Pattern Recognition Letters 30 (2009) 285–297

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Separability of ternary codes for sparse designs of error-correcting output codes

Sergio Escalera *, Oriol Pujol, Petia Radeva

Centre de Visió per Computador, Campus UAB, Edifici O, 08193 Bellaterra, Barcelona, Spain
Dept. Matemàtica Aplicada i Anàlisi, Universitat de Barcelona, Gran Via 585, 08007 Barcelona, Spain

ARTICLE INFO

Article history:
Received 8 January 2008
Received in revised form 1 October 2008
Available online 15 October 2008

Communicated by W. Pedrycz

Keywords:
Error-correcting output codes
Embedding of dichotomizers
Sparse random designs

ABSTRACT

Error-correcting output codes (ECOC) represent a successful framework to deal with multi-class categorization problems based on combining binary classifiers. With the extension of the binary ECOC to the ternary ECOC framework, ECOC designs have been proposed in order to better adapt to distributions of the data. In order to decode ternary matrices, recent works redefined many decoding strategies that were formulated to deal with just two symbols. However, the coding step also is affected, and therefore, it requires to be reconsidered. In this paper, we present a new formulation of the ternary ECOC distance and the error-correcting capabilities in the ternary ECOC framework. Based on the new measure, we stress on how to design coding matrices preventing codification ambiguity and propose a new sparse random coding matrix with ternary distance maximization. The results on a wide set of UCI Machine Learning Repository data sets and in a real speed traffic sign categorization problem show that when the coding design satisfies the new ternary measures, significant performance improvement is obtained independently of the decoding strategy applied.

1. Introduction

In the literature, one can find several powerful types of binary classifiers. However, when one needs to deal with multi-class classification problems, many learning techniques fail to manage this information. Instead, it is common to construct the classifiers to distinguish between just two classes, and to combine them. In this sense, error-correcting output codes were born as a general framework to combine binary problems to address the multi-class problem. The strategy was introduced by Dietterich and Bakiri (1995). Based on the error-correcting principles (Dietterich and Bakiri, 1995) and because of its ability to correct the bias and variance errors of the base classifiers (Kong and Dietterich, 1995), ECOC has been successfully applied to a wide range of Computer Vision applications, such as face recognition (Windeatt and Ardeshir, 2003), face verification (Kittler et al., 2001), text recognition (Ghani, 2001) or manuscript digit classification (Zhou and Suen, 2005).

The ECOC technique can be broken down into two general stages: encoding and decoding. Given a set of classes, the coding stage designs a codeword[1] for each class based on different binary problems. The decoding stage makes a classification decision for a given test sample based on the value of the output code.

At the coding step, given a set of N classes to be learnt, n different bi-partitions (groups of classes) are formed, and n binary problems (dichotomizers) are trained. As a result, a codeword of length n is obtained for each class, where each bit of the code corresponds to the response of a given dichotomizer (coded by +1, -1, according to its class set membership). Arranging the codewords as rows of a matrix, we define a *coding matrix* M, where $M \in \{-1, 1\}^{N \times n}$ in the binary case. The most well-known binary coding strategies are the one-versus-all strategy (Nilsson, 1965), where each class is discriminated against the rest of classes, and the dense random strategy (Allwein et al., 2002), where a random matrix M is generated maximizing the rows and columns separability in terms of the Hamming distance (Dietterich and Bakiri, 1995). In Fig. 1a, the one-versus-all ECOC design for a 4-class problem is shown. The white regions of the coding matrix M correspond to the positions coded by 1, and the black regions to -1. Thus, the codeword for class c_1 is $\{1, -1, -1, -1\}$. Each column j of the coding matrix codifies a binary problem learnt by its corresponding dichotomizer h_i. For instance, dichotomizer h_1 learns c_1 against classes c_2, c_3 and c_4, dichotomizer h_2 learns c_2 against classes c_1, c_3 and c_4, etc. An example of a dense random matrix for a 4-class problem is shown in Fig. 1c.

It was when Allwein et al. (2002) introduced a third symbol (the zero symbol) in the coding process that the coding step received special attention. This symbol increases the number of partitions of classes to be considered in a ternary ECOC framework by allowing some classes to be ignored. Then, the ternary coding

* Corresponding author. Address: Centre de Visió per Computador, Campus UAB, Edifici O, 08193 Bellaterra, Barcelona, Spain. Tel.: +34 687291957; fax: +34 93 581 16 70.
E-mail addresses: sergio@maia.ub.edu (S. Escalera), oriol@maia.ub.edu (O. Pujol), petia@maia.ub.edu (P. Radeva).

[1] The codeword is a sequence of bits of a code representing each class, where each bit identifies the membership of the class for a given binary classifier.

0167-8655/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2008.10.002

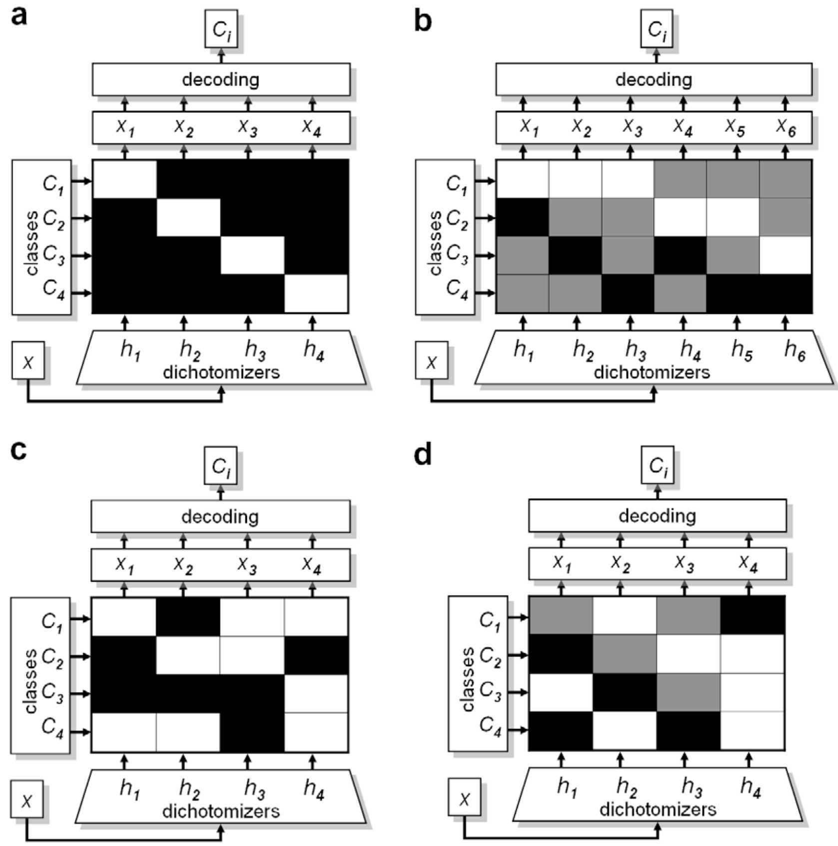

Fig. 1. One-versus-all (a), one-versus-one (b), dense random (c), and (d) sparse random ECOC designs.

matrix becomes $M \in \{-1, 0, 1\}^{N\times n}$. In this case, the symbol zero means that a particular class is not considered by a certain binary classifier. Thanks to this, strategies such as one-versus-one (Hastie and Tibshirani, 1998) and sparse random coding (Allwein et al., 2002) have been formulated in the ECOC framework. Fig. 1b shows the one-versus-one ECOC configuration for a 4-class problem. In this case, the grey positions correspond to the zero symbol. A possible sparse random matrix for a 4-class problem is shown in Fig. 1d. Recently, new improvements in the ternary ECOC coding demonstrate the suitability of the ECOC methodology to deal with multi-class classification problems (Pujol et al., 2006; Escalera et al., 2007). These recent designs use the knowledge of the problem-domain to learn relevant binary problems from ternary codes. The basic idea of these methods is to use the training data to guide the training process, and thus, to construct the coding matrix M focusing on the binary problems that better fits the decision boundaries of a given data set.

The decoding step was originally based on error-correcting principles under the assumption that the learning task can be modeled as a communication problem, in which class information is transmitted over a channel (Dietterich and Bakiri, 1995). During the decoding process, applying the n binary classifiers, a code is obtained for each data point in the test set. This code is compared to the base codewords of each class defined in the matrix M, and the data point is assigned to the class with the *closest* codeword. The most frequently applied decoding strategies are the Hamming (*HD*) (Nilsson, 1965) and the Euclidean (*ED*) decoding distances (Hastie and Tibshirani, 1998). With the introduction of the zero symbol, Allwein et al. (2002) showed the advantage of using a Loss-based function of the output margin of the base classifier. Recently, Escalera et al. (2008) proposed a loss-weighted strategy to decode, where a set of probabilities based on the performances of the base classifiers is used to weight the final classification decision. In Fig. 1, each ECOC codification is used to classify an input object X. The input X is tested with each dichotomizer h_i, obtaining an output $X_i, i \in \{1, .., n\}$. The final code $\{X_1, \ldots, X_n\}$ of the test input X is used by a given decoding strategy to obtain the final classification decision. Note that in both, the binary and the ternary ECOC framework, the value of each position X_j of the test codeword can not take the value zero since the output of each dichotomizer is $h_j \in \{-1, +1\}$, meaning the automatical increasing of distance/error.

To deal with multi-class categorization problems in the ternary ECOC framework, recent works redefined decoding strategies that were formulated to deal with just two symbols (Escalera et al., 2008; Allwein et al., 2002). However, the influence of the zero symbol to the error-correction capabilities and the design of the coding strategies have not been taken into account. In this paper, we formulate the ternary distance and the ternary error-correcting capabilities in the ternary ECOC framework. We propose a new sparse coding design based on maximizing the new ternary distance. We evaluate the methodology on a wide set of UCI Machine Learning Repository data sets and in a real Computer Vision problem: speed traffic sign categorization. The results show that when the new ternary distance is considered on sparse designs, significant performance improvement is obtained.

The paper is organized as follows: Section 2 overviews the ECOC random designs and presents a new sparse coding design based on ternary distance maximization. Section 3 presents the experimental results. Finally, Section 4 concludes the paper.

2. Random ECOC designs

In this section, we overview both dense and sparse random ECOC designs (Allwein et al., 2002). We show the inconsistency of the classical sparse random design and introduce a new measure for sparse coding designs.

2.1. Dense random design

Let us consider a binary ECOC matrix $M \in \{-1,1\}^{N\times n}$, where N is the number of classes and n the codeword length. Based on Eq. (6) in (Allwein et al., 2002), the minimum Hamming distance d_r among all pairs of rows can be defined as follows (Allwein et al., 2002):

$$d_r = \min_{i_1,i_2}\left\{\sum_{j=1}^{n}\left(1 - sign\left(y^j_{i_1}\cdot y^j_{i_2}\right)\right)/2\right\}$$

for $i_1,i_2 \in \{1,\ldots,N\}, i_1 \neq i_2$, being $y^j_{i_1}$ the jth position of the codeword for class c_{i_1}. Suppose that two codewords coded using $\{-1,+1\}$ values have a Hamming distance of three. Then, it means that even if we fail in a bit, we still are able to obtain the correct classification. It suggests that a distance d_r in a binary ECOC matrix M can correct $[d_r - 1]/2$ codeword errors at the decoding step (Dietterich and Bakiri, 1995). Because of these binary error-correction capabilities, many ECOC designs, such as random ECOC strategies, base the design of the ECOC coding matrix on maximizing the value d_r (Allwein et al., 2002).

