Improving Multiclass Pattern Recognition by the Combination of Two Strategies

Nicolás García-Pedrajas, *Member*, *IEEE*, and
Domingo Ortiz-Boyer, *Member*, *IEEE*

Abstract—We present a new method of multiclass classification based on the combination of one-vs-all method and a modification of one-vs-one method. This combination of one-vs-all and one-vs-one methods proposed enforces the strength of both methods. A study of the behavior of the two methods identifies some of the sources of their failure. The performance of a classifier can be improved if the two methods are combined in one, in such a way that the main sources of their failure are partially avoided.

Index Terms—Multiclass, classification, one-vs-one, one-vs-all, neural networks, support vector machines.

1 Introduction

A classification problem of K classes and n training observations consists of a set of patterns whose class membership is known. Let $S = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots (\boldsymbol{x}_n, y_n)\}$ be a set of n training samples where each instance $\boldsymbol{x}_i$ belongs to a domain $X \subset \mathbb{R}^m$. Each label is an integer from the set $Y = \{1, \ldots, K\}$. A multiclass classifier is a function $f : X \rightarrow Y$ that maps an instance $\boldsymbol{x}$ onto an element of Y.

The task is to find a definition for the unknown function, $f(\boldsymbol{x})$, given the set of training instances. Although many real-world problems are multiclass problems, $K > 2$, many of the most popular classifiers work best when facing binary problems, $K = 2$. Moreover, many algorithms are specifically designed for binary problems, such as Support Vector Machines (SVM). A class binarization is a mapping of a multiclass problem on several two-class problems in a way that allows a derivation of a prediction for the multiclass problem from the predictions of the two-class classifiers. The two-class classifier is usually referred to as *base learner*.

Among the proposed methods for approaching multiclass problems as many, possibly simpler, two-class problems, we can make a rough classification in three groups: one-vs-all, one-vs-one, and error correcting output codes based methods:

- **One-vs-All (OVA).** This method has been proposed independently by several authors [1], [2]. OVA method constructs K binary classifiers. Classifier ith, f_i, is trained using all the patterns of class i as positive instances and the patterns of the other classes as negative instances. An example is classified in the class whose corresponding classifier has the highest output. This classifier decision function, f, is defined as:

$$f(\boldsymbol{x}) = \arg \max_{j \in \{1, \ldots, K\}} f_j(\boldsymbol{x}).$$

- **One-vs-One (OVO).** This method, proposed in [3], constructs $K(K-1)/2$ classifiers [4]. Classifier ij, named f_{ij}, is trained using all the patterns from class i as positive instances, all the patterns from class j as negative instances, and disregarding the rest. There are different methods of combining the obtained classifiers, the most common is a simple voting scheme [5]. When classifying a new instance each one of the base classifiers casts a vote for one of the two classes used in its training.

  Another approach for the combination of the trained classifier is the *Decision Directed Acyclic Graph* (DDAG) [6]. This method constructs a rooted binary acyclic graph using the classifiers. The nodes are arranged in a triangle with the root node at the top, two nodes in the second layer, four in the third layer, and so on. To evaluate a DDAG on input pattern $\boldsymbol{x}$, starting at the root node the binary function is evaluated, and the next node visited depends upon the results of this evaluation. The final answer is the class assigned by the leaf node visited at the final step.
- **Error Correcting Output Codes (ECOC).** Dietterich and Bakiri [7] suggested the use of error correcting codes for multiclass classification. This method uses a matrix M of $\{-1, 1\}$ values of size $K \times F$, where F is the number of binary classifiers. The ith column of the matrix induces a partition of the classes into two *metaclasses*. Instance $\boldsymbol{x}$ belonging to class i is a positive instance for the jth classifier if and only if $M_{ij} = 1$. If we designate f_j as the sign of the jth classifier, the decision implemented by this method, $f(\boldsymbol{x})$, using the Hamming distance between each row of the matrix M and the output of the F classifiers is given by:

$$f(\boldsymbol{x}) = \arg \min_{r \in 1, \ldots, K} \sum_{i=1}^{F} \left(\frac{1 - \operatorname{sign}(M_{ri} f_i(\boldsymbol{x}))}{2} \right).$$

  There are other approaches that use other distance measures between the outputs of the classifiers and each row of the coding matrix, or more sophisticated methods [8].

  The usefulness of this approach relies heavily on the independence of the classifiers [9], without which the error correcting approach would fail. Rifkin and Klautau [10] suggested that if the binary classifiers are fine-tuned more accurately, the independence of the classifiers diminishes, and so does the efficiency of this approach.

