Available online at www.sciencedirect.com

SCIENCE DIRECT®

Pattern Recognition Letters 26 (2005) 83–90

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Using diversity measures for generating error-correcting output codes in classifier ensembles

Ludmila I. Kuncheva [*]

School of Informatics, University of Wales Bangor, Dean Street, Bangor, Gwynedd LL57 1UT, United Kingdom

Received 5 July 2004
Available online 14 October 2004

Abstract

Error-correcting output codes (ECOC) are used to design diverse classifier ensembles. Diversity within ECOC is traditionally measured by Hamming distance. Here we argue that this measure is insufficient for assessing the quality of code for the purposes of building accurate ensembles. We propose to use diversity measures from the literature on classifier ensembles and suggest an evolutionary algorithm to construct the code.
© 2004 Elsevier B.V. All rights reserved.

Keywords: Statistical pattern recognition; Classifier ensembles; Diversity measures; Error-correcting output codes (ECOC); Evolutionary algorithms

1. Introduction

ECOC are developed for pattern recognition problems with multiple classes (Aha and Blankert, 1997; Dietterich and Bakiri, 1991, 1995; Masulli and Valentini, 2000a,b; Shapire, 1997; Windeatt and Ghaderi, 2001, 2003). The idea is to avoid solving the multiclass problem directly and to break it into dichotomies instead. Each classifier in the ensemble discriminates between two (possibly compound) classes. Consider an example where $\Omega = \{\omega_1, \ldots, \omega_{10}\}$ is the set of class labels. We can break Ω into $\Omega = \{\Omega^{(1)}, \Omega^{(0)}\}$ where $\Omega^{(1)} = \{\omega_1, \ldots, \omega_5\}$ and $\Omega^{(0)} = \{\omega_6, \ldots, \omega_{10}\}$, called *a dichotomy*. Discriminating between $\Omega^{(1)}$ and $\Omega^{(0)}$ will be the task of one of the classifiers in the ensemble. Each classifier is assigned a different dichotomy.

Diversity between the classifiers is a highly desirable characteristic of the ensemble. The presumption in using ECOC is that diverse classifiers are obtained from diverse dichotomies. While minimum Hamming distance is the traditional measure for diversity in ECOC, in this paper we propose to use diversity measures originally

[*] Tel.: +44 1248 383661; fax: +44 1248 361429.
E-mail address: l.i.kuncheva@bangor.ac.uk

0167-8655/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2004.08.019

devised for classifier outputs. Minimum Hamming distance guarantees the error-correcting capability of the code. However, we may wish to compromise on this guarantee in order to get a more diverse ensemble *on the average*, as explained later by an example. This idea brings in diversity measures used in classifier combination (Kuncheva and Whitaker, 2003).

The paper is organized as follows. Section 2 explains ECOC and gives some code generating methods. While minimum Hamming distance is a traditional measure for the error-correcting quality of a code, Section 3 looks into the need for another measure of the quality of the code when used for classifier ensembles. Section 4 suggests an application of diversity measures to evaluating ECOC. Section 5 proposes an evolutionary algorithm for ECOC construction, using a diversity measure as its fitness function. Section 6 gives our comments and conclusions.

2. Error-correcting output codes (ECOC)

Let $\Omega = \{\omega_1, \ldots, \omega_c\}$ be a set of class labels. Suppose that each classifier codes the respective compound class $\Omega^{(1)}$ as 1 and compound class $\Omega^{(0)}$ as 0. Then every class ω_j, $j = 1, \ldots, c$, will have a binary "profile" or a *codeword*. For example, suppose that there are 5 classifiers. A possible class profile (codeword) for ω_1 is $[0, 1, 1, 0, 1]^T$. This means that ω_1 is in the respective $\Omega^{(1)}$ sets for classifiers D_2, D_3, and D_5 and in the respective $\Omega^{(0)}$ sets for classifiers D_1 and D_4.

2.1. The code matrix

We can represent each dichotomy as a binary vector of length c with 1's for the classes in $\Omega^{(1)}$ and 0's for the classes in $\Omega^{(0)}$. The set of all such vectors has 2^c elements. However, not all of them correspond to different splits. Consider $[0, 1, 1, 0, 1]^T$ and $[1, 0, 0, 1, 0]^T$. Even though the Hamming distance between the two binary vectors is equal to the maximum possible value, 5, the two subsets are identical, only with swapped labels. Since there are two copies of each split within the total of 2^c splits, the number of different splits is $2^{(c-1)}$. The splits $\{\Omega, \emptyset\}$ and the corresponding $\{\emptyset, \Omega\}$ are of no use because they do not represent any discrimination task. Therefore the number of possible *different* splits of a set of c class labels into two non-empty disjoint subsets (dichotomies) is $2^{(c-1)} - 1$.

