Pairwise Classification as an Ensemble Technique

Johannes Fürnkranz
Austrian Research Institute for Artificial Intelligence
Schottengasse 3, A-1010 Wien, Austria
juffi@oefai.at

OEFAI-TR-2002-20

Abstract

In this paper we investigate the performance of pairwise (or round robin) classification, originally a technique for turning multi-class problems into two-class problems, as a general ensemble technique. In particular, we show that the use of round robin ensembles will also increase the classification performance of decision tree learners, which could directly handle multi-class problems. The performance gain is not as large as for bagging and boosting, but on the other hand round robin ensembles have a clear semantics. Furthermore, we show that the advantage of pairwise classification over direct multi-class classification and one-against-all binarization increases with the number of classes, and that round robin ensembles form an interesting alternative for problems with ordered class values.

1 Introduction

In a recent paper (Fürnkranz, 2001), we analyzed the performance of pairwise classification (which we call *round robin* learning) for handling multi-class problems in rule learning. Most rule learning algorithms handle multi-class problems by converting them into a series of two-class problems, one for each class, each using the examples of the corresponding class as positive examples, and all others as negative examples. This procedure is known as *one-against-all* class binarization. Round robin binarization, on the other hand, converts a c-class problem into a series of two-class problems by learning one classifier for each pair of classes, using only training examples for these two classes and ignoring all others. A new example is classified by submitting it to each of the $c(c-1)/2$ binary classifiers, and combining their predictions via simple voting. The most important result of the previous study was that this procedure not only increases predictive accuracy, but that it is also no more expensive than the more commonly used one-against-all approach.

Obviously, round robin classifiers may also be interpreted as an ensemble classifier that, similar to error-correcting output codes (Dietterich and Bakiri, 1995), constructs an ensemble by transforming the learning problem into multiple other problems and learning a classifier for each of them.[1] In this paper, we will investigate the question whether round robin class-binarization can also improve performance for learning algorithms that can naturally handle multi-class problems, in our case decision tree learners. We will start with a brief recapitulation of our previous results on round robin learning (Section 2), and then investigate two questions: First, in Section 3, we will investigate the performance of round-robin binarization as a general ensemble technique and compare its performance to bagging and boosting. We will also evaluate a straight-forward integration of bagging and round robin learning. As more classes result in a larger ensemble of classifiers, it is reasonable to expect that the performance of round robin ensembles depends crucially on the number of classes of the problem. In Section 4, we will investigate this relation on classification problems with identical attributes but varying numbers of classes, which were obtained by discretizing the target variables of regression problems. Our results will show that round robin learning can indeed improve the performance of the c4.5 and c5.0 decision tree learners, and that a higher number of classes increases its performance, in particular in comparison to a one-against-all binarization.

2 Round Robin Classification

In this section, we will briefly review round robin learning (aka pairwise classification) in the context of our previous work in rule learning (Fürnkranz, 2001; 2002b). Separate-and-conquer rule learning algorithms (Fürnkranz, 1999) are typically formulated in a concept learning framework, where the goal is to find a definition for an unknown concept, which is implicitly defined via a set of positive and negative examples. Within this framework, multi-class problems, i.e., problems in which examples may belong to (exactly) one of several categories, are usually addressed by defining a separate concept learning problem for each class. Thus the original learning problem is split into a series of binary concept learning problems—one for each class—where the positive training examples are those belonging to the corresponding class and the negative training examples are those belonging to all other classes. This technique for dealing with multi-class problems in rule learning has been proposed by Clark and Boswell (1991), but is also well-known in other areas such as neural networks (Anand et al., 1995), support vector machines (Cortes and Vapnik, 1995), or statistics (cf. multi-response linear regression). A variant of the technique, in which classes

[1] In fact, Allwein et al. (2000) show that pairwise classification (and other class binarization techniques) are a special case of a generalized version of error-correcting output codes, which allows to specify that certain classes should be ignored for some problems (in addition to assigning them to a positive or a negative class, as conventional output codes do).