Allwein et al. [11] proposed a unifying approach for the different methods using a coding matrix with three values, $\{-1, 0, 1\}$, with 0 meaning *don't care*. For example, in the one-vs-all approach, the coding matrix has K columns, all the diagonal elements set to 1, and all other elements set to -1. For one-vs-one, we have a matrix of $K \times \binom{K}{2}$, in which each column corresponds to a pair (c_1, c_2). For this column, the matrix has +1 in row c_1, -1 in row c_2, and zeros in all other rows.

Moreira and Mayoraz [12] developed a combination of different classifiers, considering the output of each one as a probability of the pattern of belonging to a certain class. This method needs the training of $K(K+1)/2$ classifiers.

The approach we propose is based on the idea that the combination of the methods one-vs-all and one-vs-one can be able to obtain a classifier that outperforms both methods separately. This belief is based on a study of these two methods when the classification they achieve for a given instance is inaccurate. This study is presented in the next section.

This paper is organized as follows: Section 2 describes our approach. Section 3 shows the experimental setup. Section 4 shows the results of the experiments carried out, and, finally, Section 5 states the conclusions of our work and future research lines.

• *The authors are with the Department of Computing and Numerical Analysis, Edificio Einstein, 3a Planta, Campus Universitario de Rabanales, 14071 Córdoba, Spain. E-mail: {npedrajas, dortiz}@uco.es.*

Manuscript received 27 May 2005; revised 19 Sept. 2005; accepted 21 Oct. 2005; published online 13 Apr. 2006.
Recommended for acceptance by L. Kuncheva.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0269-0505.