Let us consider the distance d_c between all pairs of columns and their opposites:

$$d_c = \min_{j_1,j_2}\{min(A(j_1,j_2), B(j_1,j_2))\}$$

being

$$A(j_1,j_2) = \sum_{i=1}^{N}\left(1 - sign\left(y^{j_1}_i\cdot y^{j_2}_i\right)\right)\Big/2$$
$$B(j_1,j_2) = \sum_{i=1}^{N}\left(1 - sign\left(-1\cdot(y^{j_1}_i)\cdot y^{j_2}_i\right)\right)\Big/2 \quad (1)$$

where $j_1,j_2 \in \{1,\ldots,n\}, j_1 \neq j_2$. High value of d_c contributes to consider different sub-partitions of classes and to increase the variability of the knowledge of the classifiers. Note that in Eq. (1) the factor (-1) is used to take into account the independence of the class ordering, i.e. the base classifier learns the same problem from the partition C_1 versus C_2 and from C_2 versus C_1.

The dense random ECOC strategy (Allwein et al., 2002) tries to maximize simultaneously both previous d_r and d_c distances to design matrices where the decoding strategies are able to obtain a correct classification still when there exist failures in some bits of the tested codewords. The dense random strategy generates a high number of random coding matrices M of length n, where the values $\{+1,-1\}$ have a certain probability to appear (usually $P(1) = P(-1) = 0.5$). Studies on the performance of the dense random strategy suggest a length of $n = 10\log N$ (Allwein et al., 2002). In order to assure optimal performance of ECOC classification, for the set of generated dense random matrices, the optimal one should maximize the Hamming decoding measure between rows d_r and columns d_c (also considering the opposites), taking into account that each column of the matrix M must contain both different symbols $\{-1,+1\}$.

In Fig. 2 some coding errors are shown. Fig. 2a has a dichotomizer (h_3) with all the elements coded by -1. In this case, we do not have two groups of classes to split. Fig. 2b has the hypotheses h_1 and h_4 splitting the same sub-groups of classes in opposite order, which exactly learns the same problem. The coding matrix M of Fig. 2c is not able to distinguish between classes c_1 and c_3 since their respective codewords y_1 and y_3 are the same. The three previous problems in the ECOC designs do not occur when we use standard coding strategies such as one-versus-one or one-versus-all. When we use the dense random strategy defined in (Allwein et al., 2002), one needs to consider each dichotomizer to have positions coded by +1 and -1 in order to maximize the Hamming decoding measure among the columns and their opposites; and to have a high Hamming decoding value between rows, which prevents the errors produced in Fig. 2a–c, respectively. As commented in the Allwein's paper (Allwein et al., 2002): "*For each problem, we picked a code with high value of ρ and did not have any identical columns.*"

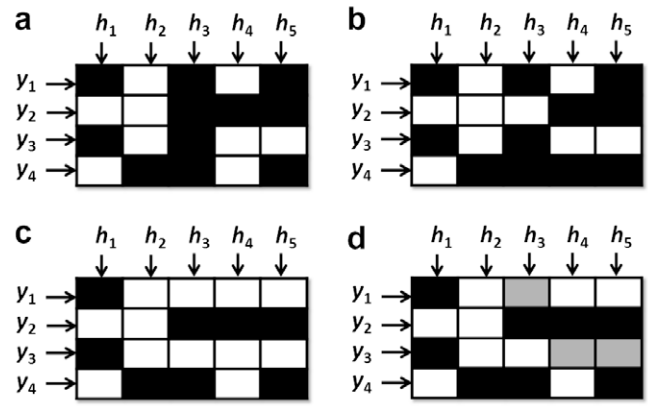

Fig. 2. Wrong binary and ternary ECOC designs. (a) Wrong hypothesis h_3. (b) Redundant hypotheses h_1 and h_4. (c) Repeated codewords y_1 and y_3 for classes c_1 and c_3. (d) Codification error between classes c_1 and c_3.

2.2. Classical sparse random design

One of the main limitations of the binary ECOC framework is the need of considering all classes for each binary classifier. Although a high distance d_r and d_c can be computed, the selection of the most relevant sub-partition of classes for different multi-classification problems is not assured in the coding design. This fact implies the need of designing large codes to increase the discriminating ability of the combined set of binary problems. Moreover, taking into account the whole set of classes for each classifier significantly reduces the number of possible sub-partitions of classes to consider.

To take into account a higher number of possible classifiers, a third symbol was introduced in the ECOC framework (Allwein et al., 2002). In this sense, the sparse random strategy is designed in the same way than the Dense design, but it includes the third symbol zero with another probability to appear, given by $P(0) = 1 - P(-1) - P(1)$. Studies suggest a sparse code length of $15\log N$ (Allwein et al., 2002).

We consider that to increase the class separability in the ternary ECOC framework, the distance d_c of the binary case can be maintained since all three symbols $\{-1,0,+1\}$ have influence on the information learnt by each dichotomizer. It means that the distance between columns produced by the positions coded by zero increases the variability of the classifiers. However, we argue that the use of the codewords separability maximizing the measure d_r to design a sparse random matrix may contain inconsistency.

2.2.1. Sparse random design with ternary separability

Let us show an example to analyze sparse designs. A zero symbol in a class code introduces *one degree of freedom*, that means that both +1 and -1 are possible values during the test classification since the class has not been taken into account to train the corresponding dichotomizer. Any codeword y_i containing the zero symbol defines an extended set of possible codewords that could be obtained by examples of the class c_i. In this sense, a possible codeword $y_1 = \{1, 0, 0\}$ can be disambiguated into its extended set of codewords $Y_1^e = \{\{1, 1, 1\}, \{1, 1, -1\}, \{1, -1, 1\}, \{1, -1, -1\}\}$, where each of the four codewords of y_1 is a possible representation of the same codeword y_1, and possible representation means that any test example of class c_1 would give a codeword from Y_1^e. Now, a possible codeword for a second class $y_2 = \{1, 1, 1\}$ corresponds to one of the four possible representations of $y_1 (y_2 \in Y_1^e)$.

Let us consider another example of codewords of length six. Suppose that we randomly define two codewords $y_1 = \{1, 1, 1, 0, 0, 0\}$ and $y_2 = \{0, 0, 0, 1, 1, 1\}$ in a sparse random design. If we use the classical distance d_r between y_1 and y_2, we obtain a class separability of three. However, based on the previous example, if we disambiguate y_1 and y_2, we obtain that $Y_1^e \cap Y_2^e = \{1, 1, 1, 1, 1, 1\}$. Thus, an input test codeword $X = \{1, 1, 1, 1, 1, 1\}$ belongs to both previous codewords, which implies a wrong sparse design.

Finally, observe the ternary coding matrix M of Fig. 2d. Suppose that the matrix M of the figure receives an input test data sample which codeword corresponds to $X = \{-1, 1, 1, 1, 1\}$. This codeword matches with the four positions different of zero from class c_1 and the three from class c_3. In this case, $X \in Y_1^e$ and $X \in Y_3^e$. Thus, both classes can be a possible solution. However, the *HD* between codewords y_1 and y_3 produces a value of 1.5. Note that in the literature (Allwein et al., 2002), a sparse random matrix is generated by selecting the matrix from a previous set of matrices that maximizes the distances d_r and d_c. As commented, the *HD* between columns containing the third symbol is still useful since the zero positions help to create a rich set of partitions to be learnt. However, the measure d_r for the row separability in terms of the *HD*, as claimed, is inconsistent. Instead, to assure that the coding matrix M splits all pairs of classes, each pair of codewords of M should be split by at least one hypothesis.

Definition 1. The *ternary separability* condition of a matrix M is fulfilled if for any two codewords there exists a dichotomizer that discriminate them, that is

$$\forall (y_{i_1}, y_{i_2}) | i_1, i_2 \in \{1, \ldots, N\}, \quad i_1 \neq i_2, \qquad \exists h_j | (c_{i_1} \in C_1^j, c_{i_2} \in C_2^j) \vee (c_{i_2} \in C_1^j, c_{i_1} \in C_2^j)$$

where C_1^j and C_2^j are the two subsets of classes for hypothesis h_j, respectively. Then, we can define the distance between two codewords in a ternary symbol-based ECOC:

Definition 2. The *ternary distance* between two codewords (y_1, y_2) is defined as

$$d(y_1, y_2) = \sum_{j=1}^{n} \frac{1}{2} \left| y_1^j \right| \left| y_2^j \right| \left(1 - y_1^j y_2^j \right)$$

It defines the number of different bits between two codewords without taking into account the positions coded by zero. Note that the term $\frac{1}{2}(1 - y_1 y_2)$ is equivalent to the standard Hamming distance estimated in the binary case expressed in a more compact way. Thus, the weighting term $|y_1^j||y_2^j|$ makes the distance to ignore the zero positions which do not give information about the classes separability. Then, the pair of codewords (y_{i_1}, y_{i_2}) that are split by the minimum number of hypothesis in a ternary ECOC matrix M defines the new distance d_t:

Definition 3. The *distance* d_t of a coding matrix M is defined as follows:

$$d_t = \underset{i_1, i_2}{argmin} \sum_{j=1}^{n} \frac{1}{2} \left| y_{i_1}^j \right| \left| y_{i_2}^j \right| \left(1 - y_{i_1}^j y_{i_2}^j \right)$$

where the term d_t defines the distance between the pair of codewords that are split by the minimum number of binary problems in a ternary symbol-based ECOC matrix.

Based on the new ternary distance, we can define the error-correcting capabilities in the ternary ECOC framework. As the distance in the ternary case has been reformulated, the new measure of error-correction also changes. Having a N-multi-class classification problem in the binary ECOC framework, a distance d_r between rows of M can correct $[d_r - 1]/2$ bits errors. In the ternary case, the maximum class separability is defined by the measure d_t. Thus, on a sparse ECOC matrix, $[d_t - 1]/2$ bits errors can be corrected.[2]

As the use of the distance d_r applied to the classical design of the sparse random matrix M produces inconsistencies, we suggest to redefine the coding stage of the sparse random designs. A good codification of a ternary matrix should assure the highest number of codeword bits splitting each pair of rows; that is to maximize the value d_t. Therefore, we propose to use the new measure of ternary separability for the sparse random design. In this case, the selected random matrix should be that one which maximizes simultaneously d_c and d_t.

3. Results

We discuss the data, comparatives, and measurements of the experiments before the results are presented.