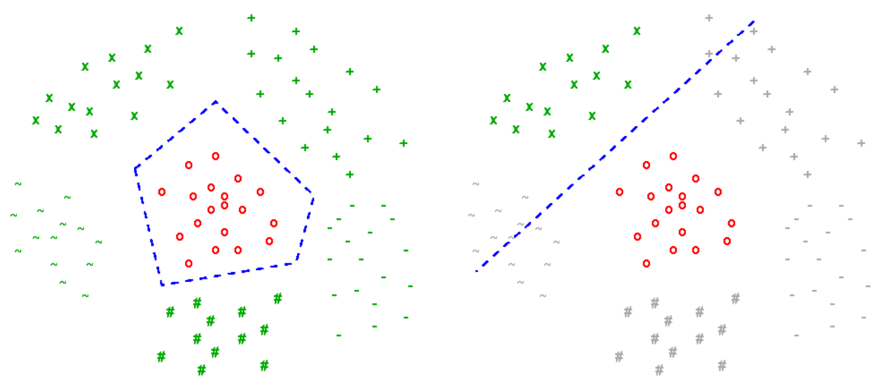

Figure 1: *One-against-all class binarization* (left) turns each c-class problem into c binary problems, one for each class, where each of these problems uses the examples of its class as the positive examples (here o), and all other examples as negatives. *Round robin class binarization* (right) turns each c-class problem into $c(c-1)/2$ binary problems, one for each pair of classes (here 0 and x) ignoring the examples of all other classes.

are first ordered (e.g., according to their relative frequencies in the training set) is used in the ripper rule learning algorithm (Cohen, 1995).[2]

The basic idea of round robin classification is to convert a c-class problem into $c(c-1)/2$ binary problems, one for each pair of classes. Note that the binary decision problems in the round robin case not only contain fewer training examples (all examples that do not belong to the pair of classes are ignored), but that the decision boundaries that separate pairs of classes may also be considerably simpler than the boundaries that discriminate one class from all other classes. In fact, in the example shown in Figure 1, each pair of classes can be separated with a linear decision boundary, while more complex boundaries are required to separate a class from all other classes.[3] While this idea is known from the literature (cf. Section 8 of (Fürnkranz, 2002b) for a brief survey), in particular from the field of support vector machines (Schmidt and Gish, 1996; Hastie and Tibshirani, 1998; Kreßel, 1999; Hsu and Lin, 2002), the main contributions of (Fürnkranz, 2001) were to empirically evaluate the technique for rule learning

[2]An alternative setting for rule learning is the the decision list framework, first formalized by Rivest (1987), where rules are evaluated by dynamically labeling each learned rule with the class that is most common among the examples it covers. This approach is, e.g., used in the original CN2 system (Clark and Niblett, 1989).

[3]Evidence for this was also seen in practical applications: Knerr et al. (1992) observed that the classes of a digit recognition task were pairwise linearly separable, while the corresponding one-against-all task was not amenable to single-layer networks, while Hsu and Lin (2002) obtained a larger advantage of round robin binarization over unordered binarization for support vector machines with a linear kernel than for support vector machines with a non-linear kernel on several benchmark problems.

algorithms and to show that it is preferable to the one-against-all technique that is used in most rule learning algorithms. In particular, round robin binarization helps ripper to outperform c5.0 on multi-class problems, whereas c5.0 outperforms the original version of ripper on the same problems. More importantly, we analyzed the computational complexity of the approach, and demonstrated that despite the fact that its complexity is quadratic in the number of classes, the algorithm is no slower than the conventional one-against-all technique. It is easy to see this, if we consider that in the one-against-all case, each training example is used c times (namely in each of the c binary problems), while in the round robin approach, each examples is only used $c-1$ times, namely only in those binary problems, where its own class is paired against one of the other $c-1$ problems. Furthermore, the advantage of pairwise classification increases for computationally expensive (super-linear) learning algorithms. The reason is that expensive learning algorithms learn many small problems much faster than a few large problems. For details we refer to (Fürnkranz, 2002b).

3 Round Robin Ensembles

In this section we suggest that round robin classification may also be interpreted as an ensemble technique, and its performance gain may be viewed in this context. Obviously, the final prediction is made by exploiting the redundancy provided by multiple models, each of them being constructed from a subset of the original data. However, contrary to subsampling approaches like bagging and boosting, these datasets are constructed deterministically.[4] In this respect pairwise classification shares more similarities with error-correcting output codes (Dietterich and Bakiri, 1995), but differs from it through the fixed procedure for setting up the new binary problems and the fact that each of the new problems is smaller than the original problem. In particular the latter fact may often cause the subproblems in pairwise classification to be conceptually simpler than the original problem (as illustrated in Figure 1).