Let L be the chosen number of classifiers in the ensemble. The class assignments for the ensemble (the dichotomies) can be represented as a binary *code matrix* C of size $c \times L$. The (i, j)th entry of C, denoted $C(i, j)$ is 1 if class ω_i is in $\Omega_j^{(1)}$ or 0, if class ω_i is in $\Omega_j^{(0)}$. Thus each row of the code matrix is a codeword and each column is a classifier assignment. An example of a code matrix for $c = 4$ classes with all possible $2^{(4-1)} - 1 = 7$ different dichotomies is shown in Table 1.

Let $[s_1, \ldots, s_L]$, $s_i \in \{0, 1\}$, be the binary output of the L classifiers in the ensemble for a given input $\mathbf{x}$. The Hamming distance between the classifier outputs and the codewords for the classes is calculated as $\sum_{i=1}^{L} |s_i - C(j, i)|$. In the standard set-up the input is labeled in the class with the smallest distance (decoding phase). Ties are broken randomly. A more sophisticated decoding strategies are discussed by Windeatt and Ghaderi (2001, 2003).

To take the most advantage of an ECOC ensemble, the code matrix should be built according to two main criteria.

Row separation. In order to avoid misclassifications, the codewords should be as far apart from one another as possible. We can still recover the correct label for $\mathbf{x}$ even if several classifiers have guessed wrongly. A measure of the quality of an error-correcting code is the minimum Hamming distance, H_c, between any pair of codewords. The number of errors that the code is guaranteed to be able to correct is $\lfloor \frac{H_c - 1}{2} \rfloor$. (Here $\lfloor a \rfloor$ denotes the "floor", i.e., the nearest integer smaller than a.)

Table 1
Exhaustive ECOC for $c = 4$ classes ($L = 7$ classifiers)

	D_1	D_2	D_3	D_4	D_5	D_6	D_7
ω_1	0	0	0	1	0	1	1
ω_2	0	0	1	0	0	0	0
ω_3	0	1	0	0	1	0	1
ω_4	1	0	0	0	1	1	0

Column separation. It is important that the dichotomies given as the assignments to the ensemble members are as different from each other as possible too. This will drive the ensemble towards low correlation between the classification errors which will hopefully increase the ensemble accuracy (Dietterich and Bakiri, 1995). The distance between the columns must be maximized keeping in mind that the complement of a column gives the same split of the set of classes. Therefore, the column separation should be sought by maximizing

$$H_L = \min_{i,j,i\neq j} \min \left\{ \sum_{k=1}^{c} |C(k,i) - C(k,j)|, \sum_{k=1}^{c} |1 - C(k,i) - C(k,j)| \right\}, \quad i,j \in \{1,2,\dots,L\}. \tag{1}$$

2.2. ECOC generation methods

One-per-class. The standard ECOC is the so called "one-per-class" code. This encoding is commonly used as the target output for training neural network classifiers for multiple classes. The target output for class ω_j is a codeword with c elements, containing 1 at position j and 0's elsewhere. Thus the code matrix is the identity matrix of size c and we only build $L = c$ classifiers. This encoding is of low quality because the Hamming distance between any two rows is 2, and so the error-correcting power is $\lfloor \frac{2-1}{2} \rfloor = 0$.

All pairs. In this model every pair of classes is taken as $\Omega^{(1)}$ and the remaining $c - 2$ classes form $\Omega^{(0)}$.

There are $L = \frac{c(c-1)}{2}$ classifiers (columns). [1] Consider the codeword for class ω_i. There will be $c - 1$ ones, while all the remaining positions will be zeros, because class ω_i appears exactly in $c - 1$ pairs. Every pair of codewords share one and only one 1, that is, only one classifier. Suppose that codewords C_1 and C_2 shared 2 classifiers, D_1 and D_2. This means that classifier D_1 has both ω_1 and ω_2 in its $\Omega^{(1)}$ set, and so does classifier D_2. Since we assumed that every pair of classes forms the $\Omega^{(1)}$ set for one and only one classifier, having D_1 and D_2 with the same $\Omega^{(1)}$ set contradicts our construction assumption. On the other hand, the two codewords cannot be void of a common classifier (a 1 at the same position) because if this is the case, the pair of classes whose codewords we are considering will have no dedicated classifier. This again contradicts our set-up. Thus any two codewords mismatch at the $c - 2$ ones in codeword C_1 not shared by codeword C_2 and also at the $c - 2$ ones in codeword C_2 not shared by codeword C_1. Since this argument applies to any pair of codewords, the minimum Hamming distance across the whole code is $2(c - 2)$. Therefore the power of the 'all pairs' code is $\lfloor \frac{2(c-2)-1}{2} \rfloor = c - 3$.