0162-8828/06/$20.00 © 2006 IEEE Published by the IEEE Computer Society

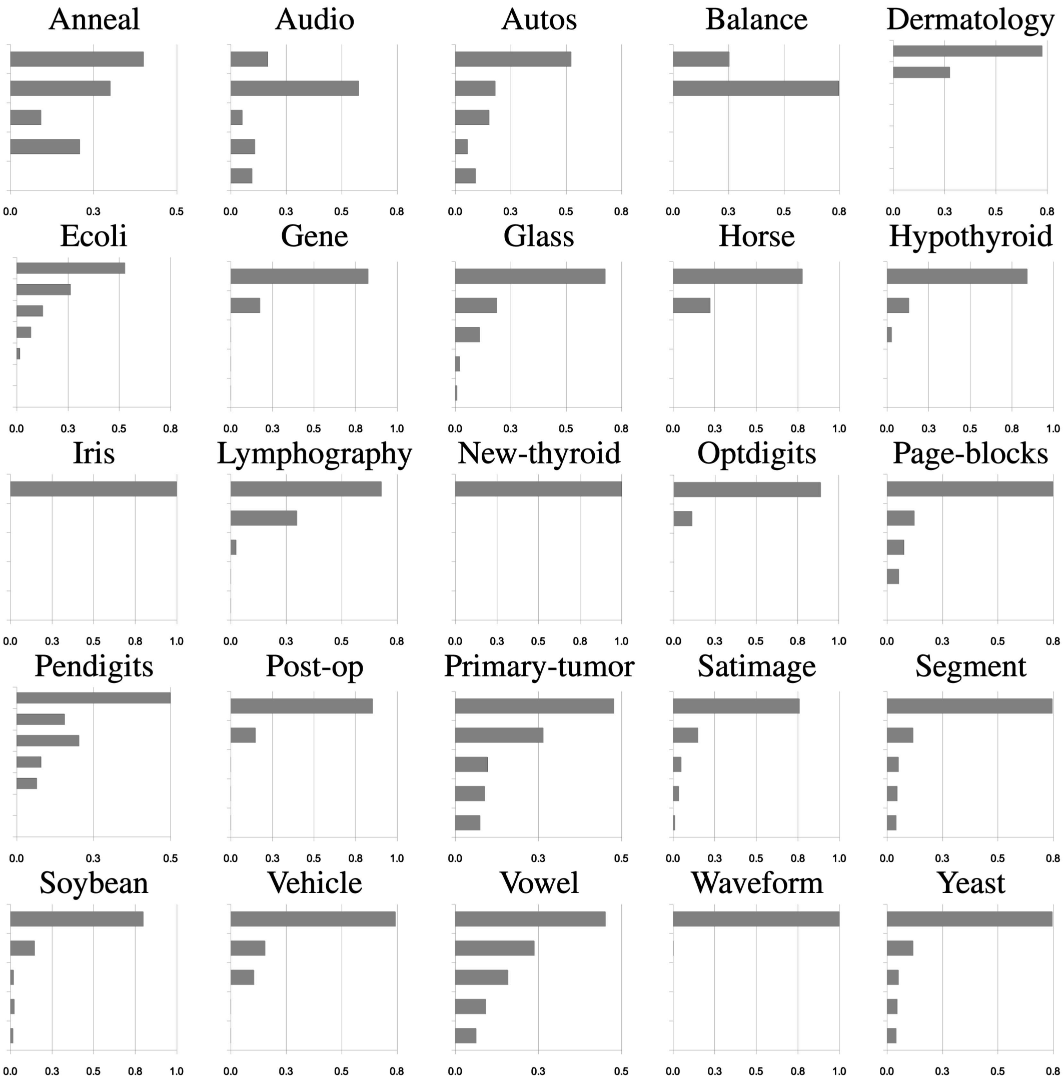

Fig. 1. Percentage of times when an instance is misclassified and the correct answer was, from top to bottom in each graphic, the second, third, fourth, fifth, and sixth highest output classifier in one-vs-all method. The results are averaged over 10 runs of the experiments for each data set.

2 Combination of One-vs-All and One-vs-One

In order to gain some insight into how one-vs-all and one-vs-one work, we have carried out a study of these two methods. A recent paper [10] claims that most of the reported differences in performance between one-vs-all and one-vs-one methods are due to poor tuning of parameters for one-vs-all method or the selection of too naive base learners. In this way, none of them is preferable to the other as a general rule, and a combination of both that tries to alleviate their weaknesses may yield to better performance.

The first experiment was intended to characterize the behavior of one-vs-all when its prediction of the class of a given instance was erroneous. If we are going to improve this classifier, we are interested in when and how it makes mistakes. The analysis studied the outputs of the K classifiers. Fig. 1 shows for each data set used (see Table 1 for a description of the data sets) and 10 runs of the method, the percentage of times that, provided the classification is inaccurate, the correct class corresponds to the second, third, fourth, fifth, and sixth highest output classifier. It is interesting to note that for a significant percentage, when the one-vs-all method fails to classify a pattern correctly, the second highest output identifies the correct output.

With regards to the one-vs-one method, it is interesting to note that many of the binary base classifiers are forced to give wrong answers for many instances, because each classifier must assign every pattern to one of two classes. If a pattern belongs to class i, all the classifiers that are not trained to differentiate this class will cast wrong votes. However, the votes of the classifiers that contain class i should be able to overrule these wrong votes. If the class values are independent, it is unlikely that many classifiers vote the same wrong class, but the probability of such situation increases if there are similarities among the classes [13]. We think that in real-world problems, these similarities are plausible. For instance, in a medical classification problem, some of the features of different diseases can be common to many patients, even if the class (different disease or different types of the same disease) to which they belong is not the same.

This problem was suggested by Geoffrey Hinton. Simulation results [4] show that this problem may be real. Moreover, it seems to be a problem inherent to the one-vs-one approach, and it is not clear whether any approach to this method could solve it. If the class values are not independent, and there is some similarity between the correct class and other class, the likelihood of a wrong classification would increase [13].

TABLE 1
Summary of Data Sets

Data set	Cases	Classes	Features			Inputs
			C	B	N	
Anneal	898	5	6	14	18	59
Audiology	226	24	–	61	8	93
Autos	205	6	15	4	6	72
Balance	625	3	4	–	–	4
Dermatology	366	6	1	1	32	34
Ecoli	336	8	7	–	–	7
Gene	3175	3	–	–	60	120
Glass	214	6	9	–	–	9
Horse	364	3	13	2	5	58
Hypothyroid	3772	4	7	20	2	29
Iris	150	3	4	–	–	4
Lymphography	148	4	–	9	6	38
New-thyroid	215	3	5	–	–	5
Optdigitis	5620	10	64	–	–	64
Page-blocks	5473	5	10	–	–	10
Pendigits	10992	10	16	–	–	16
Post-op	90	3	1	–	7	20
Primary-tumor	339	22	–	14	3	23
Satimage	6435	6	36	–	–	36
Segment	2310	7	19	–	–	19
Soybean	683	19	–	16	19	82
Vehicle	846	4	18	–	–	18
Vowel	990	11	10	–	–	10
Waveform	5000	3	40	–	–	40
Yeast	1484	10	8	–	–	10

The features of each data set can be C(continuous), B(binary), or N(nominal). The Inputs column shows the actual number of inputs of the classifier.

So, our second experiment was intended to characterize the main source of inaccurate classifications for one-vs-one method. Again, we focused on wrong answers. The experiment studied the behavior of the classifiers that tell the correct class of an instance apart from the other classes. Fig. 2 shows the percentage of accuracy of the classifiers trained using the correct class of the instance when the overall classification fails. That is, if an instance of class i is not correctly classified, we test all the classifiers $f_{ik}, 1 \leq k \leq K, k \neq i$, and report their percentage of accuracy. We see that the classification given by these classifiers is accurate to a high percentage, and that the cause of failing of the method is mostly the classifiers that cannot cast a correct vote for a given instance.