In previous work (Fürnkranz, 2001), we observed that the improvements in accuracy obtained by r^3 (a round robin version of ripper) over ripper were quite similar to those obtained by c5.0-boost (c5.0 called with the option -b, i.e., 10 iterations of boosting) over c5.0 on the same problems. Round robin binarization seemed to work whenever boosting worked, and vice versa. Figure 2 plots the error ratios of r^3/ripper versus those of c5.0-boost/c5.0. The correlation coefficient r^2 was about 0.618, which is in the same range as correlation coefficients for bagging and boosting (Opitz and Maclin, 1999). We interpreted this as weak evidence that the performance gains of round robin learning may be comparable to those of other ensemble methods and that it could be used as a general method for improving a learner's performance on multi-class problems. We will further investigate this question in this section and will in particular

[4] Boosting is also deterministic if the base learner is able to use weighted examples. Often, however, the example weights are interpreted as probabilities which are used for drawing the sample for the next boosting iteration.

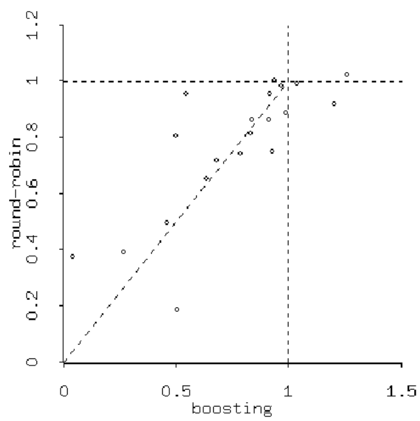

Figure 2: Error reductions ratios of boosting (x-axis) vs. round robin (y-axis).

focus upon a comparison of round robin learning with boosting (Section 3.1) and bagging (Section 3.2), and upon the potential of combining it with these techniques. Large parts of this section will also appear in (Fürnkranz, 2002b).

3.1 Comparison to Boosting

As a first step, we perform a direct comparison of the performance of c5.0 and c5.0-boost (c5.0 called with the parameter -b, i.e., 10 iterations of boosting) to a round robin procedure with c5.0 as the base learning algorithm. Thus, each c-class problem is turned into $c(c-1)/2$ binary problems and c5.0 is used to learn a decision tree for each of them. For predicting the class of a test example, it is submitted to all $c(c-1)/2$ classifiers and their predictions are combined via unweighted voting. Ties are broken in favor of larger classes. Table 1 shows the results of a 10-fold cross-validation on 17 datasets with 4 or more classes from the UCI repository (Blake and Merz, 1998).

The first thing to note is that the performance of c5.0 does indeed improve by about 10% on average[5] if round robin binarization is used as a pre-processing step for multi-class problems. This is despite the fact that c5.0 can directly handle multi-class problems and does not depend on a class binarization routine. However, the direct comparison between round robin classification and boosting shows that the improvement of c5.0-rr over c5.0 is not as large as the improvement provided by boosting: although there are a few cases where round robin outperforms boosting, c5.0-boost seems to be more reliable than c5.0-rr, producing an average error reduction of more than 26% on these 17 datasets. The correlation between the error reduction rates of c5.0-boost and c5.0-rr is

[5] As these are relative performance measures, we use a geometric average so that x and $1/x$ average to 1.

Table 1: *Boosting:* A comparison between round robin binarization and boosting, both with c5.0 as a base learner. The first column shows the results of c5.0, while the next three column pairs show the results of round robin learning, boosting, and the combination of both, all with c5.0 as a base learner. For these, we give both the absolute error rate and the performance ratio relative to the base learner c5.0. The last line shows the geometric average of these ratios.

dataset	c5.0	round robin		boosting		both	
abalone	78.48	75.08	*0.957*	77.88	*0.992*	74.67	*0.951*
car	7.58	5.84	*0.771*	3.82	*0.504*	1.85	*0.244*
glass	35.05	24.77	*0.707*	27.57	*0.787*	22.90	*0.653*
image	3.20	2.90	*0.905*	1.60	*0.500*	1.73	*0.541*
lr spectrometer	51.22	51.79	*1.011*	46.70	*0.912*	51.98	*1.015*
optical	9.20	5.04	*0.547*	2.46	*0.267*	2.54	*0.277*
page-blocks	3.09	2.98	*0.964*	2.58	*0.834*	2.78	*0.899*
sat	13.82	13.16	*0.953*	9.32	*0.675*	9.00	*0.651*
solar flares (c)	15.77	15.69	*0.995*	16.41	*1.041*	16.70	*1.059*
solar flares (m)	4.90	4.90	*1.000*	5.90	*1.206*	5.83	*1.191*
soybean	9.66	6.73	*0.697*	6.59	*0.682*	6.44	*0.667*
thyroid (hyper)	1.11	1.14	*1.024*	1.03	*0.929*	1.33	*1.190*
thyroid (hypo)	0.58	0.69	*1.182*	0.32	*0.545*	0.53	*0.909*
thyroid (repl.)	0.72	0.74	*1.037*	0.90	*1.259*	0.90	*1.259*
vehicle	26.24	29.20	*1.113*	24.11	*0.919*	23.17	*0.883*
vowel	21.72	19.49	*0.898*	8.89	*0.409*	14.75	*0.679*
yeast	43.26	40.63	*0.939*	41.85	*0.967*	40.77	*0.942*
average			*0.909*		*0.735*		*0.757*