Both one-per-class and all-pairs codes are *equidistant* because the Hamming distances between every pair of codewords are the same (Windeatt and Ghaderi, 2003).

Exhaustive codes. Dietterich and Bakiri (1995) give the following procedure for generating all possible $2^{(c-1)} - 1$ different classifier assignments for c classes. They suggest that exhaustive codes should be used for $3 \leqslant c \leqslant 7$.

(1) Row 1 is all ones.
(2) Row 2 consists of $2^{(c-2)}$ zeros followed by $2^{(c-2)} - 1$ ones.
(3) Row 3 consists of $2^{(c-3)}$ zeros, followed by $2^{(c-3)}$ ones, followed by $2^{(c-3)}$ zeros, followed by $2^{(c-3)} - 1$ ones.
(4) In row i, there are alternating $2^{(c-i)}$ zeros and ones.
(5) The last row is 0, 1, 0, 1, 0, 1, ..., 0.

The exhaustive code for $c = 4$ obtained through this procedure is given in Table 2. Note that Table 2 presents the same ensemble as Table 1. Columns 1, 2, 3 and 5 of Table 1 (the classifier assignments) have the 0's and the 1's swapped, thus keeping the

Table 2
Exhaustive ECOC for $c = 4$ classes ($L = 7$ classifiers)

	D_1	D_2	D_3	D_4	D_5	D_6	D_7
ω_1	1	1	1	1	1	1	1
ω_2	0	0	0	0	1	1	1
ω_3	0	0	1	1	0	0	1
ω_4	0	1	0	1	0	1	0

[1] Valid for $c > 4$.

same dichotomies, and the columns are permuted as [4, 6, 7, 3, 5, 2, 1]. [2]

For $8 \leqslant c \leqslant 11$ Dietterich and Bakiri (1995) suggest to select columns from the exhaustive code by an optimization procedure. Note that for $c = 3$ the exhaustive code will be the same as the one-per-class code which shows that problems with a small number of classes might not benefit from the ECOC approach. For values of c larger than 11, random code generation is recommended.

Random generation. Authors of studies on ECOC ensembles share the opinion that random generation of the codewords is a reasonably good method (Dietterich and Bakiri, 1995; Shapire, 1997; Windeatt and Ghaderi, 2003). Although these studies admit that more sophisticated procedures might lead to better codes, they also suggest that the improvement in the code might have only marginal effect on the ensemble accuracy.

3. Why is minimum Hamming distance insufficient for ECOC classifier ensembles?

Minimum Hamming distance looks into the worst-case scenario. It *guarantees* that a given amount of errors can be corrected. This makes the minimum Hamming distance an attractive measure for deriving bounds on the error. As noted by Windeatt and Ghaderi (2003), high minimum distance between any pair of codewords implies a reduced bound on the generalization error. Maximizing the minimum Hamming distance will work particularly well for equidistant codes. However, sometimes reducing bounds on the error, which may be quite loose to start with, is not very practical. We may wish to design a code which is allowed to fail occasionally in recovering the true class label for a small number of objects but which *on average* will perform better than a code with a larger minimum Hamming distance.

To illustrate this point, consider a problem with 5 classes and 5 classifiers. Two code matrices are shown in Fig. 1. The minimum and average H_c are also displayed.

The matrices with the pairwise Hamming distances between the codewords (rows of the code matrices), H_c, are given underneath. The minimum H_c for ensemble 1 is 2, and the minimum H_c for ensemble 2 is 1. According to the maximum $\min H_c$ criterion, we will prefer ensemble 1 to ensemble 2. A simulation was run to estimate classification accuracies of the two ensembles under the following assumptions: Each of the 5 classes comes with the same probability of $\frac{1}{5}$. Each classifier makes a mistake with probability $p = 0.2$. (A 'mistake' here means that the 0's and the 1's in the column for the respective classifier are swapped.) The mistakes made by the classifiers are independent.