So, we can state that one of the main sources of the one-vs-one method failing to achieve a correct classification is the votes casted by the binary classifiers that have not been trained using the correct class of the instance. This is usually termed as the problem of *incompetent classifiers*.

Moreover, as Fig. 1 shows, a high percentage of the times when the one-vs-all approach makes a mistake, the correct answer is the second highest classifier. If we take the classifier with the two highest outputs, one of them gives the correct answer most of the times.

If we consider these two facts together, we can develop a classifier combining the predictions given by these two methods. This classifier should avoid the problems of the one-vs-one and one-vs-all approaches by means of the following features:

- The use of the one-vs-all classifier must not be limited to obtain the class with the highest output. Fig. 1 shows that the second highest output is the correct answer for a significant percentage of patterns.
- The classifiers obtained by the one-vs-one method are highly accurate, even when the overall classification of the method is erroneous. However, the accuracy of the method is undermined by the use of classifiers that cannot cast a correct vote, as they have not been trained using the class of the given pattern. So, our method should only use one of these classifiers when there is a high probability of distinguishing the correct class of the instance.

With these two main ideas in mind, we designed the approach that is presented in this paper. As we have stated, we consider multiclass classification problem of K classes, $K > 2$. The proposed method is named *all-and-one* (A&O) as it is the combination of the one-vs-all and one-vs-one methods. A&O works as follows:

1. Train $f_i, i = 1, \ldots, K$, classifiers using for training f_k classifier the instances of class k as positive ones and the rest of the instances as negative ones (as in OVA).
2. Train $f_{ij}, i = 2, \ldots, K, j \leq i$, classifiers using for training f_{kl} classifier the instances of class k as positive ones and the instances of class l as negative ones (as in OVO).
3. For classifying an unknown pattern x evaluate $f_i, i = 1, \ldots, K$ and obtain the two classes, c_1 and c_2, whose corresponding classifiers, f_{c_1} and f_{c_2}, have the two highest values. Evaluate classifier $f_{c_1c_2}$ and assign the pattern to c_1 or c_2 accordingly.

The main drawback of this approach is the necessity of training more classifiers. If we have K classes, we need to train $K(K+1)/2$, against the $K(K-1)/2$ of the one-vs-one method, or the K of the one-vs-all method. However, once the classifier has been trained, the testing step only needs to test $K+1$ classifiers, against the $K(K-1)/2$ of the one-vs-one approach, or the K classifiers of one-vs-all.

Nevertheless, the actual number of classifiers that we must train has been greatly reduced with a modification to the standard one-vs-one approach. The binary classifiers are trained "*on demand*." Only the classifiers that are needed to tell apart two classes, as a result of the first classification with the one-vs-all classifiers, are created. The reduction in the number of classifiers is significant, specially in the problems with a large number of classes. If, in the test step, we must decide between two classes and the classifier that separate them has not been created, we apply the standard one-vs-all classifier.[1]

In the next two sections, we evaluate the soundness of this approach empirically in a wide variety of real-world problems. We think that in the absence of a complete theory, that unfortunately cannot always be devised for real-world problems where underlying distributions of instances and noise are not known, empirical results provide useful knowledge of the performance of the different methods.

3 Experimental Setup

In order to get a clear idea of the relative performance of the proposed model, we chose the 25 data sets from the UCI Machine Learning Repository [14] summarized in Table 1.

The parameters of the algorithms are common to all the performed experiments. We carried out two sets of experiments using as base learners a multilayer perceptron (MLP) with one hidden layer of 10 nodes and hyperbolic tangent transfer function and a Support Vector Machine (SVM) with a Gaussian kernel. For the MLP, the learning algorithm was a standard back-propagation with a learning rate $\eta = 0.15$ and a momentum coefficient $\alpha = 0.1$. The network was trained for 100,000 iterations of the back-prop algorithm. For the SVM, we used a value of $\gamma = 0.1$ and $C = 10$.[2]

[1] In practice, the frequency of happening of such a case is extremely low.

[2] For some of the data sets, these parameters obtained very bad results with all the methods. For these data sets, the values of γ and C were manually adjusted.