- *Data*: The data used for the experiments consists of 16 multi-class data sets from the UCI Machine Learning Repository database (Asuncion and Newman, 2007). The details of the data sets are shown in Table 1.We also use the video sequences obtained from a Mobile Mapping System (Casacuberta et al., 2004) to test the methods in a real traffic sign categorization problem.
- *Comparative*: For the comparative, we use the classical sparse random (Allwein et al., 2002), dense random, one-versus-all, and the new sparse random design with ternary distance maximization. To decode, we use 13 state-of-the-art decoding strategies: Hamming decoding (*HD*) (Dietterich and Bakiri, 1995), Euclidean decoding (*ED*) (Hastie and Tibshirani, 1998), inverse Hamming decoding (*IHD*) (Windeatt and Ghaderi, 2003), attenuated Euclidean decoding (*AED*) (Escalera et al., 2007), loss-based decoding with linear (*LLB*) and exponential (*ELB*) loss-functions (Allwein et al., 2002), probabilistic decoding (*PD*) (Passerini et al., 2004), Laplacian decoding (*LAP*) (Escalera et al., 2006), pessimistic β-density distribution decoding (β-*DEN*) (Escalera et al., 2006), linear loss-weighted (*LLW*) with discrete and continuous outputs of the classifier (Escalera et al., 2008), and the exponential loss-weighted (*ELW*) with discrete and continuous outputs of the classifier (Escalera et al., 2008). The base classifiers used for the experiments are Gentle Adaboost with 50 runs of decision stumps (Friedman et al., 1998), the linear support vector machines (*SVM*),[3] and a tuned Support Vector Machines with Radial Basis Function kernel (Vapnik, 1995).[4]
- *Measurements*: The data used in the experiments is normalized to an hypercube with a side length of one. To measure the performance of the different strategies we apply stratified tenfold cross-validation and test for confidence interval at 95% with a two-tailed t-test.

[2] We realize that the error-correcting capability also depends on the way that the decoding strategies are applied.

[3] The regularization parameter C is set to 1 for all the experiments. We selected this parameter after a preliminary set of experiments. We decided to keep the parameter fixed for the sake of simplicity and easiness of replication of theexperiments, though we are aware that this parameter might not be optimal for all data sets. Nevertheless, since the parameters arethe same for all the compared methods any weakness in the results will also be shared.

[4] Osu-svm-toolbox. URL http://svm.sourceforge.net.

Table 1
UCI repository data sets characteristics.

Problem	Train	Features	Classes	Problem	Train	Features	Classes
Dermathology	366	34	6	OptDigits	5620	64	10
Iris	150	4	3	Shuttle	14,500	9	7
Ecoli	336	8	8	Vehicle	846	18	4
Wine	178	13	3	Segmentation	2310	19	7
Glass	214	9	7	Pendigits	10,992	16	10
Thyroid	215	5	3	Letter	20,000	16	26
Vowel	990	10	11	Satimage	6435	36	7
Balance	625	4	3	Yeast	1484	8	10

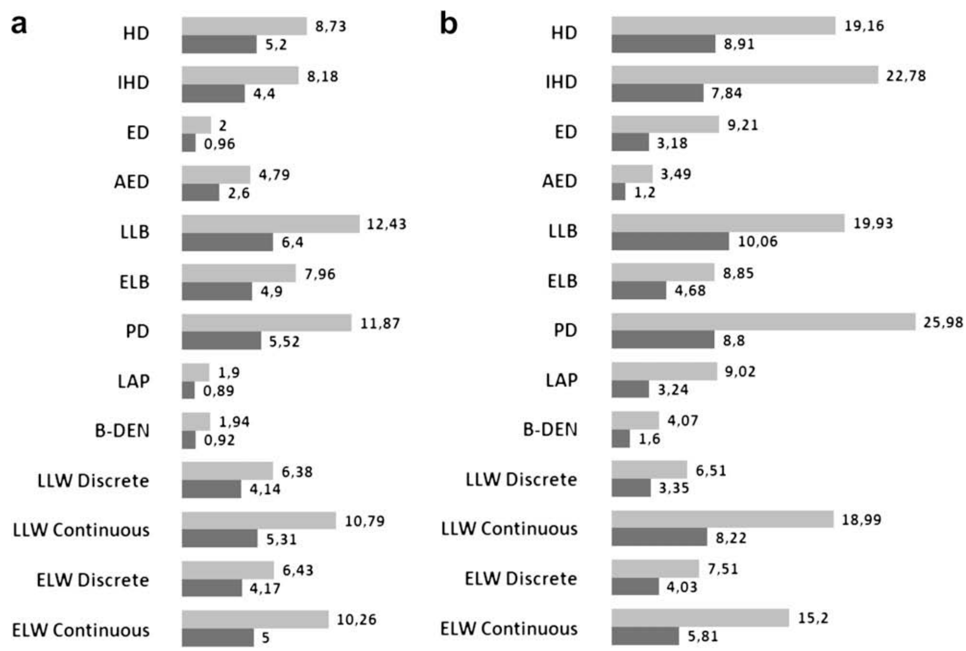

Fig. 3. Absolute (light lines) and relative (dark lines) improvements for the sparse random designs compared with classical sparse random using ternary distance maximization for Gentle Adaboost (left) and Linear *SVM* (right) on the UCI experiments, respectively.

3.1. UCI classification

In this experiment, we classify the 16 multi-class UCI Machine Learning Repository data sets of Table 1. To test the sparse random strategies, we generated a set of 20000 arbitrary random matrices with a length of the codewords of N, where the probabilities of appearance of each symbol are $P(0) = P(1) = P(-1) = 1/3$. From exactly the same set of generated matrices, we selected the classical sparse random matrix by the one which maximizes d_r and d_c, and the new sparse random matrix by selecting the one which maximizes d_t and d_c. To decode, the commented 13 decoding strategies are applied over the sparse random designs for Gentle Adaboost and Linear *SVM* as the base classifiers.

Tables 2 and 3 of Appendix A show the performance results and confidence intervals applying stratified tenfold cross-validation for Gentle Adaboost and Linear *SVM*, respectively. To show the performance improvements by selecting the new sparse random matrix, the absolute and relative improvements are shown in Fig. 3. The relative improvement is computed as the division between the performance of the new sparse design and the classical one, and the absolute improvement corresponds to the direct difference of performances. The light bars correspond to the absolute improvement, and the dark lines to the relative one. Note that simply changing the decision on the selection of the sparse matrix from the same set of generated random matrices, the performance significantly increases independently of the decoding strategy applied. It is produced since the maximization of d_t assures us to select the matrix with the higher number of bits splitting codewords (and thus, classes).

The same experiment is also computed for the dense random design. In this case, the probabilities of appearance of each symbol are $P(1) = P(-1) = 1/2$. Tables 4 and 5 of Appendix B show the performance results and confidence intervals applying stratified tenfold cross-validation for Gentle Adaboost and Linear *SVM*, respectively. The absolute and relative improvements are shown in Fig. 4. In this case, though the absolute and relative improvements have less impact compared to the previous experiment, one can observe that our approach performs better for most of the decoding strategies.

3.2. Real multi-class traffic sign categorization

For this experiment, we use the video sequences obtained from a Mobile Mapping System (Casacuberta et al., 2004) to test the methods in a real traffic sign Computer Vision problem. In this system, the position and orientation of the different traffic signs are

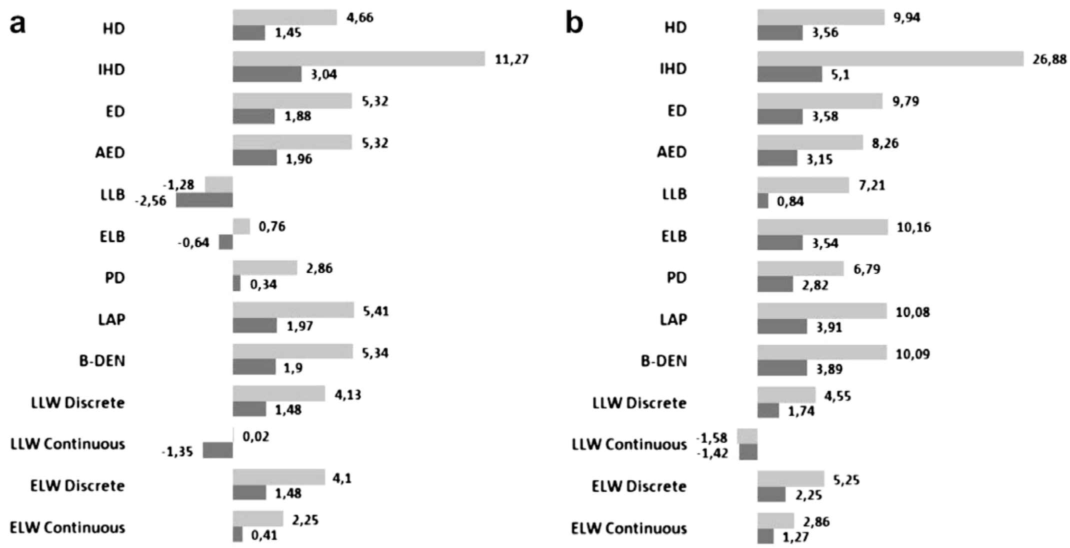

Fig. 4. Absolute (light lines) and relative (dark lines) improvements for the sparse random designs compared with classical dense random using ternary distance maximization for Gentle Adaboost (left) and Linear *SVM* (right) on the UCI experiments, respectively.

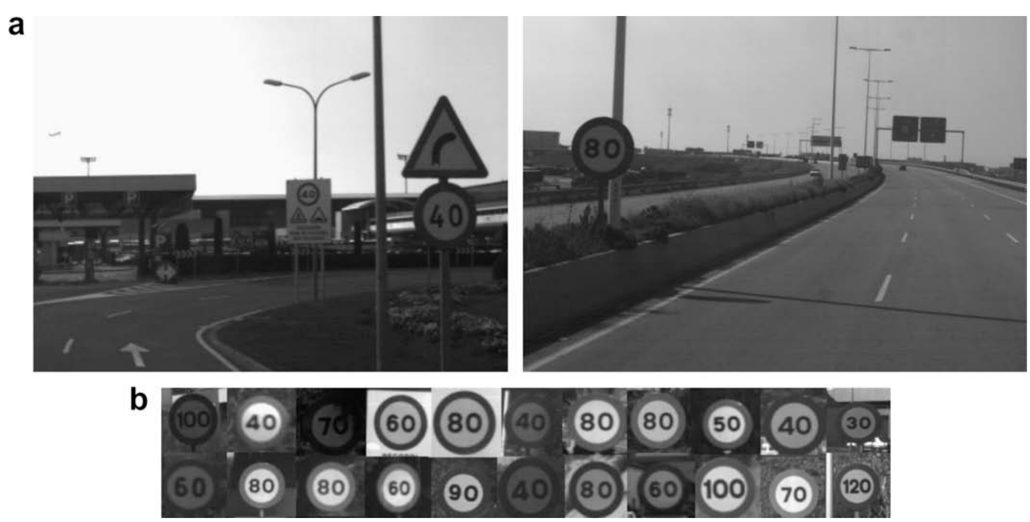

Fig. 5. (a) Samples from the road video sequences. (b) Speed data set samples.

measured with video cameras fixed on a moving vehicle. The system has a stereo-pair of calibrated cameras, which are synchronized with a GPS/INS system. The result of the acquisition step is a set of stereo-pairs of images with their position and orientation information. We choose the speed data set since the low resolution of the images, the non-controlled conditions, and the high similarity among classes make the categorization a difficult task. Fig. 5 shows examples of video sequences and samples of the speed database used in the experiments. The database contains a total of 2500 samples divided in nine classes. Each sample is composed by 1200 pixel-based features after smoothing the image and applying histogram equalization. For this experiment, we applied the same random criteria than at the previous experiment, with a length of codewords of nine bits (equal to the number of classes).

Table 6 of Appendix C shows the performance results and confidence intervals applying stratified tenfold cross-validation. To show the performance improvements by selecting the new sparse random matrix, the absolute and relative improvements are shown in Fig. 6 for Gentle Adaboost and Linear *SVM*, respectively. The light bars correspond to the absolute improvement, and the dark lines to the relative one. In this experiment, one can see that the ternary sparse maximization criterion also obtains performance improvements for all decoding strategies.