We simulated 10,000 objects and kept a count of the number of objects classified correctly. The simulation procedure for a single object was as follows.

(1) Pick a class label with probability $\frac{1}{5}$. Call it "the true label", and denote it by i, $i \in \{1, 2, 3, 4, 5\}$.
(2) Copy the code matrix in another matrix, C. For each classifier, decide with probability $p = 0.2$ whether it will make an error for this object. If yes, swap the 0's and the 1's in the corresponding column of C.
(3) If there were no misclassifications, the codeword for this object would be row i of the original code matrix. With the misclassifications made by the classifiers, the codeword now is the ith row of C, denoted C_i. We calculate the Hamming distances between C_i and each row of the original code matrix.
(4) The class label assigned by the ensemble is determined by the minimum of the five distances. In case of a tie, the assigned label is decided with equal probability between the tied labels. If the assigned label matches the true label, i, we increment the count for the correct classification.

[2] The Matlab line `C=[num2str(ones(2^(c-1)-1,1))'; dec2bin(0:2^(c-1)-2)']` produces a code matrix (of string type) for a given c. Since there are exactly $2^{(c-1)} - 1$ different assignments, the exhaustive code is obtained by enumerating the numbers from 0 to $2^{(c-1)} - 2$, converting them from decimal to binary and appending a string of 1's to be the first row of the code matrix.

Codematrix 1

	D_1	D_2	D_3	D_4	D_5
ω_1	1	0	0	0	0
ω_2	0	1	0	0	0
ω_3	0	0	1	0	0
ω_4	0	0	0	1	0
ω_5	0	0	0	0	1

Codematrix 2

	D_1	D_2	D_3	D_4	D_5
ω_1	1	0	0	0	0
ω_2	1	1	1	1	0
ω_3	1	0	1	1	1
ω_4	0	0	0	0	0
ω_5	0	1	0	1	1

H_c (min $H_c = 2$)

	ω_1	ω_2	ω_3	ω_4	ω_5
ω_1	0	2	2	2	2
ω_2	2	0	2	2	2
ω_3	2	2	0	2	2
ω_4	2	2	2	0	2
ω_5	2	2	2	$\boxed{2}$	0

mean $H_c = 2$

H_c (min $H_c = 1$)

	ω_1	ω_2	ω_3	ω_4	ω_5
ω_1	0	3	3	1	4
ω_2	3	0	2	4	3
ω_3	3	2	0	4	3
ω_4	$\boxed{1}$	4	4	0	3
ω_5	4	3	3	3	0

mean $H_c = 3$

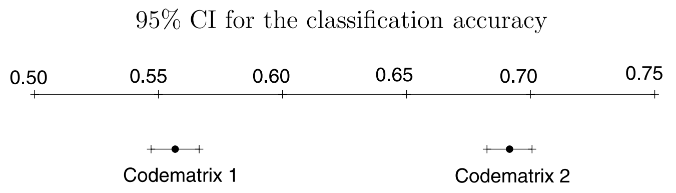

Fig. 1. An example of two ECOC ensembles. Maximizing the minimum Hamming distance will give preference to ensemble 1 which is less accurate *on average*.

The simulation was run separately for each of the two code matrices. The 95% confidence intervals (CI) for the accuracies of the two ensembles are shown in Fig. 1. Ensemble 2 outperforms ensemble 1 by a large margin, showing that the minimum Hamming distance may not be the best criterion.

4. Using diversity measures for ECOC

The idea of this study is to augment the set of measures of the quality of an ECOC for classifier ensembles with row- and column-separation measures based on classifier diversity. For this study we chose the "disagreement" measure of diversity (see Kuncheva and Whitaker, 2003). The "disagreement" measure between codewords C_i and C_j is equivalent to the relative Hamming distance

$$D_{i,j} = \frac{N^{01} + N^{10}}{N^{00} + N^{11} + N^{01} + N^{10}} = \frac{N^{01} + N^{10}}{L}, \qquad (2)$$

where N^{mn} is the number of bits for which C_i has value m and C_j has value n, where $m, n \in \{0, 1\}$, and L is the length of the codeword (total number

Table 3
H and D for ECOC generated by the one-per-class and all-pairs methods, and for the two code matrices from Fig. 1

	Row separation (codewords)	Column separation (dichotomies)
One-per-class (=Codematrix 1)	$H_c = 2$ $D_c = \frac{2}{c}(= 0.4)$	$H_L = 2$ $D_L = \frac{2}{c}(= 0.4)$
All-pairs	$H_c = 2(c-2)$ $D_c = \frac{4(c-2)}{c(c-1)}$	$H_L = \min\{2, c-4\}, c \geqslant 4$ $D_L = \frac{c^3-5c^2+22c-32-\lvert c-8\rvert(c^2-5c+6)}{2c(c^2-c-2)}$
Codematrix 2	$H_c = 1$ $D_c = 0.6$	$H_L = 1$ $D_L = 0.32$

of classifiers). D varies between 0 and 1. Larger values of D are desirable.

