On the Decoding Process in Ternary Error-Correcting Output Codes

Sergio Escalera, Oriol Pujol, and Petia Radeva

Abstract—A common way to model multiclass classification problems is to design a set of binary classifiers and to combine them. Error-Correcting Output Codes (ECOC) represent a successful framework to deal with these type of problems. Recent works in the ECOC framework showed significant performance improvements by means of new problem-dependent designs based on the ternary ECOC framework. The ternary framework contains a larger set of binary problems because of the use of a "do not care" symbol that allows us to ignore some classes by a given classifier. However, there are no proper studies that analyze the effect of the new symbol at the decoding step. In this paper, we present a taxonomy that embeds all binary and ternary ECOC decoding strategies into four groups. We show that the zero symbol introduces two kinds of biases that require redefinition of the decoding design. A new type of decoding measure is proposed, and two novel decoding strategies are defined. We evaluate the state-of-the-art coding and decoding strategies over a set of UCI Machine Learning Repository data sets and into a real traffic sign categorization problem. The experimental results show that, following the new decoding strategies, the performance of the ECOC design is significantly improved.

Index Terms—Error-correcting output codes, decoding, multiclass classification, embedding of dichotomizers.

1 Introduction

Error-correcting Output Codes are a general framework to combine binary problems to address the multiclass problem [6]. It has been successfully applied to a wide range of applications, such as face recognition [13], face verification [14], text recognition [15], or manuscript digit classification [16].

The ECOC framework consists of two steps: a coding step, where a codeword[1] is assigned to each class, and a decoding technique, where, given a test sample, it looks for the most similar class codeword. The most well-known coding strategy is the one-versus-all [1], where each class is discriminated against the rest of classes. But, it was not until Allwein et al. in [5] introduced a third symbol (the zero symbol)[2] in the coding process that this step has received special attention. The zero symbol increases the number of subgroups of classes to be considered in the ternary ECOC framework by allowing some classes to be ignored. Because of this, strategies such as one-versus-one [8] and random sparse coding [5] can be formulated in the same ECOC framework. However, though different coding designs are possible, most of them are predefined without taking into account the knowledge of the problem domain. In this way, large codes are required to give rich information of the data (with its corresponding computational cost increment). However, one cannot guarantee that the learned problems are the most suitable ones for a given task. Concerned with this problem, Utschick and Weichselberger [20] have proposed the first problem-dependent ECOC design. In their work, they optimize a maximum-likelihood objective function by means of the expectation maximization algorithm in order to improve the process of binary coding. As mentioned by the authors: "*the results of some experiments make us believe that for many polychotomous classification problems, the one-versus-all method is still the optimal choice for the output coding*." Crammer and Singer [9] also have reported improvement in the design of the ECOC codes. In their work, they show that the exhaustive computation of the coding matrix is NP-hard with the number of classes. As an alternative, the authors propose a method for a heuristical search of the optimal coding matrix by relaxing the representation of the code matrix from discrete to continuous values. Recently, new approaches involving heuristics for the design of problem-dependent output codes have been proposed [10], [12], [27] with successful results.

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 [6]. In this sense, based on the rules used by the decoding strategies, they can be grouped into three types: those based on distance measurements between the output code and the target codeword, those which are based on estimating class membership probabilities, and those which are based on transforming the pattern space [28]. The first attempt for ECOC decoding is the Hamming Decoding

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.

2. In the literature, the binary ECOC has been applied indistinctively for $\{-1, +1\}$ (based on Machine Learning theory) or $\{0, 1\}$ (based on communication problems) symbols. For the ternary case, the third symbol is set to zero, and the $\{-1, +1\}$ symbols identify the class membership.

• *The authors are with the Departament de Matemàtica Aplicada i Anàlisi, Universitat de Barcelona and the Computer Vision Center, Universitat Autonoma de Barcelona, Gran via de les Corts 585, 08007 Barcelona, Spain. E-mail: {sergio, oriol, petia}@maia.ub.es.*

Manuscript received 10 Jan. 2008; revised 6 Aug. 2008; accepted 24 Oct. 2008; published online 30 Oct. 2008.
Recommended for acceptance by J. Matas.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2008-01-0019.
Digital Object Identifier no. 10.1109/TPAMI.2008.266.