3.3. UCI classification using RBF SVM

In the previous experiments, the parameters for the Linear *SVM* classifier were fixed by default to compare the performance of the different coding and decoding strategies at the same conditions. However, complex classifiers and optimizations can improve the results of the strategies. In particular, the authors of Rifkin and Klautau (2004) show that the simple one-versus-all scheme is as accurate as any other schemes when complex base classifiers are applied. In this sense, we include a brief experiment considering a *SVM* with Radial Basis Function kernel optimized via cross-validation applied over the new sparse random design and the one-

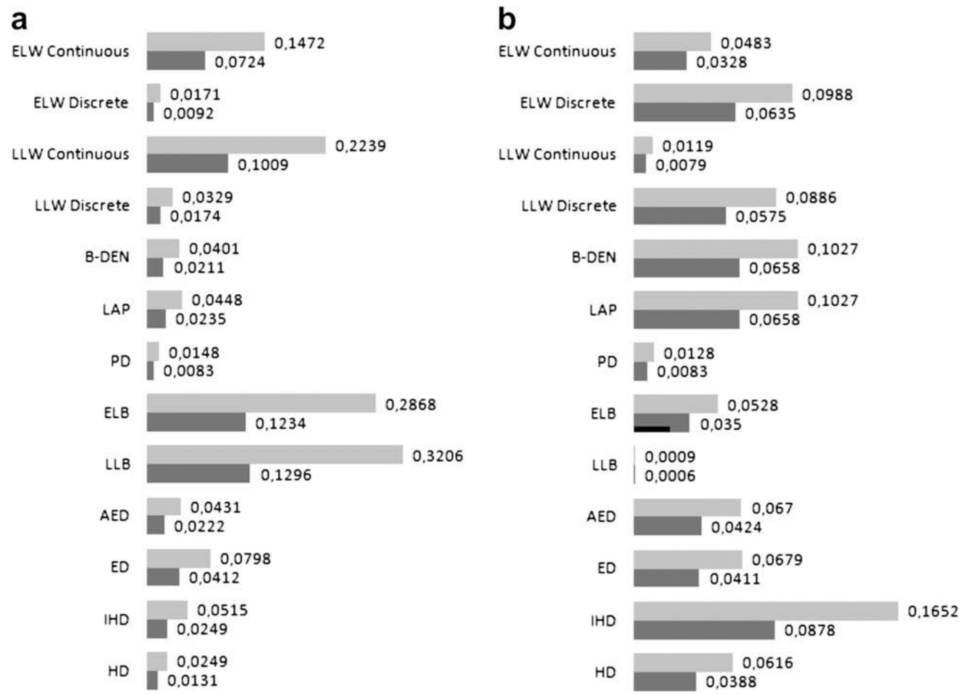

Fig. 6. Absolute (light lines) and relative (dark lines) improvement for the sparse random designs using ternary distance maximization for Gentle Adaboost (left) and Linear *SVM* (right) on the Traffic sign categorization experiment, respectively.

versus-all strategies using the set of decoding strategies and UCI data sets to look for the behavior of the new sparse design when a more complex base classifier is applied.

For this experiment, the sigma and regularization parameters were tested from 0.1 increasing per 0.05 up to one and from one increasing per five up to 150, respectively. The design of the sparse random matrix is done at same condition than at the previous experiments, and considering a length of the codeword of 2N, being N the number of classes. The UCI data sets used correspond to the eight data sets described in the first column of Table 1: Dermathology, Iris, Ecoli, Wine, Glass, Thyroid, Vowel, and Balance. The performances obtained in this experiment are numerically shown in Table 7 of Appendix D. To show the performance improvements by selecting the new sparse random matrix, the absolute and relative improvements are shown in Fig. 7 for *RBF SVM*. The light bars correspond to the absolute improvement, and the dark lines to the relative one. In Table 7 one can see that the performances obtained using *RBF SVM* are superior to the ones obtained at the previous experiments for Gentle Adaboost and Linear *SVM* as the base classifiers. Fig. 7 shows that the absolute and relative improvements in this case are less significant in this experiment, but still in most cases we outperform the results obtained by the one-versus-all strategy using *RBF SVM*. In the cases where we obtain inferior results, these differences are not significant.

As a conclusion of the experiments, we can state that the distance d_t based on maximizing the ternary separability allows high splitting of the classes codewords. In the previous experiments significant performance improvements are obtained, independently of the decoding strategy applied, when the sparse matrix is selected by maximizing the d_t criterion. Note that the classical sparse matrix is selected from the same set of matrices as the new sparse matrix, but it obtains very inferior results. This suggests that for designs that consider the new measures, class separability is increased. Thus, the decoding strategies are able to distinguish among different codewords with higher confidence. Moreover, the ternary distance can be applied to problem-dependent ECOC

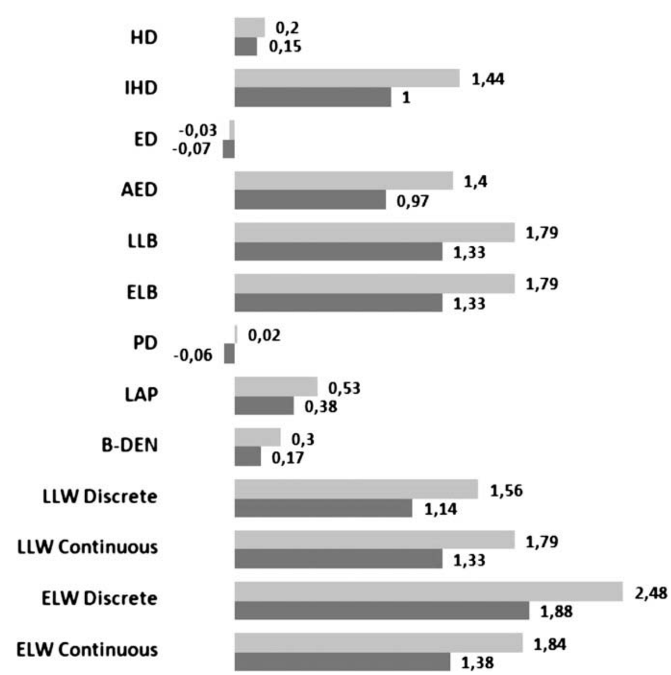

Fig. 7. Absolute (light lines) and relative (dark lines) improvements for the new sparse random design compared to the one-versus-all strategy using *RBF SVM* on the UCI data sets.

schemes, assuring the consistence of the designs. At the same time, the new measures can also help the decoding strategies to evaluate those positions of codewords that directly affect class separability.

Note that this measure corresponds to the attenuated Euclidean decoding (*AED*), as described in (Escalera et al., 2007), which is included in the experimental evaluation of the paper. This method and the rest of strategies designed to deal with a ternary decoding obtain robust results. However, the best results are obtained by Laplacian, β-density, and loss-weighted decoding variants, which, as can be easily shown, subsume the *AED* approach but also include

Table 2
Sparse random results using Gentle Adaboost on the UCI data sets.