We should note that when we measure diversity between codewords (rows), we use directly (2). However, when column separation is measured (diversity between dichotomies), we must take into account that the inverse of a binary vector represents the same dichotomy. Therefore the diversity between classifiers D_i and D_j (columns of the code matrix) is

$$D_{i,j} = \min\left\{\frac{N^{01}+N^{10}}{c}, \frac{N^{00}+N^{11}}{c}\right\}. \qquad (3)$$

To measure the total diversity between the codewords, we take the average across all pairs

$$D_c = \frac{2}{c(c-1)} \sum_{i<j} D_{i,j}, \quad i, j = 1, \ldots, c. \qquad (4)$$

For the total diversity between the dichotomies (columns of the code matrix), we average the corresponding pairwise diversities (3) as

$$D_L = \frac{2}{L(L-1)} \sum_{i<j} M_{i,j}, \quad i, j = 1, \ldots, L. \qquad (5)$$

The new measures suggested here, D_c and D_L, account for the overall diversity in the ensemble. Table 3 shows the values of H and D for ECOC generated by the one-per-class and all-pairs methods as functions of the number of classes c. We also give the values of the measures for the two code matrices in the example in Fig. 1. Codematrix 1 produced values as given in the One-per-class section ($D_c = D_L = 0.4$). The calculations for the one-per-class method are straightforward. The all-pairs method needs some combinatorial and algebraic manipulations. [3]

To formulate one criterion function (maximum H or D), we have to combine the row- and column-separation measures. The simplest way to do so is to take the average for D, $D = \frac{1}{2}(D_c + D_L)$, so as to have a combined measure in the same range of values and the same interpretation. For H we use $H = H_c + H_L$.

According to D from Table 3, ensemble 2 should be preferred to ensemble 1 because the sum of the two disagreement values is larger. Conversely, H would choose ensemble 1. The results from the simulation experiment clearly favoured ensemble 2.

5. Generating ECOC by an evolutionary algorithm (EA)

Random search or guided variants of random search can be used to generate ECOC. We propose to use an evolutionary algorithm as opposed to pure random search. The reason is that the evolutionary procedure is effective yet simple and might lead to better ECOCs than pure random search.

The "chromosome" is the whole code matrix, with its rows concatenated into a vector of dimensionality $L \times c$ (L classifiers and c classes). The procedure starts with generating and evaluating a set of m chromosomes, called the population.

[3] The details of the derivations for Table 3 can be found at http://www.informatics.bangor.ac.uk/~kuncheva/data_public/ecoc_derivations.pdf.

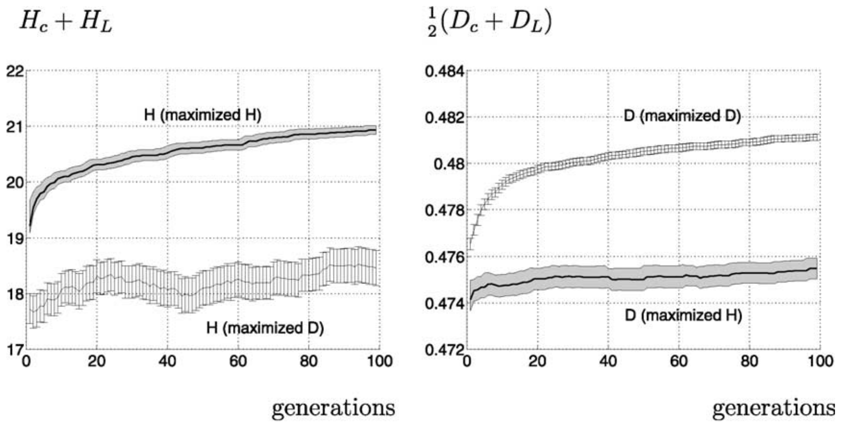

Fig. 2. *H* and *D* as functions of the number of generations in the EA (average from 100 runs; 95% confidence intervals displayed).