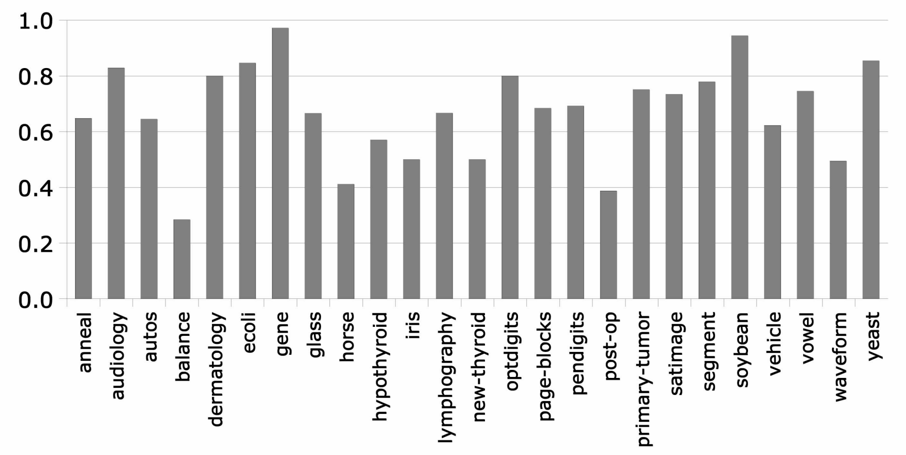

Fig. 2. Percentage of times when an instance is misclassified and the classifiers for the correct class give accurate answers in the one-vs-one method. The results are averaged over 10 runs of the experiments for each data set.

TABLE 2
Test Error Rates and Number of Classifiers Using the Six Compared Methods and a Neural Network as Base Learner

Dataset	Test error						Number of classifiers		
	OVO	**OVA**	**DDAG**	**ECOC 30**	**ECOC 50**	**A&O**	**OVO**	**OVA**	**A&0**
Anneal	0.0198	0.0142	0.0202	0.0112	0.0109	0.0168	10	5	9.7
Audiology	0.1864	0.1864	0.1955	0.1924	0.1955	0.1727	276	24	81.8
Autos	0.2350	0.2500	0.2450	0.2683	0.2600	0.2450	15	6	14.8
Balance	0.0597	0.0774	0.0629	0.0763	0.0645	0.0613	3	3	3.0
Dermatology	0.0306	0.0407	0.0306	0.0333	0.0361	0.0296	15	6	15.0
Ecoli	0.1576	0.1454	0.1591	0.1485	0.1374	0.1485	28	8	19.1
Gene	0.1274	0.1294	0.1263	0.1464	0.1407	0.1232	3	3	3.0
Glass	0.2873	0.3032	0.2889	0.3064	0.3143	0.2810	15	6	14.5
Horse	0.3667	0.3750	0.3556	0.3620	0.3306	0.3389	3	3	3.0
Hypothyroid	0.0279	0.0281	0.0276	0.0291	0.0271	0.0249	6	4	6.0
Iris	0.0489	0.0578	0.0489	0.0711	0.0667	0.0489	3	3	3.0
Lymphography	0.1536	0.1607	0.1643	0.1690	0.1571	0.1429	6	4	5.5
New-thyroid	0.0429	0.0333	0.0429	0.0460	0.0413	0.0429	3	3	3.0
Optdigits	0.0246	0.0308	0.0247	0.0300	0.0245	0.0240	45	10	44.4
Page-blocks	0.0374	0.0384	0.0379	0.0348	0.0339	0.0358	10	5	9.8
Pendigits	0.0093	0.0177	0.0095	0.0170	0.0095	0.0117	45	10	44.8
Post-op	0.4778	0.4630	0.4741	0.4926	0.4889	0.4556	3	3	3.0
Primary-tumor	0.5768	0.6243	0.5879	0.5899	0.5839	0.5970	231	22	98.2
Satimage	0.1191	0.1177	0.1127	0.1162	0.1049	0.1109	15	6	15.0
Segment	0.0749	0.1043	0.0797	0.0876	0.0900	0.0830	21	7	13.0
Soybean	0.0603	0.0706	0.0632	0.0622	0.0549	0.0514	171	19	97.8
Vehicle	0.2024	0.2095	0.1988	0.1945	0.1901	0.1929	6	4	6.0
Vowel	0.2926	0.4030	0.2855	0.3720	0.3000	0.3178	55	11	55.0
Waveform	0.1662	0.1484	0.1662	0.1537	0.1601	0.1662	3	3	3.0
Yeast	0.4313	0.4396	0.4412	0.3932	0.3849	0.4214	45	10	25.5
Mean error	0.1686	0.1788	0.1700	0.1762	0.1683	0.1658			
Standard deviation	0.1577	0.1666	0.1586	0.1605	0.1562	0.1575			

The number of classifiers of DDAG is the same as OVO.

The source code in C of all the reported algorithms is available upon request to the authors.

The generalization error was estimated using 10-fold cross-validation. OVO, OVA, DDAG, and A&O methods used the same set of base classifiers for each repetition of the experiments. That is, we train K classifiers using the method one-vs-all, and $K(K-1)/2$ classifiers using the method one-vs-one. These two sets of classifiers are then combined using the methods OVO, OVA, DDAG, and A&O.