	HD	*IHD*	*ED*	*AED*	*LLB*	*ELB*	*PD*	*LAP*	β-*DEN*	*LLW* disc.	*LLW* cont.	*ELW* disc.	*ELW* cont.
Derma	0.588	0.634	0.636	0.647	0.549	0.587	0.444	0.636	0.636	0.650	0.452	0.650	0.436
	0.027	0.012	0.011	0.009	0.017	0.021	0.035	0.011	0.011	0.008	0.051	0.008	0.042
	0.926	0.923	0.926	0.923	0.896	0.926	0.945	0.926	0.926	**0.929**	0.920	**0.929**	0.940
	0.017	0.018	0.017	0.015	0.024	0.021	0.013	0.017	0.017	**0.017**	0.015	**0.017**	0.015
Iris	0.933	0.933	0.933	0.933	0.953	0.953	0.953	0.933	0.933	0.933	0.953	0.933	0.953
	0.019	0.019	0.019	0.019	0.014	0.014	0.014	0.019	0.019	0.019	0.014	0.019	0.014
	0.926	0.926	0.926	0.926	**0.960**	**0.960**	**0.960**	0.926	0.926	0.933	**0.960**	0.933	**0.960**
	0.020	0.020	0.020	0.020	**0.014**	**0.014**	**0.014**	0.020	0.020	0.019	**0.014**	0.019	**0.014**
Ecoli	0.373	0.379	0.367	0.533	0.284	0.302	0.493	0.370	0.357	0.539	0.443	**0.551**	0.477
	0.017	0.016	0.021	0.014	0.019	0.020	0.017	0.018	0.021	0.015	0.033	**0.011**	0.029
	0.373	0.379	0.367	0.533	0.284	0.302	0.493	0.370	0.357	0.539	0.443	**0.551**	0.477
	0.017	0.016	0.021	0.014	0.019	0.020	0.017	0.018	0.021	0.015	0.033	**0.011**	0.029
Wine	0.943	0.943	0.943	0.943	**0.960**	0.954	0.954	0.943	0.943	0.943	**0.960**	0.943	**0.960**
	0.018	0.018	0.018	0.018	**0.011**	0.013	0.013	0.018	0.018	0.018	**0.011**	0.018	**0.011**
	0.949	0.949	0.949	0.949	**0.960**	0.960	0.954	0.949	0.949	0.949	0.954	0.949	**0.960**
	0.012	0.012	0.012	0.012	**0.011**	0.011	0.013	0.012	0.012	0.012	0.013	0.012	**0.011**
Glass	0.592	0.569	0.592	0.588	0.486	0.598	0.614	0.592	0.592	0.592	0.530	0.592	0.605
	0.037	0.040	0.037	0.036	0.036	0.039	0.032	0.037	0.037	0.037	0.028	0.037	0.036
	0.655	0.646	0.645	0.645	0.626	0.640	0.643	0.645	0.645	0.645	0.579	0.645	0.625
	0.026	0.025	0.032	0.033	0.031	0.028	0.034	0.032	0.032	0.032	0.035	0.032	0.032
Thyroid	0.907	0.907	0.907	0.907	**0.921**	**0.921**	0.911	0.907	0.907	0.907	**0.921**	0.907	**0.921**
	0.026	0.026	0.026	0.026	**0.027**	**0.027**	0.026	0.026	0.026	0.026	**0.027**	0.026	**0.027**
	0.898	0.898	0.898	0.898	**0.921**	**0.921**	0.911	0.898	0.898	0.898	**0.921**	0.898	**0.921**
	0.025	0.025	0.025	0.025	**0.027**	**0.027**	0.026	0.025	0.025	0.025	**0.027**	0.025	**0.027**
Vowel	0.382	0.279	0.441	0.430	0.362	0.396	0.439	0.443	0.443	0.451	0.405	0.449	0.431
	0.020	0.012	0.022	0.025	0.023	0.024	0.020	0.021	0.021	0.024	0.022	0.025	0.019
	0.443	0.373	0.452	0.441	0.449	0.465	0.452	0.454	0.454	0.441	0.472	0.441	**0.481**
	0.023	0.021	0.025	0.023	0.031	0.029	0.023	0.026	0.026	0.024	0.027	0.024	**0.027**
Balance	0.425	0.425	0.801	0.801	0.481	0.639	0.785	0.801	0.801	0.788	0.710	0.788	0.735
	0.020	0.020	0.040	0.040	0.027	0.048	0.037	0.040	0.040	0.051	0.050	0.051	0.052
	0.504	0.504	0.504	0.504	0.730	0.721	0.800	0.504	0.504	**0.809**	0.756	**0.809**	0.756
	0.061	0.061	0.061	0.061	0.079	0.079	0.077	0.061	0.061	**0.082**	0.077	**0.082**	0.077
Yeast	0.433	0.402	0.435	0.393	0.421	0.415	0.345	0.435	0.435	0.395	0.401	0.402	0.403
	0.021	0.011	0.020	0.008	0.014	0.015	0.008	0.019	0.019	0.013	0.020	0.016	0.018
	0.435	0.408	0.436	0.454	0.429	0.425	**0.464**	0.435	0.435	0.447	0.413	0.447	0.425
	0.012	0.011	0.012	0.013	0.014	0.013	**0.010**	0.012	0.012	0.014	0.015	0.014	0.014
Satimage	0.795	0.766	0.814	0.639	0.766	0.787	0.637	0.814	0.814	0.673	0.638	0.656	0.705
	0.019	0.018	0.018	0.014	0.025	0.026	0.021	0.018	0.018	0.025	0.034	0.015	0.030
	0.789	0.776	0.814	0.807	0.814	0.820	0.829	0.814	0.814	0.818	0.832	0.818	**0.833**
	0.019	0.017	0.021	0.020	0.019	0.019	0.017	0.021	0.021	0.019	0.020	0.019	**0.020**
Letter	0.803	0.821	0.823	0.841	0.821	0.828	0.812	0.834	0.838	0.848	0.855	0.862	0.880
	0.016	0.016	0.018	0.018	0.017	0.017	0.017	0.017	0.015	0.014	0.015	0.015	0.016
	0.839	0.840	0.850	0.863	0.836	0.845	0.834	0.860	0.876	0.872	0.885	0.874	**0.889**
	0.016	0.018	0.016	0.017	0.017	0.017	0.016	0.017	0.016	0.014	0.015	0.014	**0.015**
Pendigits	0.839	0.818	0.858	0.889	0.836	0.857	0.848	0.927	0.932	0.932	0.940	0.932	0.947
	0.011	0.010	0.009	0.010	0.009	0.007	0.011	0.010	0.010	0.008	0.009	0.007	0.010
	0.859	0.848	0.883	0.921	0.872	0.889	0.869	0.942	0.942	0.952	0.953	0.960	**0.967**
	0.010	0.010	0.010	0.010	0.009	0.007	0.011	0.009	0.011	0.007	0.008	0.006	**0.011**
Segment	0.921	0.922	0.921	0.891	0.711	0.863	0.920	0.921	0.921	0.927	0.865	0.928	0.925
	0.010	0.009	0.010	0.012	0.016	0.006	0.010	0.010	0.010	0.010	0.008	0.009	0.008
	0.939	0.933	0.938	0.933	0.897	0.919	0.938	0.939	0.938	0.938	0.935	0.938	**0.941**
	0.009	0.010	0.009	0.009	0.015	0.014	0.008	0.009	0.009	0.009	0.009	0.009	**0.009**
Optdigits	0.753	0.716	0.796	0.740	0.787	0.795	0.783	0.795	0.796	0.769	0.794	0.773	0.809
	0.018	0.016	0.022	0.023	0.026	0.025	0.024	0.023	0.022	0.021	0.025	0.021	0.020
	0.769	0.651	0.811	0.779	0.685	0.724	0.810	0.811	0.811	**0.815**	0.772	**0.815**	0.803
	0.022	0.016	0.025	0.030	0.018	0.019	0.023	0.026	0.026	**0.023**	0.025	**0.023**	0.024
Shuttle	0.658	0.703	0.702	0.710	0.653	0.669	0.716	0.702	0.702	0.702	0.691	0.702	0.699
	0.023	0.024	0.023	0.036	0.021	0.022	0.027	0.023	0.023	0.023	0.019	0.023	0.019
	0.723	0.724	0.723	0.724	0.730	**0.734**	0.727	0.723	0.723	0.727	0.730	0.729	0.730
	0.033	0.029	0.033	0.032	0.031	**0.029**	0.025	0.033	0.033	0.033	0.034	0.032	0.030
Vehicle	0.850	0.849	**0.998**	**0.998**	0.854	0.859	0.944	**0.998**	**0.998**	**0.998**	0.936	**0.998**	0.990
	0.000	0.000	**0.000**	**0.000**	0.002	0.004	0.008	**0.000**	**0.000**	**0.000**	0.022	**0.000**	0.006
	0.998	0.989	**0.998**	**0.998**	0.779	0.957	0.853	**0.998**	**0.998**	**0.998**	0.817	**0.998**	0.967
	0.000	0.003	**0.000**	**0.000**	0.067	0.020	0.152	**0.000**	**0.000**	**0.000**	0.078	**0.000**	0.021

a more complex weighting procedure in order to obtain more precise results.

Concerning to the decoding step, note that the use of the proposed ternary *distance* d_t makes sense in the case that two ternary

Table 3
Sparse random results using Linear *SVM* on the UCI data sets.

	HD	*IHD*	*ED*	*AED*	*LLB*	*ELB*	*PD*	*LAP*	*β-DEN*	*LLW* disc.	*LLW* cont.	*ELW* disc.	*ELW* cont.
Derma	0.374	0.382	0.440	0.868	0.456	0.623	0.853	0.440	0.719	0.870	0.766	0.870	0.835
	0.005	0.007	0.024	0.017	0.042	0.027	0.030	0.024	0.024	0.015	0.060	0.015	0.041
	0.936	0.847	0.936	0.939	0.950	0.950	**0.961**	0.936	0.936	0.936	0.953	0.933	0.953
	0.011	0.028	0.011	0.011	0.010	0.010	**0.009**	0.011	0.011	0.011	0.010	0.012	0.010
Iris	0.666	0.666	**0.973**	**0.973**	0.666	0.920	0.773	**0.973**	**0.973**	**0.973**	0.780	**0.973**	0.933
	0.010	0.010	**0.010**	**0.010**	0.010	0.019	0.019	**0.010**	**0.010**	**0.010**	0.023	**0.010**	0.016
	0.720	0.720	0.720	0.720	0.926	0.926	0.826	0.720	0.720	**0.973**	0.940	**0.973**	0.933
	0.025	0.025	0.025	0.025	0.020	0.020	0.022	0.025	0.025	**0.010**	0.020	**0.010**	0.021
Ecoli	0.684	0.269	0.726	0.723	0.699	0.743	0.275	0.726	0.726	0.683	0.413	0.613	0.411
	0.022	0.025	0.031	0.030	0.026	0.031	0.029	0.031	0.031	0.038	0.033	0.058	0.042
	0.758	0.726	0.758	0.737	**0.770**	0.761	0.766	0.758	0.758	0.667	0.616	0.711	0.610
	0.026	0.029	0.026	0.023	**0.029**	0.027	0.031	0.026	0.026	0.037	0.044	0.026	0.047
Wine	0.932	0.932	0.932	0.932	**0.955**	**0.955**	0.949	0.932	0.932	0.932	**0.955**	0.932	**0.955**
	0.016	0.016	0.016	0.016	**0.013**	**0.013**	0.009	0.016	0.016	0.016	**0.013**	0.016	**0.013**
	0.932	0.932	0.932	0.932	**0.955**	**0.955**	0.949	0.932	0.932	0.932	**0.955**	0.932	**0.955**
	0.016	0.016	0.016	0.016	**0.013**	**0.013**	0.009	0.016	0.016	0.016	**0.013**	0.016	**0.013**
Glass	0.438	0.460	0.446	0.457	0.428	0.456	0.452	0.452	0.446	0.443	0.457	0.427	0.458
	0.024	0.029	0.033	0.023	0.025	0.029	0.022	0.028	0.033	0.021	0.030	0.031	0.033
	0.503	0.509	0.503	0.484	**0.552**	0.523	0.524	0.503	0.503	0.504	0.517	0.504	0.534
	0.028	0.019	0.028	0.029	**0.028**	0.030	0.028	0.028	0.028	0.043	0.031	0.043	0.036
Thyroid	0.814	0.814	0.943	0.943	0.818	**0.948**	0.897	0.943	0.943	0.943	0.856	0.943	**0.948**
	0.009	0.009	0.021	0.021	0.007	**0.015**	0.023	0.021	0.021	0.021	0.011	0.021	**0.020**
	0.916	0.916	0.916	0.916	0.934	0.934	0.855	0.916	0.916	0.943	0.934	0.943	0.934
	0.026	0.026	0.026	0.026	0.025	0.025	0.019	0.026	0.026	0.021	0.025	0.021	0.025
Vowel	0.343	0.308	0.341	0.301	0.313	0.298	0.241	0.352	0.352	0.357	0.328	0.359	0.360
	0.018	0.018	0.017	0.021	0.021	0.020	0.025	0.018	0.018	0.025	0.029	0.025	0.023
	0.382	0.376	0.366	0.314	0.365	0.367	0.269	0.360	0.362	0.368	0.334	0.364	0.379
	0.026	0.029	0.018	0.010	0.027	0.022	0.018	0.016	0.017	0.018	0.021	0.020	0.015
Balance	**0.855**	**0.855**	**0.855**	**0.855**	0.833	0.833	0.822	**0.855**	**0.855**	**0.855**	0.845	**0.855**	**0.855**
	0.041	**0.041**	**0.041**	**0.041**	0.035	0.035	0.040	**0.041**	**0.041**	**0.041**	0.045	**0.041**	**0.041**
	0.855	**0.855**	**0.855**	**0.855**	0.833	0.833	0.822	**0.855**	**0.855**	**0.855**	0.845	**0.855**	**0.855**
	0.041	**0.041**	**0.041**	**0.041**	0.035	0.035	0.040	**0.041**	**0.041**	**0.041**	0.045	**0.041**	**0.041**
Yeast	0.380	0.385	0.380	0.378	0.379	0.390	0.217	0.380	0.381	0.341	0.210	0.346	0.221
	0.012	0.013	0.012	0.013	0.016	0.012	0.005	0.012	0.013	0.017	0.009	0.021	0.006
	0.491	0.476	0.493	0.489	0.492	0.495	**0.506**	0.493	0.493	0.484	0.472	0.483	0.497
	0.018	0.015	0.019	0.018	0.014	0.019	**0.016**	0.018	0.018	0.024	0.024	0.024	0.031
Satimage	0.718	0.710	0.726	0.615	0.670	0.725	0.634	0.726	0.726	0.618	0.403	0.638	0.660
	0.016	0.014	0.014	0.019	0.048	0.019	0.021	0.014	0.014	0.019	0.028	0.030	0.021
	0.724	0.626	0.734	0.732	0.656	0.750	0.739	0.734	0.734	0.776	0.489	0.776	**0.782**
	0.014	0.009	0.013	0.014	0.014	0.011	0.012	0.013	0.013	0.017	0.037	0.017	**0.018**
Letter	0.632	0.637	0.648	0.662	0.648	0.652	0.643	0.671	0.672	0.702	0.705	0.703	0.710
	0.010	0.009	0.008	0.009	0.014	0.009	0.010	0.011	0.009	0.010	0.008	0.009	0.009
	0.642	0.653	0.663	0.675	0.649	0.660	0.648	0.678	0.692	0.708	0.717	0.718	**0.729**
	0.010	0.009	0.007	0.008	0.006	0.007	0.008	0.010	0.008	0.007	0.008	0.008	**0.010**
Pendigits	0.878	0.887	0.893	0.912	0.902	0.902	0.897	0.913	0.916	0.917	0.921	0.918	0.927
	0.008	0.007	0.008	0.009	0.009	0.010	0.009	0.008	0.009	0.008	0.014	0.013	0.013
	0.897	0.908	0.917	0.932	0.910	0.912	0.901	0.938	0.939	0.941	0.944	0.948	**0.953**
	0.010	0.009	0.008	0.009	0.014	0.013	0.011	0.010	0.009	0.008	0.014	0.009	**0.010**
Segment	0.706	0.627	0.838	0.800	0.475	0.849	0.716	0.837	0.837	0.837	0.700	0.837	0.851
	0.013	0.015	0.006	0.005	0.010	0.007	0.003	0.007	0.007	0.007	0.030	0.007	0.007
	0.793	0.727	0.800	0.810	0.727	0.843	0.751	0.840	0.840	0.844	0.791	0.844	**0.856**
	0.005	0.012	0.005	0.006	0.010	0.007	0.006	0.005	0.005	0.006	0.014	0.006	**0.007**
Optdigits	0.710	0.664	0.767	0.738	0.616	0.763	0.719	0.769	0.769	0.769	0.620	0.768	0.813
	0.019	0.021	0.017	0.021	0.029	0.012	0.020	0.016	0.016	0.015	0.047	0.018	0.024
	0.795	0.573	0.797	0.785	0.664	0.845	0.832	0.797	0.797	0.812	0.719	0.812	**0.847**
	0.030	0.016	0.029	0.027	0.021	0.030	0.025	0.029	0.029	0.031	0.023	0.031	**0.030**
Shuttle	0.520	0.722	0.703	0.702	0.670	0.730	0.620	0.703	0.703	0.703	0.704	0.703	0.728
	0.011	0.016	0.022	0.022	0.026	0.017	0.029	0.022	0.022	0.022	0.025	0.022	0.014
	0.728	0.728	0.728	0.730	0.742	0.781	0.763	0.728	0.728	0.736	0.751	0.736	**0.776**
	0.025	0.025	0.025	0.027	0.026	0.018	0.018	0.025	0.025	0.025	0.026	0.025	**0.021**
Vehicle	**0.977**	**0.977**	**0.977**	**0.977**	0.892	**0.977**	0.969	**0.977**	**0.977**	**0.977**	**0.977**	**0.977**	**0.977**
	0.003	**0.003**	**0.003**	**0.003**	0.014	**0.003**	0.007	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**
	0.977	**0.977**	**0.977**	**0.977**	0.902	**0.977**	0.971	**0.977**	**0.977**	**0.977**	**0.977**	**0.977**	**0.977**
	0.003	**0.003**	**0.003**	**0.003**	0.013	**0.003**	0.008	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**