very weak ($r^2 = 0.276$), which refutes our earlier hypothesis, and brings up the question whether there is a fruitful combination of boosting and round robin classification. Unfortunately, the last column of Table 1 answers this question negatively: although there are some cases where the combination performs better than both of its constituents, the results of using round robin classification with c5.0-boost as a base learner does—on average—not lead to performance improvements over boosting.

These results are analogous to the results of Schapire (1997) who compared AdaBoost.OC (error-correcting output codes as a binarization scheme for conventional 2-class AdaBoost) with AdaBoost.M1 (Freund and Schapire, 1997), AdaBoost's straight-forward adaptation for multi-class base learners (a version of which is presumably also implemented in c5.0 (Quinlan, 1996)), and found no significant differences between the two (using c4.5 (Quinlan, 1993) as a base learner). Similar to our comparison between c5.0-boost and round robin binarization, Schapire (1997) also found that boosting outperformed binarization via error-correcting output codes. In subsequent work, Allwein et al. (2000) showed that the performance gain of pairwise classification using AdaBoost as a base learner is on average indiscernible from the performance gain of alternative binarization schemes, including some employing error-correcting output codes (such as AdaBoost.OC).

3.2 Comparison to Bagging

A natural extension of the round robin procedure is to consider training multiple classifiers for each pair of classes (analogous to sports and games tournaments where each team plays each other team several times). For algorithms with random components (such as ripper's internal split of the training examples, or the random initialization of back-propagation neural networks) this could simply be performed by running the algorithm on the same dataset with different random seeds. For other algorithms there are two options: randomness could be injected into the algorithm's behavior (Dietterich, 2000) or random subsets of the available data could be used for training the algorithm. The latter procedure is more or less equivalent to bagging (Breiman, 1996). We will evaluate this option in this section.

Bagging was implemented by drawing 10 samples with replacement from the available data. Ties were broken in the same way as for round robin binarization, i.e., by simple voting using the *a priori* class probability as a tie breaker. Similarly, bagging was integrated with round robin binarization by drawing 10 independent samples of each pairwise classification problem. Thus we obtained a total of $10c(c-1)/2$ predictions for each c-class problem, which again were simply voted. The number of 10 iterations was chosen arbitrarily (to conform to c5.0-boost's default number of iterations) and is certainly not optimal (in both cases).

Table 2: *Bagging:* A comparison of round robin learning versus bagging and of the combination of both using ripper, c5.0 and c5.0-boost as the base classifiers.

	base	round robin	bagging	both
ripper	1.0	0.747	0.811	0.685
c5.0	1.0	0.909	0.864	0.838
c5.0-boost	1.0	1.029	0.977	1.019

Table 2 shows the results of a comparison of round robin learning, bagging, and a combination of both for ripper, c5.0, and c5.0-boost as base learners. We omit the detailed results here and show only the geometric average of the improvement rates of the ensemble techniques.[6] The results show that the performance of the simple round robin (second column) can be improved considerably by integrating it with bagging (last column), in particular for ripper. The bagged round robin procedure reduces ripper's error on the datasets to about 68.5% of the original error (third line from the bottom). Again, the advantage of the use of round robin learning is less pronounced for c5.0 (it is even below

[6]Detailed results for ripper can be found in (Fürnkranz, 2002b).

the improvement given by our simple bagging procedure), and the combination of c5.0-boost and round robin learning does not produce an additional gain.