The ECOC approach cannot share these classifiers due to its different definition. We used a random coding of 30 and 50 bits per class, with approximately equal random split. We have used this approach as in several previous works [7], [15], [8] random codes performed as well or better than codes designed for its error-correcting properties. The class of the pattern is chosen using the L1-norm, the most common distance measure for ECOC.

The use of t-tests, or other pairwise tests, for the comparison of several classification methods has been criticized in several papers [16]. The t-tests can provide an accurate evaluation of the probability of obtaining the observes outcomes by chance, but it has limited ability to predict relative performance even on further data set instances from the same domain, let alone in other domains. Moreover, as more data sets and classification algorithms are used, the probability of type I error happening increases dramatically.

To avoid these problems, the comparisons between the performance of the different methods were made following the comparison design of Webb [17]. We perform a single significance

TABLE 3
Test Error Rates and Number of Classifiers Using the Six Compared Methods and a SVM as Base Learner

Dataset	Test error						Number of classifiers		
	OVO	**OVA**	**DDAG**	**ECOC 30**	**ECOC 50**	**A&O**	**OVO**	**OVA**	**A&0**
Anneal	0.0090	0.0090	0.0090	0.0090	0.0090	0.0090	10	5	10.0
Audiology	0.1636	0.1636	0.1636	0.1636	0.1637	0.1591	276	24	84.8
Autos	0.2250	0.3250	0.2250	0.2950	0.2950	0.2650	15	6	14.5
Balance	0.0919	0.0919	0.0919	0.0936	0.0919	0.0919	3	3	3.0
Dermatology	0.0417	0.0417	0.0417	0.0417	0.0417	0.0417	15	6	0.0
Ecoli	0.1606	0.1606	0.1606	0.1545	0.1545	0.1576	28	8	26.3
Gene	0.1076	0.1076	0.1076	0.1120	0.1142	0.1076	3	3	3.0
Glass	0.2810	0.2810	0.2667	0.2762	0.2857	0.2572	15	6	15.0
Horse	0.3556	0.3611	0.3639	0.3583	0.3472	0.3528	3	3	3.0
Hypothyroid	0.0390	0.0395	0.0400	0.0385	0.0392	0.0387	6	4	6.0
Iris	0.0400	0.0533	0.0400	0.0496	0.0533	0.0400	3	3	3.0
Lymphography	0.1357	0.1357	0.1357	0.1357	0.1357	0.1357	6	4	5.7
New-thyroid	0.0333	0.0333	0.0333	0.0333	0.0333	0.0333	3	3	3.0
Optdigits	0.0185	0.0181	0.0185	0.0192	0.0185	0.0185	45	10	45.0
Page-blocks	0.0320	0.0331	0.0318	0.0331	0.0333	0.0316	10	5	10.0
Pendigits	0.0033	0.0037	0.0034	0.0042	0.0038	0.0033	45	10	45.0
Post-op	0.4000	0.4111	0.4000	0.4000	0.4111	0.4000	3	3	3.0
Primary-tumor	0.5940	0.5728	0.5970	0.5879	0.6000	0.5879	231	22	117.1
Satimage	0.0746	0.0775	0.0740	0.0778	0.0786	0.0749	15	6	15.0
Segment	0.0277	0.0294	0.0277	0.0307	0.0303	0.0273	21	7	21.0
Soybean	0.0515	0.0456	0.0500	0.0500	0.0500	0.0470	171	19	120.7
Vehicle	0.1786	0.1703	0.1786	0.1715	0.1703	0.1774	6	4	6.0
Vowel	0.2232	0.2687	0.2101	0.2828	0.2767	0.2182	55	11	38.8
Waveform	0.1396	0.1408	0.1396	0.1408	0.1412	0.1396	3	3	3.0
Yeast	0.4284	0.4764	0.4297	0.4703	0.4507	0.4264	45	10	39.6
Mean error	0.1542	0.1620	0.1536	0.1612	0.1612	0.1537			
Standard deviation	0.1536	0.1597	0.1539	0.1590	0.1590	0.1529			

The number of classifiers of DDAG is the same as OVO.

test for every pair of algorithms. This test is a sign test on the win/draw/loss record of the two algorithms across all data sets.

In all the tables, the p-values of the corresponding tests are shown. The error measure is given by $E = \frac{1}{P}\sum_{i=1}^{P} e_i$, where e_i is 1 if pattern i is misclassified and 0 otherwise. In all the tables, we show this error measure.

4 Experimental Results

The generalization error of the six methods for the 25 data sets is shown in Table 2 for the neural network and in Table 3 for the SVM. In a first approach, it is interesting to note that all the methods present a similar standard deviation.