codewords are compared (as in the case of comparing two codewords of two classes), since the terms $|y_1^j|$ and $|y_2^j|$ may take the one and zero values. However, the test codeword takes binary values, and thus, the use of the factor $|y_2^j|$ does not make sense at the

Table 4
Sparse random and dense random results using Gentle Adaboost on the UCI data sets.

	HD	*IHD*	*ED*	*AED*	*LLB*	*ELB*	*PD*	*LAP*	*β-DEN*	*LLW* disc.	*LLW* cont.	*ELW* disc.	*ELW* cont.
Derma	0.910	0.923	0.910	0.910	0.940	0.940	0.926	0.910	0.910	0.915	0.937	0.915	0.940
	0.047	0.037	0.047	0.047	0.035	0.035	0.043	0.047	0.047	0.047	0.039	0.047	0.035
	0.926	0.923	0.926	0.923	0.896	0.926	**0.945**	0.926	0.926	0.929	0.920	0.929	0.940
	0.017	0.018	0.017	0.015	0.024	0.021	**0.013**	0.017	0.017	0.017	0.015	0.017	0.015
Iris	0.933	0.933	0.933	0.933	0.953	0.953	0.953	0.933	0.933	0.933	0.953	0.933	0.953
	0.043	0.043	0.043	0.043	0.027	0.027	0.027	0.043	0.043	0.043	0.027	0.043	0.027
	0.926	0.926	0.926	0.926	**0.960**	**0.960**	**0.960**	0.926	0.926	0.933	**0.960**	0.933	**0.960**
	0.020	0.020	0.020	0.020	**0.014**	**0.014**	**0.014**	0.020	0.020	0.019	**0.014**	0.019	**0.014**
Ecoli	0.373	0.379	0.367	0.533	0.284	0.302	0.493	0.370	0.357	0.539	0.443	**0.551**	0.477
	0.017	0.016	0.021	0.014	0.019	0.020	0.017	0.018	0.021	0.015	0.033	**0.011**	0.0293
	0.373	0.379	0.367	0.533	0.284	0.302	0.493	0.370	0.357	0.539	0.443	**0.551**	0.477
	0.017	0.016	0.021	0.014	0.019	0.020	0.017	0.018	0.021	0.015	0.033	**0.011**	0.029
Wine	0.949	0.949	0.949	0.949	**0.960**	0.954	0.954	0.949	0.949	0.949	**0.960**	0.949	**0.960**
	0.025	0.025	0.025	0.025	**0.023**	0.027	0.027	0.025	0.025	0.025	**0.023**	0.025	**0.023**
	0.949	0.949	0.949	0.949	**0.960**	**0.960**	0.954	0.949	0.949	0.949	0.954	0.949	**0.960**
	0.012	0.012	0.012	0.012	**0.011**	**0.011**	0.013	0.012	0.012	0.012	0.013	0.012	**0.011**
Glass	0.560	0.451	0.560	0.560	0.578	0.583	0.577	0.560	0.560	0.527	0.578	0.532	0.578
	0.099	0.106	0.099	0.099	0.085	0.085	0.094	0.099	0.099	0.080	0.079	0.085	0.079
	0.655	0.646	**0.645**	**0.645**	0.626	0.640	0.643	**0.645**	**0.645**	**0.645**	0.579	**0.645**	0.625
	0.026	0.025	**0.032**	**0.033**	0.031	0.028	0.034	**0.032**	**0.032**	**0.032**	0.035	**0.032**	0.032
Thyroid	0.907	0.907	0.907	0.907	**0.921**	**0.921**	0.911	0.907	0.907	0.907	**0.921**	0.907	**0.921**
	0.052	0.052	0.052	0.052	**0.054**	**0.054**	0.053	0.052	0.052	0.052	**0.054**	0.052	**0.054**
	0.898	0.898	0.898	0.898	**0.921**	**0.921**	0.911	0.898	0.898	0.898	**0.921**	0.898	**0.921**
	0.025	0.025	0.025	0.025	**0.027**	**0.027**	0.026	0.025	0.025	0.025	**0.027**	0.025	**0.027**
Vowel	0.274	0.241	0.274	0.274	0.323	0.332	0.315	0.274	0.274	0.297	0.332	0.297	0.331
	0.041	0.037	0.041	0.041	0.045	0.047	0.045	0.041	0.041	0.041	0.049	0.041	0.048
	0.443	0.373	0.452	0.441	0.449	0.465	0.452	0.454	0.454	0.441	0.472	0.441	**0.481**
	0.023	0.021	0.025	0.023	0.031	0.029	0.023	0.026	0.026	0.024	0.027	0.024	**0.027**
Balance	0.504	0.504	0.504	0.504	0.730	0.721	0.800	0.504	0.504	**0.809**	0.756	**0.809**	0.756
	0.123	0.123	0.123	0.123	0.159	0.159	0.155	0.123	0.123	**0.164**	0.154	**0.164**	0.154
	0.504	0.504	0.504	0.504	0.730	0.721	0.800	0.504	0.504	**0.809**	0.756	**0.809**	0.756
	0.061	0.061	0.061	0.061	0.079	0.079	0.077	0.061	0.061	**0.082**	0.077	**0.082**	0.077
Yeast	0.468	0.224	0.468	0.468	**0.481**	0.479	0.415	0.468	0.468	0.469	0.452	0.468	0.454
	0.026	0.031	0.026	0.026	**0.024**	0.024	0.025	0.026	0.026	0.036	0.026	0.036	0.026
	0.435	0.408	0.436	0.454	0.429	0.425	0.464	0.435	0.435	0.447	0.413	0.447	0.425
	0.012	0.011	0.012	0.013	0.014	0.013	0.010	0.012	0.012	0.014	0.015	0.014	0.014
Satimage	0.799	0.765	0.799	0.799	0.840	**0.842**	0.839	0.799	0.799	0.807	0.838	0.807	0.838
	0.049	0.048	0.049	0.049	0.038	**0.037**	0.036	0.049	0.049	0.047	0.039	0.047	0.040
	0.789	0.776	0.814	0.807	0.814	0.820	0.829	0.814	0.814	0.818	0.832	0.818	0.833
	0.019	0.017	0.021	0.020	0.019	0.019	0.017	0.021	0.021	0.019	0.020	0.019	0.020
Letter	0.843	0.833	0.845	0.845	0.837	0.845	0.827	0.857	0.882	0.878	**0.894**	0.885	0.907
	0.031	0.034	0.033	0.033	0.033	0.031	0.034	0.035	0.029	0.030	**0.036**	0.031	0.030
	0.839	0.840	0.850	0.863	0.836	0.845	0.834	0.860	0.876	0.872	0.885	0.874	0.889
	0.016	0.018	0.016	0.017	0.017	0.017	0.016	0.017	0.016	0.014	0.015	0.014	0.015
Pendigits	0.903	0.921	0.932	0.932	0.913	0.923	0.918	0.947	0.947	0.948	0.953	0.950	0.955
	0.019	0.024	0.022	0.022	0.020	0.015	0.020	0.020	0.018	0.017	0.017	0.013	0.020
	0.859	0.848	0.883	0.921	0.872	0.889	0.869	0.942	0.942	0.952	0.953	0.960	**0.967**
	0.010	0.010	0.010	0.010	0.009	0.007	0.011	0.009	0.011	0.007	0.008	0.006	**0.011**
Segment	0.930	0.934	0.930	0.930	0.945	0.945	0.942	0.930	0.930	0.939	**0.951**	0.939	**0.951**
	0.016	0.019	0.016	0.016	0.016	0.015	0.013	0.016	0.016	0.018	**0.017**	0.018	**0.017**
	0.939	0.933	0.938	0.933	0.897	0.919	0.938	0.939	0.938	0.938	0.935	0.938	0.941
	0.009	0.010	0.009	0.009	0.015	0.014	0.008	0.009	0.009	0.009	0.009	0.009	0.009
Optdigits	0.805	0.665	0.805	0.805	0.859	0.851	0.853	0.835	0.835	0.844	**0.868**	0.844	**0.868**
	0.020	0.019	0.020	0.020	0.015	0.014	0.016	0.020	0.020	0.021	**0.016**	0.021	**0.016**
	0.769	0.651	0.811	0.779	0.685	0.724	0.810	0.811	0.811	0.815	0.772	0.815	0.803
	0.022	0.016	0.025	0.030	0.018	0.019	0.023	0.026	0.026	0.023	0.025	0.023	0.024
Shuttle	0.640	0.656	0.640	0.640	0.717	0.721	0.725	0.640	0.640	0.714	0.724	0.714	0.723
	0.025	0.020	0.025	0.025	0.036	0.037	0.029	0.025	0.025	0.041	0.044	0.041	0.044
	0.723	0.724	0.723	0.724	0.730	**0.734**	0.727	0.723	0.723	0.727	0.730	0.729	0.730
	0.033	0.029	0.033	0.032	0.031	**0.029**	0.025	0.033	0.033	0.033	0.034	0.032	0.030
Vehicle	0.997	0.997	0.997	0.997	**0.998**	**0.998**	0.979	0.997	0.997	0.997	**0.998**	0.997	**0.998**
	0.001	0.001	0.001	0.001	**0.001**	**0.001**	0.031	0.001	0.001	0.001	**0.001**	0.001	**0.001**
	0.998	0.989	**0.998**	**0.998**	0.779	0.957	0.853	**0.998**	**0.998**	**0.998**	0.817	**0.998**	0.967
	0.000	0.003	**0.000**	**0.000**	0.067	0.020	0.152	**0.000**	**0.000**	**0.000**	0.078	**0.000**	0.021

decoding step. In this case, the *distance* should omit this factor, and becomes:

$$d(y,x) = \sum_{j=1}^{n} \frac{1}{2} |y^j| (1 - y^j x^j)$$

Table 5
Sparse random and dense random results using Linear *SVM* on the UCI data sets.