Note that these average performance ratios are always relative to the base learner. This means they are only comparable within a line not between lines. For example, c5.0's performance as a base learner was considerably better than ripper's by a factor of about 0.891. In terms of absolute performances, the best performing algorithm (on average) was bagged c5.0-boost, which has about 64% of the error rate of basic ripper. This confirms previous good results with combinations of bagging and boosting (Pfahringer, 2000; Krieger et al., 2001). In comparison, the combination of round robin and bagging for ripper (68.5% of ripper's error rate) is relatively close behind, in particular if we consider the bad performance of ripper in comparison to c5.0. An evaluation of a boosting variant of ripper (such as slipper; Cohen and Singer, 1999) would be of interest.

Even though they do not reach the same performance level as alternative ensemble methods, we believe that round robin ensembles nevertheless deserve attention because of the fact that each of the classifiers in the ensemble has a clearly defined semantic (namely to predict whether an unseen example is more likely to be of class i or class j, for all pairs of classes $i \neq j$), which may lead to a better understanding of the predictions of the ensemble. In fact, Pyle (1999, p.16) proposes a very similar technique called *pairwise ranking* in order to facilitate human decision-making in ranking problems. He claims that it is easier for a human to determine an order between n items if one makes pairwise comparisons between the individual items and then adds up the wins for each item, instead of trying to order the items right away.

4 Dependence on Number of Classes

The size of a round robin ensemble (without the use of bagging) depends on the number of classes in the problem. In this section, we will analyze the behavior of round-robin learning with a varying number of classes. To this end, we decided to follow the experimental set-up described in (Frank and Hall, 2001), where regression problems were converted into classification problems with a varying number of classes using an equal-frequency discretization on the target variable (i.e., the resulting problems are class-balanced). We use exactly the same datasets for our evaluation, and compare j48 (the c4.5 clone implemented in the Weka data mining library; Witten and Frank 2000) to j48-rr, a version that uses pairwise classification with j48 as a base learner. The implementation of j48-rr was provided by Richard Kirkby, which also gave us the opportunity to confirm our previous findings on an independent implementation of the algorithm.

Table 3 shows the 10-fold cross-validation error rates of each algorithm on each of the 29 problems, together with a sign that indicates whether j48 (+) or the round robin version (−) had the higher estimated accuracy. No significance test was used to compute these signs, but the null hypothesis that the performance of j48 and j48-rr is identical on these 29 datasets can be rejected

Table 3: Comparison of the error rates of j48 and j48-rr on 29 regression datasets, which were class-discretized to classification problems with 3, 5, and 10 class values.

	3 classes			5 classes			10 classes		
dataset	j48	j48-rr		j48	j48-rr		j48	j48-rr	
Abalone	36.10	35.37	–	53.66	49.37	–	73.16	68.83	–
Ailerons	25.21	24.87	–	43.02	41.83	–	63.35	61.17	–
Delta Ailerons	19.67	19.61	–	44.46	42.54	–	58.73	54.80	–
Elevators	37.76	35.30	–	52.24	47.80	–	71.38	66.29	–
Delta Elevators	30.13	29.17	–	52.31	49.00	–	63.09	57.64	–
2D Planes	13.39	13.39	–	24.63	24.66	+	46.95	45.75	–
Pole Telecom	4.37	4.40	+	4.95	5.08	+	9.16	9.38	+
Friedman Artificial	19.73	19.52	–	35.15	34.31	–	58.96	56.79	–
MV Artificial	0.47	0.48	+	0.81	0.82	+	1.82	1.91	+
Kinematics	36.29	35.74	–	56.37	53.70	–	75.70	72.92	–
CPU Small	21.54	21.01	–	36.19	34.32	–	57.81	54.83	–
CPU Act	19.25	18.82	–	33.23	31.46	–	54.27	51.63	–
House 8L	30.46	29.53	–	49.75	47.32	–	69.59	65.81	–
House 16H	31.79	30.52	–	50.98	47.65	–	70.72	65.97	–
Auto MPG	21.98	24.10	+	40.50	41.58	+	60.40	62.74	+
Auto Price	14.97	14.40	–	37.92	34.53	–	63.21	66.29	+
Boston Housing	25.10	24.11	–	40.38	38.44	–	61.05	58.16	–
Diabetes	48.84	53.26	+	74.42	64.19	–	77.44	75.12	–
Pyrimidines	50.54	46.35	–	58.24	60.81	+	75.81	78.51	+
Triazines	46.72	46.88	+	61.08	63.17	+	83.06	83.17	+
Machine CPU	28.04	25.79	–	42.87	39.86	–	63.44	60.96	–
Servo	24.31	19.88	–	44.67	39.52	–	65.33	57.72	–
Wisconsin Breast C.	63.20	63.35	+	76.60	74.43	–	88.87	86.65	–
Pumadyn 8NH	34.02	33.43	–	53.96	49.22	–	76.18	71.72	–
Pumadyn 32H	22.44	21.54	–	37.35	35.17	–	58.12	54.93	–
Bank 8FM	13.98	14.16	+	26.86	26.25	–	50.29	48.33	–
Bank 32NH	44.21	43.53	–	62.59	60.84	–	75.74	71.04	–
California Housing	20.97	20.68	–	36.66	35.45	–	57.30	56.41	–
Stocks	8.75	8.51	–	13.09	13.42	+	27.40	28.05	+