The population is duplicated into an offspring set. Each bit of each offspring chromosome is mutated with a specified probability P_{mut}. Each offspring chromosome (a code matrix) is evaluated. The population and the offspring sets are then pooled and the best *m* of the chromosomes survive to be the next population. The loop is run for a specified number of times, called "generations".

The evaluation of a chromosome is done by first breaking it, rearranging back the code matrix and then calculating the chosen measure *M* (*H* or *D*). We ran the EA for $c = 50$ classes and $L = 15$ classifiers. One hundred runs were carried out starting from different initialization. The parameters of the algorithms were as follows: $m = 10$, $P_{\text{mut}} = 0.15$, number of generations was 100. The averaged measures across the 100 runs of EA together with the 95% confidence intervals are shown in Fig. 2.

As expected, the measure being maximised shows higher mean and smaller variation about the mean. The 95% confidence intervals for the mean are calculated as $\mu \pm 1.96\frac{\sigma}{\sqrt{100}}$.

6. Comments and conclusions

In this paper we propose to use diversity measures for ECOC design in addition to the standard minimum Hamming distance measure *H*. We look into the disagreement measure *D*, which is the averaged Hamming distance. The motivation for the new measure came from the fact that maximizing the minimum *H* is not necessarily optimal with respect to the overall correctness of the ECOC. This was demonstrated by an example in Section 3. An evolutionary algorithm was implemented to design ECOCs using the measures as the fitness function.

It is true in general that more diverse classifiers make a better ensemble than less diverse classifiers but the relationship is not straightforward (Kuncheva and Whitaker, 2003). Besides, having diverse dichotomies does not automatically mean that the classifiers built to solve these dichotomies will be diverse. Thus the rationale for improving the diversity relies on a chain of intuitive *assumptions*.

There is *no direct experiment* by which we can compare different criteria. The EA will optimize whichever criterion we set as its fitness function. Therefore it is not surprising that the best values of measure *M* were found by running EA with *M* as the criterion function (*M* standing for *D* or *H*). The ultimate test for the proposed criteria is the performance of ensembles built using the corresponding ECOCs. A set of experiments can be carried out using real multiclass data sets. The errors can be compared across the two design criteria for ECOC (*H* and *D*).

There is no issue of training and generalization in this study. This is because the goal is to devise a concrete structure (ECOC) which can then be used in training and testing classifier ensembles.

It is worth investigating the *relative importance of the two components* of the total diversity (M_c and M_L). Unequal weights can be assigned to these. It is possible that different measures are suitable for the row separation and column separation. The same search algorithm can be applied with a correspondingly modified fitness function.

References

Aha, D.W., Blankert, R.L., 1997. Cloud classification using error-correcting output codes. Artificial Intell. Appl.: Natural Resour., Agric. Environ. Sci. 11 (1), 13–28.

Dietterich, T.G., Bakiri, G., 1991. Error-correcting output codes: A general method for improving multiclass inductive learning programs. In: Proceedings of the 9th National Conference on Artificial Intelligence, AAAI-91. AAAI Press, pp. 572–577.

Dietterich, T.G., Bakiri, G., 1995. Solving multiclass learning problems via error-correcting output codes. J. Artificial Intell. Res. 2, 263–286.

Kuncheva, L.I., Whitaker, C.J., 2003. Measures of diversity in classifier ensembles. Mach. Learn. 51, 181–207.

Masulli, F., Valentini, G., 2000a. Comparing decomposition methods for classification. In: Proc. International Conference on Knowledge-Based Intelligent Engineering Systems and Applied Technologies (Kes 2000), Brighton, UK, pp. 788–792.

Masulli, F., Valentini, G., 2000b. Effectiveness of error-correcting output codes in multiclass learning problems. In: Kittler, J., Roli, F. (Eds.), Multiple Classifier Systems, Lecture Notes in Computer Science, vol. 1857. Springer, Cagliari, Italy, pp. 107–116.

Shapire, R.E., 1997. Using output codes to boost multiclass learning problems. In: Proceedings of the 14th International Conference on Machine Learning.

Windeatt, T., Ghaderi, R., 2001. Binary labelling and decision level fusion. Inform. Fusion 2, 103–112.

Windeatt, T., Ghaderi, R., 2003. Coding and decoding strategies for multi-class learning problems. Inform. Fusion 4, 11–21.