	HD	*IHD*	*ED*	*AED*	*LLB*	*ELB*	*PD*	*LAP*	*β-DEN*	*LLW* disc.	*LLW* cont.	*ELW* disc.	*ELW* cont.
Derma	0.808	0.306	0.808	0.808	0.814	0.814	0.808	0.808	0.808	0.808	0.814	0.808	0.814
	0.012	0.001	0.012	0.012	0.011	0.011	0.010	0.012	0.012	0.012	0.011	0.012	0.011
	0.936	0.847	0.936	**0.939**	0.950	0.950	0.961	0.936	0.936	0.936	0.953	0.933	0.953
	0.011	0.028	0.011	**0.011**	0.010	0.010	0.009	0.011	0.011	0.011	0.010	0.012	0.010
Iris	0.720	0.720	0.720	0.720	0.926	0.926	0.826	0.720	0.720	**0.973**	0.940	**0.973**	0.933
	0.025	0.025	0.025	0.025	0.020	0.020	0.022	0.025	0.025	**0.010**	0.020	**0.010**	0.021
	0.720	0.720	0.720	0.720	0.926	0.926	0.826	0.720	0.720	**0.973**	0.940	**0.973**	0.933
	0.025	0.025	0.025	0.025	0.020	0.020	0.022	0.025	0.025	**0.010**	0.020	**0.010**	0.021
Ecoli	0.681	0.721	0.681	0.681	0.764	0.764	**0.785**	0.681	0.681	0.715	0.704	0.677	0.703
	0.033	0.028	0.033	0.033	0.034	0.034	**0.018**	0.033	0.033	0.027	0.052	0.034	0.048
	0.758	0.726	0.758	0.737	0.770	0.761	0.766	0.758	0.758	0.667	0.616	0.711	0.610
	0.026	0.029	0.026	0.023	0.029	0.027	0.031	0.026	0.026	0.037	0.044	0.026	0.047
Wine	0.932	0.932	0.932	0.932	**0.955**	**0.955**	0.949	0.932	0.932	0.932	**0.955**	0.932	**0.955**
	0.016	0.016	0.016	0.016	**0.013**	**0.013**	0.009	0.016	0.016	0.016	**0.013**	0.016	**0.013**
	0.932	0.932	0.932	0.932	**0.955**	**0.955**	0.949	0.932	0.932	0.932	**0.955**	0.932	**0.955**
	0.016	0.016	0.016	0.016	**0.013**	**0.013**	0.009	0.016	0.016	0.016	**0.013**	0.016	**0.013**
Glass	0.400	0.327	0.400	0.400	0.349	0.349	0.334	0.400	0.400	0.377	0.414	0.373	0.415
	0.030	0.015	0.030	0.030	0.032	0.032	0.054	0.030	0.030	0.028	0.038	0.037	0.034
	0.503	0.509	0.503	0.484	**0.552**	0.523	0.524	0.503	0.503	0.504	0.517	0.504	0.534
	0.028	0.019	0.028	0.029	**0.028**	0.030	0.028	0.028	0.028	0.043	0.031	0.043	0.036
Thyroid	0.916	0.916	0.916	0.916	0.934	0.934	0.855	0.916	0.916	**0.943**	0.934	**0.943**	0.934
	0.026	0.026	0.026	0.026	0.025	0.025	0.019	0.026	0.026	**0.021**	0.025	**0.021**	0.025
	0.916	0.916	0.916	0.916	0.934	0.934	0.855	0.916	0.916	**0.943**	0.934	**0.943**	0.934
	0.026	0.026	0.026	0.026	0.025	0.025	0.019	0.026	0.026	**0.021**	0.025	**0.021**	0.025
Vowel	0.281	0.228	0.281	0.281	0.332	0.329	0.324	0.281	0.281	0.311	**0.385**	0.311	0.362
	0.031	0.016	0.031	0.031	0.025	0.027	0.023	0.031	0.031	0.030	**0.029**	0.030	0.023
	0.382	0.376	0.366	0.314	0.365	0.367	0.269	0.360	0.362	0.368	0.334	0.364	0.379
	0.026	0.029	0.018	0.010	0.027	0.022	0.018	0.016	0.017	0.018	0.021	0.020	0.015
Balance	**0.855**	**0.855**	**0.855**	**0.855**	0.833	0.833	0.822	**0.855**	**0.855**	**0.855**	0.845	**0.855**	**0.855**
	0.041	**0.041**	**0.041**	**0.041**	0.035	0.035	0.040	**0.041**	**0.041**	**0.041**	0.045	**0.041**	**0.041**
	0.855	**0.855**	**0.855**	**0.855**	0.833	0.833	0.822	**0.855**	**0.855**	**0.855**	0.845	**0.855**	**0.855**
	0.041	**0.041**	**0.041**	**0.041**	0.035	0.035	0.040	**0.041**	**0.041**	**0.041**	0.045	**0.041**	**0.041**
Yeast	0.278	0.169	0.278	0.278	0.265	0.266	0.336	0.278	0.278	0.405	0.451	0.407	0.448
	0.015	0.016	0.015	0.015	0.015	0.014	0.012	0.015	0.015	0.012	0.017	0.010	0.017
	0.491	0.476	0.493	0.489	0.492	0.495	**0.506**	0.493	0.493	0.484	0.472	0.483	0.497
	0.018	0.015	0.019	0.018	0.014	0.019	**0.016**	0.018	0.018	0.024	0.024	0.024	0.031
Satimage	0.720	0.702	0.720	0.720	0.773	0.774	0.748	0.720	0.720	**0.795**	0.617	**0.795**	0.806
	0.012	0.018	0.012	0.012	0.014	0.014	0.009	0.012	0.012	**0.026**	0.024	**0.026**	0.025
	0.724	0.626	0.734	0.732	0.656	0.750	0.739	0.734	0.734	0.776	0.489	0.776	0.782
	0.014	0.009	0.013	0.014	0.014	0.011	0.012	0.013	0.013	0.017	0.037	0.017	0.018
Letter	0.657	0.667	0.672	0.672	0.652	0.662	0.658	0.687	0.703	0.712	0.718	0.716	0.721
	0.008	0.008	0.008	0.008	0.008	0.008	0.009	0.010	0.009	0.009	0.009	0.009	0.009
	0.642	0.653	0.663	0.675	0.649	0.660	0.648	0.678	0.692	0.708	0.717	0.718	**0.729**
	0.010	0.009	0.007	0.008	0.006	0.007	0.008	0.010	0.008	0.007	0.008	0.008	**0.010**
Pendigits	0.921	0.917	0.932	0.932	0.917	0.921	0.903	0.936	0.938	0.936	0.940	0.942	0.956
	0.009	0.010	0.009	0.009	0.010	0.010	0.010	0.009	0.009	0.008	0.011	0.014	0.012
	0.897	0.908	0.917	0.932	0.910	0.912	0.901	0.938	0.939	0.941	0.944	0.948	**0.953**
	0.010	0.009	0.008	0.009	0.014	0.013	0.011	0.010	0.009	0.008	0.014	0.009	**0.010**
Segment	0.818	0.807	0.818	0.818	0.851	0.848	0.732	0.818	0.818	0.857	**0.867**	0.857	0.865
	0.006	0.005	0.006	0.006	0.006	0.007	0.006	0.006	0.006	0.006	**0.005**	0.006	0.007
	0.793	0.727	0.800	0.810	0.727	0.843	0.751	0.840	0.840	0.844	0.791	0.844	0.856
	0.005	0.012	0.005	0.006	0.010	0.007	0.006	0.005	0.005	0.006	0.014	0.006	0.007
Optdigits	0.820	0.766	0.820	0.820	0.874	0.876	0.872	0.820	0.820	0.845	**0.879**	0.845	0.877
	0.028	0.031	0.028	0.028	0.027	0.028	0.025	0.028	0.028	0.029	**0.029**	0.029	0.027
	0.795	0.573	0.797	0.785	0.664	0.845	0.832	0.797	0.797	0.812	0.719	0.812	0.847
	0.030	0.016	0.029	0.027	0.021	0.030	0.025	0.029	0.029	0.031	0.023	0.031	0.030
Shuttle	0.712	0.744	0.712	0.712	0.748	0.749	0.742	0.712	0.712	0.737	0.742	0.737	0.747
	0.029	0.021	0.029	0.029	0.018	0.017	0.018	0.029	0.029	0.027	0.027	0.027	0.020
	0.728	0.728	0.728	0.730	0.742	**0.781**	0.763	0.728	0.728	0.736	0.751	0.736	0.776
	0.025	0.025	0.025	0.027	0.026	**0.018**	0.018	0.025	0.025	0.025	0.026	0.025	0.021
Vehicle	0.962	0.956	**0.977**	**0.977**	0.909	0.947	0.939	**0.977**	**0.977**	**0.977**	**0.977**	**0.977**	**0.977**
	0.006	0.004	**0.003**	**0.003**	0.010	0.008	0.008	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**
	0.977	**0.977**	**0.977**	**0.977**	0.902	**0.977**	0.971	**0.977**	**0.977**	**0.977**	**0.977**	**0.977**	**0.977**
	0.003	**0.003**	**0.003**	**0.003**	0.013	**0.003**	0.008	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**	**0.003**

4. Conclusions

In this paper, we introduced a new formulation of the ternary distance that defines the classes separability in the ternary ECOC framework. We showed that the rows separability in terms of the Hamming distance of the binary ECOC framework can not be applied in the ternary case. Based on the new measure, we illustrated that the design of the standard sparse random strategy is inconsistent, and a new sparse random construction is presented. The results show that the new design applied with any state-of-the-art decoding strategy outperforms the classical approach. The results on a wide set of UCI Machine Learning Repository data sets and in a real speed traffic sign Computer Vision categorization problem show that when the coding designs satisfy the new ternary measures, significant performance improvements are obtained independently of the decoding strategy applied.

Acknowledgments

This work has been supported in part by projects TIN2006-15308-C02, FIS PI061290, and CONSOLIDER-INGENIO CSD 2007-00018.