with 99% confidence in all three cases, even using the simple sign test (which has a comparably high Type II error). In all three cases, j48-rr outperformed j48 in 22 out of 29 datasets. Four of the datasets (*Pole Telecom*, *MV Artificial*, *Auto MPG*, and *Triazines*) seem to be completely unamenable to pairwise classification, i.w., j48 performs better in all three classification settings.

This, however, does not tell us about the size of the improvement. Inspection of a few cases in Table 3 reveals that on several datasets the advantage of j48-rr over j48 seems to increase with the number of classes, at least for the step from three to five classes (cf., e.g., *Abalone*). In an attempt to make this observation

Table 4: Error, training time, testing time for a round robin version of j48, a one-against-all version of j48, regular j48, and the binarization technique for ordered classification of Frank and Hall (2001).

	j48-rr	j48-1a		j48		j48-ORD
error						
3	26.82	26.57	*0.99*	27.39	*1.02*	26.30
5	40.92	42.48	*1.04*	42.93	*1.04*	41.43
10	58.40	63.83	*1.10*	60.63	*1.03*	58.92
training time						
3	17.65	35.34	*1.66*	15.99	*0.82*	
5	27.90	53.52	*1.53*	24.58	*0.78*	
10	45.47	84.78	*1.38*	35.76	*0.64*	
testing time						
3	0.66	0.37	*0.50*	0.22	*0.27*	
5	1.64	0.50	*0.30*	0.30	*0.17*	
10	6.81	0.67	*0.14*	0.48	*0.09*	

more objective, we summarized the results of these two algorithms in Table 4, and also included the results of j48-1a, a version of j48 that uses a one-against-all binarization. We show the average performance of all algorithms, and the geometric averages of the performance ratios of j48-rr over j48, and j48-1a over j48. Note that both measures are somewhat problematic: the average is dominated by results with large variations among the algorithms (particularly so for the run-time results, which are discussed below), while the performance ratios, which may be viewed as differences normalized by the performance of j48, are somewhat influenced by the fact that the default accuracy of the problems decreases with an increasing number of classes, and consequently error differences for problems with more classes receive a lower weight (assuming there is some correlation of the performance of the algorithms and the default accuracy of the problem).

The results show that the performance improvement of round robin over a one-against-all approach increases steadily with both measures. The performance improvement over j48 also increases in absolute terms, but stays about the same in relation to the error rate of j48 (the improvement is always approximately 3% of j48's error rate). This seems to indicate that the one-against-all class binarization becomes more and more dangerous for larger numbers of classes. A possible reason could be that the class distributions of the binary problems in the one-against-all case become more and more skewed for an increasing number of classes (because the number of examples for each class decreases).

The fact that we chose almost the same experimental setup as Frank and Hall (2001) allows us to evaluate the performance of round robin learning in domains with ordered classification. The only difference is that we only used a single 10-fold cross-validation, while Frank and Hall (2001) averaged ten 10-fold cross-validation runs. However, these differences are negligible: in the six experiments that we both performed—those using j48 and j48-1a—their average accuracy estimates and our estimates differed by at most 0.05. Hence we are quite confident, that the results for j48-ORD, which we computed from the tables published in (Frank and Hall, 2001), are comparable to our results for j48-rr. The interesting result is that there is almost no difference between the two. Apparently general round robin learning is as good for ordered classification as the modification to one-against-all learning that was suggested by Frank and Hall (2001). This opens up the question whether a suitable adaptation of round robin learning could further improve these results, which we leave open for future work.