It is also interesting to note that the number of classifiers needed by the A&O method is slightly larger than the OVO method for problems with fewer classes, and significantly smaller in problems with a larger number of classes. In this way, A&O needs to train fewer classifiers when it is applied to problems with many classes.

Tables 4 and 5 show the comparison of the different models as explained above for the neural network and the SVM, respectively. As we have stated, our major comparative descriptive statistic is the win/draw/loss record. In the table, the win/draw/loss record is labeled s. The first value is the number of data sets for which the algorithm of the corresponding column performs better than the algorithm of the corresponding row (*win* record), the second value is the number of data sets for which the two algorithms have the same error (*draw* record), and the third value is the number of data sets for which the algorihtm of the corresponding column performs worse than the algorithm of the corresponding row (*loss* record). The row labeled p is the result of the two-tailed sign test on the win-loss record.

The tables also present the mean of errors across all data sets. This is a very gross indication of the relative performance. Nevertheless, a low mean error can be considered indicative of a tendency toward low error rates for individual domains. We have also used another comparative statistic, the *geometric mean error ratio* of every pair of algorithms. For two algorithms a_1, with errors $e_1^1, e_2^1, \ldots, e_n^1$, and a_2, with errors $e_1^2, e_2^2, \ldots, e_n^2$ for n data sets, the geometric mean of the error ratios is $\dot{r} = \left(\prod_{i=1}^{n} e_i^1/e_i^2\right)^{1/n}$.

The row labeled $\dot{r}$ shows the geometric mean of the error ratio *column/row*. A value below 1 indicates a general advantage of the algorithm corresponding to the column to the algorithm corresponding to the row.

TABLE 4
Comparison of the Six Methods Using a Neural Network as Base Learner

Algorithm		OVO	OVA	DDAG	ECOC 30	ECOC 50	A&O
Mean		0.1686	0.1788	0.1700	0.1762	0.1683	0.1658
OVO	s		6/1/18	7/4/14	8/0/17	12/0/13	16/3/6
	p		0.0227	0.1892	0.1078	1.0000	0.0525
	$\dot{r}$		1.0810	1.0113	1.0674	0.9895	0.9788
OVA	s			17/0/8	14/0/11	17/0/8	21/0/4
	p			0.1078	0.6900	0.1078	0.0009
	$\dot{r}$			0.9356	0.9874	0.9154	0.9055
DDAG	s				8/0/17	14/1/10	17/4/4
	p				0.1078	0.5413	0.0072
	$\dot{r}$				1.0554	0.9784	0.9678
ECOC 30	s					20/0/5	19/1/5
	p					0.0041	0.0066
	$\dot{r}$					0.9270	0.9170
ECOC 50	s						13/0/12
	p						1.0000
	$\dot{r}$						0.9892

Win/draw/loss record (row s) of the algorithms against each other and p-value of the sign test (row p), and the geometric mean of the error ratio (row $\dot{r}$).

TABLE 5
Comparison of the Six Methods Using a SVM as the Base Learner

Algorithm		**OVO**	**OVA**	**DDAG**	**ECOC 30**	**ECOC 50**	**A&O**
Mean		0.1542	0.1620	0.1536	0.1612	0.1612	0.1537
OVO	s		5/8/12	5/15/5	7/5/13	4/6/15	13/10/2
	p		0.1435	1.0000	0.4807	0.4807	0.0074
	$\dot{r}$		1.0418	0.9973	1.0483	1.0461	0.9932
OVA	s			11/7/7	8/6/11	5/8/12	15/6/4
	p			0.4807	0.6476	0.1435	0.0192
	$\dot{r}$			0.9572	1.0062	1.0041	0.9534
DDAG	s				6/6/13	4/7/14	12/9/4
	p				0.1671	0.0309	0.0768
	$\dot{r}$				1.0512	1.0490	0.9985
ECOC 30	s					9/6/10	16/6/3
	p					1.0000	0.0044
	$\dot{r}$					0.9979	0.9475
ECOC 50	s						16/5/4
	p						0.0118
	$\dot{r}$						0.9494

Win/draw/loss record (row s) of the algorithms against each other and p-value of the sign test (row p), and the geometric mean of the error ratio (row $\dot{r}$).

The comparison of the combined approach to the other methods shows that this approach is able to outperform all the other algorithms with a confidence level of 10 percent for the two types of base classifiers, with the exception of ECOC with 50 classifiers and a neural network as base learner. These results are a powerful argument in favor of the combined approach, due to the number of data sets and the variety of features of these data sets.

Considering the geometric mean of the test error ratio, A&O also performs better, as the table shows values of $\dot{r}$ below 1 for all the comparisons; although, as we have said, this measure can only be considered to give a general idea of the tendencies of the relative performance of the classifiers.