Appendix A. Sparse random performances on UCI data sets

Tables 2 and 3 show the performance results on the UCI data sets for the sparse random designs using Gentle Adaboost and Linear *SVM*, respectively. For each data set shown in Tables 2 and 3, the results on the top correspond to the performance and confidence interval using the classical sparse random strategy. The results on the bottom correspond to the results using the sparse random selection based on maximizing the new ternary distance. The best results for each data set are marked in bold. Note that

Table 6
Classical sparse random results (performances on the top of each data set) and sparse random with ternary distance maximization (performances on the bottom of each data set) using Gentle Adaboost and Linear *SVM* on the speed traffic sign data set.

	HD	*IHD*	*ED*	*AED*	*LLB*	*ELB*	*PD*	*LAP*	*β-DEN*	*LLW* disc.	*LLW* cont.	*ELW* disc.	*ELW* cont.
Adaboost	0.526	0.483	0.516	0.514	0.404	0.430	0.561	0.524	0.526	0.528	0.450	0.539	0.492
	0.041	0.043	0.047	0.044	0.031	0.029	0.055	0.047	0.047	0.044	0.035	0.041	0.039
	0.539	0.508	0.557	0.537	0.533	0.553	0.570	0.547	0.547	0.546	0.551	0.548	**0.564**
	0.030	0.034	0.029	0.028	0.037	0.032	0.031	0.027	0.027	0.033	0.041	0.030	**0.038**
SVM	0.629	0.531	0.605	0.633	0.656	0.662	0.650	0.640	0.640	0.648	0.661	0.642	0.678
	0.048	0.048	0.054	0.049	0.053	0.058	0.043	0.055	0.056	0.055	0.045	0.056	0.057
	0.668	0.619	0.646	0.675	0.656	0.697	0.659	0.706	0.706	0.706	0.669	0.706	**0.711**
	0.035	0.041	0.036	0.032	0.045	0.029	0.031	0.036	0.036	0.036	0.035	0.035	**0.029**

Table 7
UCI classification performances using a tuned *RBF SVM*. For each data set from top to bottom: performance and confidence interval using the classical one-versus-all design, and performance and confidence interval using the new sparse random selection based on maximizing the new ternary distance.

	HD	*IHD*	*ED*	*AED*	*LLB*	*ELB*	*PD*	*LAP*	*β-DEN*	*LLW* disc.	*LLW* cont.	*ELW* disc.	*ELW* cont.
Derma	0.961	0.961	0.961	0.961	0.961	0.961	0.963	**0.968**	**0.968**	**0.968**	**0.968**	**0.968**	**0.968**
	0.009	0.009	0.009	0.009	0.009	0.009	0.010	**0.010**	**0.010**	**0.010**	**0.010**	**0.010**	**0.010**
	0.961	0.961	0.961	0.961	0.961	0.961	**0.968**	**0.968**	**0.968**	**0.968**	**0.968**	**0.968**	**0.968**
	0.009	0.009	0.009	0.009	0.009	0.009	**0.010**	**0.010**	**0.010**	**0.010**	**0.010**	**0.010**	**0.010**
Iris	**0.973**	**0.973**	**0.973**	**0.973**	**0.973**	**0.973**	0.966	**0.973**	**0.973**	**0.973**	**0.973**	**0.973**	**0.973**
	0.021	**0.021**	**0.021**	**0.021**	**0.021**	**0.021**	0.021	**0.021**	**0.021**	**0.021**	**0.021**	**0.021**	**0.021**
	0.973	**0.973**	**0.973**	**0.973**	**0.973**	**0.973**	0.966	**0.973**	**0.973**	**0.973**	**0.973**	**0.973**	**0.973**
	0.021	**0.021**	**0.021**	**0.021**	**0.021**	**0.021**	0.021	**0.021**	**0.021**	**0.021**	**0.021**	**0.021**	**0.021**
Ecoli	0.839	0.848	0.839	0.839	0.848	0.848	0.864	0.839	0.839	0.851	0.861	0.848	0.858
	0.037	0.041	0.037	0.037	0.038	0.038	0.045	0.037	0.037	0.036	0.042	0.038	0.043
	0.866	0.866	0.866	0.866	0.858	0.865	**0.873**	0.866	0.866	0.866	0.862	0.866	0.865
	0.036	0.041	0.037	0.037	0.039	0.028	**0.029**	0.036	0.036	0.036	0.036	0.036	0.024
Wine	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**
	0.013	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**
	0.955	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**	**0.955**
	0.013	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**	**0.013**
Glass	0.647	0.645	0.647	0.647	0.669	0.674	0.643	0.647	0.647	0.646	0.654	0.646	0.668
	0.084	0.080	0.084	0.084	0.071	0.078	0.085	0.084	0.084	0.078	0.066	0.078	0.075
	0.686	0.68	0.691	0.691	0.692	0.692	0.665	0.691	0.691	**0.695**	0.664	**0.695**	0.673
	0.077	0.073	0.081	0.088	0.077	0.077	0.091	0.081	0.081	**0.077**	0.077	**0.077**	0.072
Thyroid	**0.943**	0.938	**0.943**	**0.943**	0.938	0.938	0.938	**0.943**	**0.943**	**0.943**	0.938	**0.943**	**0.943**
	0.021	0.021	**0.021**	**0.021**	0.021	0.021	0.021	**0.021**	**0.021**	**0.021**	0.021	**0.021**	**0.021**
	0.943	0.938	**0.943**	**0.943**	0.938	0.938	0.938	**0.943**	**0.943**	**0.943**	0.938	**0.943**	**0.943**
	0.021	0.021	**0.021**	**0.021**	0.021	0.021	0.021	**0.021**	**0.021**	**0.021**	0.021	**0.021**	**0.021**
Vowel	0.807	0.775	0.807	0.807	0.844	0.844	**0.855**	0.807	0.807	0.807	0.844	0.807	0.844
	0.037	0.047	0.037	0.037	0.046	0.046	**0.046**	0.037	0.037	0.037	0.046	0.037	0.046
	0.846	0.833	0.837	0.822	0.833	0.848	0.814	0.837	0.837	0.837	0.851	0.837	0.848
	0.047	0.048	0.044	0.036	0.043	0.040	0.0510	0.0449	0.0449	0.0449	0.039	0.044	0.039
Balance	0.878	0.846	0.878	0.878	0.873	0.873	0.872	0.878	0.878	**0.897**	0.871	**0.897**	0.871
	0.062	0.082	0.062	0.062	0.078	0.078	0.041	0.062	0.062	**0.068**	0.077	**0.068**	0.078
	0.884	0.879	0.884	0.884	0.865	0.865	0.870	0.884	0.884	0.881	0.847	0.881	0.868
	0.072	0.071	0.072	0.072	0.045	0.075	0.055	0.072	0.072	0.070	0.075	0.070	0.070

in most cases, the new sparse design outperforms the results of the classical one. Only in few cases, such as at the Satimage data set with *SVM* or the Iris data set with Adaboost, there are some performances inferior to the classical approach.

Appendix B. Sparse and dense random performances on UCI data sets

Tables 4 and 5 show the performance results on the UCI data sets for the dense random designs using Gentle Adaboost and Linear *SVM*, respectively. For each data set shown in Tables 4 and 5, the results on the top correspond to the performance and confidence interval using the classical dense random strategy. The results on the bottom correspond to the results using the sparse random selection based on maximizing the new ternary distance. The best results for each data set are marked in bold. Note that in most cases, the new sparse design outperforms the results of the classical dense random.

Appendix C. Sparse random performances on speed traffic sign data set

Table 6 shows the performance results on the speed traffic data set for the sparse random designs using Gentle Adaboost and Linear *SVM*, respectively. The results on the top correspond to the performance and confidence interval using the classical sparse random strategy. The results on the bottom correspond to the results using the sparse random selection based on maximizing the new ternary distance. The best results for each data set are marked in bold. Note that almost all cases, the results obtained by the new sparse designs outperform the performances obtained by the classical approach.

Appendix D. Sparse random performances on UCI data sets using *RBF SVM*

Table 7 shows the performance results on the UCI data sets for the new sparse random and one-versus-all designs using *RBF SVM* optimized via cross-validation. For each data set shown in Table 7, the results on the top correspond to the performance and confidence interval using the classical one-versus-all design. The results on the bottom correspond to the results using the new sparse random selection based on maximizing the new ternary distance. The best results for each data set are marked in bold. Note that in most cases, the new sparse design outperforms the results of the classical one-versus-all strategy.

References

Allwein, E., Schapire, R., Singer, Y., 2002. Reducing multiclass to binary: A unifying approach for margin classifiers. In: JMLR, vol 1, pp. 113–141.

Asuncion, A., Newman, D., 2007. In: UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences. <http://mlearn.ics.uci.edu/MLRepository.html>.

Casacuberta, J., Miranda, J., Pla, M., Sanchez, S., Serra, A., Talaya, J., 2004. On the accuracy and performance of the GeoMobil system. In: Internat. Society for Photogrammetry and Remote Sensing.

Dietterich, T., Bakiri, G., 1995. Solving multiclass learning problems via error-correcting output codes. J. Artif. Intell. Res. 2, 263–282.

Escalera, S., Pujol, O., Radeva, P., 2006. Decoding of ternary error correcting output codes. In: CIARP, vol. 4225, pp. 753–763.

Escalera, S., Pujol, O., Radeva, P., 2007. Boosted landmarks of contextual descriptors and Forest-ECOC: A novel framework to detect and classify objects in clutter scenes. Pattern Recognition Lett. 28 (13), 1759–1768.

Escalera, S., Pujol, O., Radeva, P., 2008. Loss-weighted decoding for error-correcting output codes. In: Internat. Conf. on Computer Vision Theory and Applications, vol. 2, pp. 117–122.

Friedman, J., Hastie, T., Tibshirani, R., 1998. Additive logistic regression: A statistical view of boosting. Ann. Statist. 38, 337–374.

Ghani, R., 2001. Combining labeled and unlabeled data for text classification with a large number of categories. In: Internat. Conf. Data Mining, pp. 597–598.

Hastie, T., Tibshirani, R., 1998. Classification by pairwise grouping. In: NIPS, vol. 26, 451–471.

Kittler, J., Ghaderi, R., Windeatt, T., Matas, J., 2001. Face verification using error correcting output codes. In: CVPR, vol. 1, pp. 755–760.

Kong, E.B., Dietterich, T.G., 1995. Error-correcting output coding corrects bias and variance. In: ICML, pp. 313–321.

Nilsson, N.J., 1965. Learning Machines. McGraw-Hill.

Passerini, A., Pontil, M., Frasconi, P., 2004. New results on error correcting output codes of kernel machines. IEEE Trans. Neural Networks 15 (1), 45–54.

Pujol, O., Radeva, P., Vitrià, J., 2006. Discriminant ECOC: A heuristic method for application dependent design of error correcting output codes. In: PAMI, vol. 28, pp. 1001–1007.

Rifkin, R., Klautau, A., 2004. In defense of one-vs-all classification. J. Machine Learn. Res. 5, 101–141.

Vapnik, V., 1995. The Nature of Statistical Learning Theory. Springer.

Windeatt, T., Ardeshir, G., 2003. Boosted ECOC ensembles for face recognition. In: Internat. Conf. on Visual Information Engineering, pp. 165–168.

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

Zhou, J., Suen, C., 2005. Unconstrained numeral pair recognition using enhanced error correcting output coding: A holistic approach. In: Proc. Conf. on Document Analysis and Recognition, vol. 1, pp. 484–488.