We also used these experiments to get the confirmation of an independent implementation for round robin's favorable run-time results over one-against all. The lower two parts of Table 4 show the summaries for the training and test times. As expected, round robin binarization is considerably faster than a one-against-all approach, despite the fact that round robin binarization generates $c(c-1)/2$ binary problems for a c-class problem, while the one-against-all technique generates only c problems. However, the advantage seems to decrease with an increasing number of classes. This is not consistent with our expectations that the performance loss induced by the class binarization decreases with an increasing number of classes (Fürnkranz, 2002b, Theorem 11)). We are not exactly sure about the reason for this contradiction. One explanation could be that the overhead for initializing the binary learning problems (which we did not take into account in our theoretical analysis) is worse than expected, for example because of memory swapping if not all $c(c-1)/2$ training sets can be held in memory, or because the overhead of initializing the algorithm $c(c-1)/2$ times may dominate the run-time, in particular for faster training sets. The latter point is confirmed when we look at the averages (which in the case of run-times are clearly dominated by a few slow datasets), where round robin is consistently almost twice as fast than one-against all (which is approximately what we would expect from our theoretical results).

For completeness, Table 4 also shows the results from testing time, which are clearly the worst for the round robin case. Here nothing can be saved because each example has to be tested on $c(c-1)/2$ theories for the round robin case, on c theories for the one-against-all case, and on 1 theory for regular j48 (but cf. Platt et al., 2000, for a faster alternative). These results are also lower than what could be expected by the above deliberations by approximately a factor of 2, for which we do not have an explanation (possibly an inefficient implementation of the testing of the theories).

5 Conclusions

Pairwise classification is an increasingly popular technique for efficiently and effectively converting multi-class problems into binary problems. In this paper, we obtained two main results: First, we showed that round robin class binarization may be used as an ensemble method and improve classification performance even for learning algorithms that are in principle capable of directly handling multi-class problems, in particular the decision tree algorithms of the c4.5 family. However, the observed improvements are not as significant as the improvements we have obtained in previous experiments for the ripper rule learning algorithm, and do in general not reach the same performance level as boosting and bagging. We also showed how a straight-forward extension of round robin learning (namely to perform multiple experiments for each binary problem) may improve over the performance of both its constituents, round robin and bagging. Despite the fact that they did not reach the performance levels of bagging and boosting, we believe that round robin ensembles have advantages that make them a viable alternative, most notably the clearly defined semantics of each member in the ensemble.

Second, we showed that the performance improvements of round robin ensembles increase with the number of classes in the problem. While the improvement over j48 grows approximately linearly with j48's error rate, the growth of the performance increase over one-against-all class binarization is even more dramatic. We believe that this illustrates that handling many classes is a major problem for the one-against-all binarization technique, possibly because the resulting binary learning problems have increasingly skewed class distributions. At the same time, we were unable to confirm our expectations that the relative efficiency of round robin learning should improve with a larger number of classes. This might be due to the fact that our previous theoretical results underestimated the effect of the constant overhead that has to be spent for each binary problem. Nevertheless, run-times are still comparable to those of regular c4.5, so that the accuracy gain provided by round robin classification comes at very low additional costs.

In addition, we also showed that round robin binarization is a valid alternative to learning from ordered classification. We repeated the experiments of Frank and Hall (2001) and found that round robin ensembles perform similar to the special-purpose technique that was suggested in their work.

The most pressing issue for further research is an investigation of the effects of different voting schemes. At the moment, we have only tried the simplest technique, unweighted voting where each classifiers may give one vote to one class. A further step ahead might be to weight each vote with a confidence estimate provided by the base classifier, or to allow a classifier only to vote for a class if it has a certain minimum confidence in its prediction. Several studies in various contexts have compared different voting techniques for combining the predictions of the individual classifiers of an ensemble (e.g., Mayoraz and Moreira, 1997; Allwein et al., 2000; Fürnkranz, 2002a). Although the final word on this issue remains to be spoken, it seems to be the case that techniques that include confidence estimates into the computation of the final predictions are in general preferable, and should be tried for round robin ensembles (cf. also Hastie and Tibshirani, 1998; Schapire and Singer, 1999).

Acknowledgments

I would like to thank Eibe Frank and Mark Hall for providing the regression datasets (which were originally collected by Luis Torgo), Richard Kirkby for providing his implementation of pairwise classification in Weka, and the maintainers of and contributors to the UCI collection of machine learning databases.

The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Education, Science and Culture. This work is supported by the Austrian *Fonds zur Förderung der Wissenschaftlichen Forschung (FWF)* under grant no. P12645-INF.

References

E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.

