanation.

Figure 6. As is apparent, the error rate decreases greatly with the number of theories used. In fact, the error appears to decrease geometrically, which is what we would expect from optimally taking advantage of differently flawed theories (Ali & Pazzani 1996).

Discussion

We have seen how different classifiers can be usefully combined by exploiting a set of training examples to discover for which subdomain each classifier is most reliable. Our results and those of Ortega (1996) (using his MMC+FPM method) suggest the general rule that the more fine-grained the analysis of the areas of expertise of the competing classifiers', the more effectively they can be combined. Similarly, Chan and Stolfo (1995) found that learning an arbitration rule based just on the classes assigned by the classifiers was markedly less accurate than learning a rule that used both the classes and the primitive attributes of the examples themselves. We have taken this idea a step further by also using intermediate subconcepts in the classifiers themselves as features for induction. This sort of constructive induction is particularly useful for arbitration because it is just the intermediate concepts of a classifier (even if flawed) which predict well the reliability of the classifier as a whole.

Acknowledgements

The authors gratefully acknowledge the substantial contributions of Hillel Walters to this work. The second author is supported by a postdoctoral fellowship from the Fulbright Foundation.

References

Ali, K., and Pazzani, M. 1996. Error reduction through learning multiple descriptions. *Machine Learning.* (to appear).

Chan, P. K., and Stolfo, S. J. 1995. A comparative evaluation of voting and meta-learning on partitioned data. In *Proc. Int'l Conference on Machine Learning.*

Donoho, S. K., and Rendell, L. A. 1995. Rerepresenting and restructuring domain theories: A constructive induction approach. *Journal of Artificial Intelligence Research* 2:411–446.

Drastal, G., and Raatz, S. 1989. Empirical results on learning in an abstraction space. In *Proc. Int'l Joint Conference on Artificial Intelligence.*

Engelson, S. P., and Koppel, M. 1995. Distilling reliable information from unreliable theories. In *Proc. Int'l Conference on Machine Learning.*

Freund, Y. 1990. An improved boosting algorithm and its implications on learning complexity. In *Proc. Fifth Workshop on Computational Learning Theory.*

Merz, C. J. 1995. Dynamic learning bias selection. In *Proceedings of The Fifth Int'l Workshop on AI and Statistics.*

Ortega, J., and Fisher, D. 1995. Flexibly exploiting prior knowledge in empirical learning. In *Proc. Int'l Conference on Machine Learning.*

Ortega, J. 1996. *Making The Most Of What You've Got: Using Models and Data To Improve Prediction Accuracy.* Ph.D. Dissertation, Vanderbilt University.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

Schaffer, C. 1993. Selecting a classification method by cross validation. In *Proceedings of The Fourth Int'l Workshop on AI and Statistics.*