It is also interesting to note the different behavior of ECOC when the base classifier is a neural network or an SVM. The performance of ECOC with a SVM is the same for 30 and 50 classifiers. On the other hand, the performance of ECOC with a neural network is significantly better when 50 classifiers are used. The reason might be that the randomness neural networks introduced is able to obtain more benefits from the redundancy of the ECOC coding.

5 Conclusions and Future Work

In this paper, we have presented a method that combines the one-vs-all and one-vs-one approaches to multiclass classification to improve the results of both methods. In learning time, we need to train more classifiers than any of the two methods for problems with few classes, but for problems with many classes, the number of classifiers is reduced when compared with the OVO approach, although the global complexity may be comparable, due to the fact that OVO classifiers have a run-time complexity below that of OVA classifiers [13]. In testing time, the proposed method needs fewer evaluations than one-vs-one, and only one more evaluation than one-vs-all.

The proposed method has been applied to a wide range of different classification problems, showing better overall performance when compared with the widely used methods one-vs-one, one-vs-all, ddag, and ECOC using two different base classifiers. This improved performance is statistically significant with a confidence level of 10 percent, for all but one of the methods tested.

For future work, we think that this paper opens a field for other approaches that use the combination of the information given for different approaches, such as one-vs-one and one-vs-all. Our study has shown that some of the weaknesses of different methods can be avoided when they are combined.

References

[1] P. Clark and R. Boswell, “Rule Induction with CN2: Some Recent Improvements,” *Proc. Fifth European Working Session on Learning (EWSL-91),* pp. 151-163, 1991.

[2] R. Anand, K.G. Mehrotra, C.K. Mohan, and S. Ranka, “Efficient Classification for Multiclass Problems Using Modular Neural Networks,” *IEEE Trans. Neural Networks,* vol. 6, pp. 117-124, 1995.

[3] S. Knerr, L. Personnaz, and G. Dreyfus, “Single-Layer Learning Revisited: A Stepwise Procedure for Building and Training a Neural Network,” *Neurocomputing: Algorithms, Architectures, and Applications,* J. Fogelman, ed., New York: Springer-Verlag, 1990.

[4] T. Hastie and R. Tibshirani, “Classification by Pairwise Coupling,” *The Annals of Statistics,* vol. 26, no. 2, pp. 451-471, 1998.

[5] J. Friedman, “Another Approach to Polychotomous Classification,” technical report, Dept. of Statistics, Stanford Univ., 1996.

[6] J.C. Platt, N. Christiani, and J. Shawe-Taylor, “Large Margin DAGs for Multiclass Classification,” *Proc. Neural Information Processing Systems (NIPS ’99),* S.A. Solla, T.K. Leen, and K.-R. Müller, eds., pp. 547-553, 1999.

[7] T.G. Dietterich and G. Bakiri, “Solving Multiclass Learning Problems via Error-Correcting Output Codes,” *J. Artificial Intelligence Research,* vol. 2, pp. 263-286, 1995.

[8] T. Windeatt and R. Ghaderi, “Coding and Decoding Strategies for Multi-Class Learning Problems,” *Information Fusion,* vol. 4, pp. 11-21, 2003.

[9] E.B. Kong and T.G. Dietterich, “Why Error-Correcting Output Coding Works with Decision Trees,” technical report, Dept. of Computer Science, Oregon State Univ., Corvallis, 1995.

[10] R. Rifkin and A. Klautau, “In Defense of One-vs-All Classification,” *J. Machine Learning Research,* vol. 5, pp. 101-141, 2004.

[11] E.L. Allwein, R.E. Schapire, and Y. Singer, “Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers,” *J. Machine Learning Research,* vol. 1, pp. 113-141, 2000.

[12] M. Moreira and E. Mayoraz, “Improved Pairwise Coupling Classification with Correcting Classifiers,” *Proc. 10th European Conf. Machine Learning (ECML ’98),* Apr. 1997.

[13] J. Fürnkranz, “Round Robin Classification,” *J. Machine Learning Research,* vol. 2, pp. 721-747, 2002.

[14] S. Hettich, C.L. Blake, and C.J. Merz, “UCI Repository of Machine Learning Databases,” http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.

[15] R.E. Schapire, “Using Output Codes to Boost Multiclass Learning Problems,” *Proc. 14th Int’l Conf. Machine Learning,* pp. 313-321, 1997.

[16] T.G. Dietterich, “Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms,” *Neural Computation,* vol. 10, no. 7, pp. 1895-1923, 1998.

[17] G.I. Webb, “Multiboosting: A Technique for Combining Boosting and Wagging,” *Machine Learning,* vol. 40, no. 2, pp. 159-196, Aug. 2000.