R. Anand, K. G. Mehrotra, C. K. Mohan, and S. Ranka. Efficient classification for multiclass problems using modular neural networks. *IEEE Transactions on Neural Networks*, 6:117–124, 1995.

C. L. Blake and C. J. Merz. UCI repository of machine learning databases. `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 1998. Department of Information and Computer Science, Unifirsty of California at Irvine, Irvine CA.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

P. Clark and R. Boswell. Rule induction with CN2: Some recent improvements. In *Proceedings of the 5th European Working Session on Learning (EWSL-91)*, pages 151–163, Porto, Portugal, 1991. Springer-Verlag.

P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3 (4):261–283, 1989.

W. W. Cohen. Fast effective rule induction. In A. Prieditis and S. Russell, editors, *Proceedings of the 12th International Conference on Machine Learning (ML-95)*, pages 115–123, Lake Tahoe, CA, 1995. Morgan Kaufmann.

W. W. Cohen and Y. Singer. A simple, fast, and effective rule learner. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99)*, pages 335–342, Menlo Park, CA, 1999. AAAI/MIT Press.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3): 273–297, 1995.

T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–158, 2000.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

E. Frank and M. Hall. A simple approach to ordinal classification. In L. D. Raedt and P. Flach, editors, *Proceedings of the 12th European Conference on Machine Learning (ECML-01)*, pages 145–156, Freiburg, Germany, 2001. Springer-Verlag.

Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

J. Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, February 1999.

J. Fürnkranz. Round robin rule learning. In C. E. Brodley and A. P. Danyluk, editors, *Proceedings of the 18th International Conference on Machine Learning (ICML-01)*, pages 146–153, Williamstown, MA, 2001. Morgan Kaufmann Publishers.

J. Fürnkranz. Hyperlink ensembles: A case study in hypertext classification. *Information Fusion*, 2002a. Special Issue on Fusion of Multiple Classifiers.

J. Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2:721–747, 2002b.

T. Hastie and R. Tibshirani. Classification by pairwise coupling. In M. Jordan, M. Kearns, and S. Solla, editors, *Advances in Neural Information Processing Systems 10 (NIPS-97)*, pages 507–513. MIT Press, 1998. URL `http://www-stat.stanford.edu/~hastie/Papers/2class.ps`.

C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 2002. To appear.

S. Knerr, L. Personnaz, and G. Dreyfus. Handwritten digit recognition by neural networks with single-layer training. *IEEE Transactions on Neural Networks*, 3(6):962–968, 1992.

U. H.-G. Kreßel. Pairwise classification and support vector machines. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, chapter 15, pages 255–268. MIT Press, Cambridge, MA, 1999.

A. Krieger, A. J. Wyner, and C. Long. Boosting noisy data. In C. E. Brodley and A. P. Danyluk, editors, *Proceedings of the 18th International Conference on Machine Learning (ICML-2001)*, pages 274–281, Williamstown, MA, 2001. Morgan Kaufmann Publishers.

E. Mayoraz and M. Moreira. On the decomposition of polychotomies into dichotomies. In *Proceedings of the 14th International Conference on Machine Learning (ICML-97)*, pages 219–226, Nashville, TN, 1997. Morgan Kaufmann.

D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

B. Pfahringer. Winning the KDD99 classification cup: Bagged boosting. *SIGKDD explorations*, 1(2):65–66, 2000.

J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12 (NIPS-99)*, pages 547–553. MIT Press, 2000. URL `http://lara.enm.bris.ac.uk/cig/gzipped/2000/dagsvm.ps.gz`.

D. Pyle. *Data Preparation for Data Mining*. Morgan Kaufmann, San Francisco, CA, 1999.

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

J. R. Quinlan. Bagging, boosting, and C4.5. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 725–730. AAAI/MIT Press, 1996.

R. L. Rivest. Learning decision lists. *Machine Learning*, 2:229–246, 1987.

R. E. Schapire. Using output codes to boost multiclass learning problems. In D. H. Fisher, editor, *Proceedings fo the 14th International Conference on Machine Learning (ICML-97)*, pages 313–321, Nachville, TN, 1997. Morgan Kaufmann.

R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

M. S. Schmidt and H. Gish. Speaker identification via support vector classifiers. In *Proceedings of the 21st IEEE International Conference Conference on Acoustics, Speech, and Signal Processing (ICASSP-96)*, pages 105–108, Atlanta, GA, 1996.

I. H. Witten and E. Frank. *Data Mining — Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers, 2000.