(HD) [1]. The euclidean Decoding (ED) [8] is another of the most favored decoding strategies used in the literature. Still, very few alternative decoding strategies have been proposed. In [7], the use of Inverse Hamming Decoding (IHD) and the Centroid Decoding for binary ECOC designs are introduced. Other decoding strategies for nominal, discrete, and heterogeneous attributes have been proposed in [19]. With the introduction of the zero symbol, Allwein et al. [5] show the advantage of using a Loss-based function of the margin. There have been several attempts to introduce probabilities in the ECOC decoding process [17], [18]. In [17], the authors use conditional probabilities to estimate the class membership in a kernel machine approach. An alternative probabilistic design of the coding and decoding strategies is proposed in [18], where the probability of each class feature is adjusted; in particular, this decoding approach is bound to a specific base classifier.

Although the Loss-based decoding presented in [5] provides a first tentative way to decode a ternary ECOC matrix, there are no proper studies that show the behavior of each particular decoding strategy when the ternary coding is used. Note that the traditional decoding techniques are formulated to deal with just two symbols. Thus, the addition of the zero symbol requires the adjustment of the decoding step. In this paper, we analyze the ternary ECOC framework and show some inconsistencies produced at the decoding step. We define two properties to deal with a successful decoding which are analyzed for the state-of-the-art strategies over a new decoding taxonomy. Based on the presented properties, we introduce the Loss-Weighted Decoding strategy (*LW*) for discrete and continuous outputs of the classifiers, where the discrete output corresponds to the class label and the magnitude of the continuous output to the measure of confidence in the prediction. The method is based on a combination of probabilities that adjusts the weights of each position in the coding matrix M. When a continuous value is not available, we propose as an alternative the discrete Pessimistic Beta Density Distribution Decoding (β-DEN), based on estimating the probability density functions between codewords using a model of the accuracy and uncertainty of the measure. We evaluate the state-of-the-art coding and decoding strategies on different categorization scenarios: 16 UCI Machine Learning repository data sets classification and real traffic sign recognition in uncontrolled environments. The results show that when one takes into account the new measurements, significant performance improvement is obtained over any of the state-of-the-art coding and decoding designs.

The paper is organized as follows: Section 2 gives a brief introduction to the ECOC framework and overviews the state-of-the-art on coding and decoding designs for the binary and the ternary ECOC frameworks. Section 3 analyzes the ternary symbol-based ECOC definition, where a taxonomy that embeds all ternary decoding schemes is presented. The decoding strategies are analyzed in a general decomposition framework, and two novel decoding approaches are presented. Several experiments are performed in Section 5. Finally, Section 6 concludes the paper.

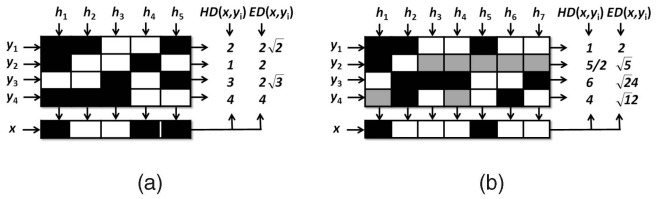


Fig. 1. (a) Binary ECOC design for a 4-class problem. An input test codeword x is classified to class c_2 using the Hamming or the euclidean Decoding. (b) Example of a ternary matrix M for a 4-class problem. A new test codeword x is classified to class c_1 using the Hamming and the euclidean Decoding.

2 ECOC Overview

Given a set of N classes to be learned in an ECOC design, n different bipartitions (groups of classes) are formed, and n binary problems (dichotomizers) over the partitions are trained. As a result, a codeword of length n is obtained for each class, where each position (bit) of the code corresponds to a response of a given dichotomizer (coded by $+1$ or -1 according to their 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. In Fig. 1a, we show an example of a binary coding matrix M. The matrix is coded using five dichotomizers $\{h_1, \ldots, h_5\}$ for a 4-class problem $\{c_1, \ldots, c_4\}$ with respective codewords $\{y_1, \ldots, y_4\}$. The hypotheses are trained by considering the labeled training data samples $\{(\rho_1, l(\rho_1)), \ldots, (\rho_m, l(\rho_m))\}$ for a set of m data samples. The white regions of the coding matrix M are coded by $+1$ (considered as a class for its respective dichotomizer h_j), and the dark regions are coded by -1 (considered as the other one). For example, the first classifier is trained to discriminate c_3 against c_1, c_2, and c_4; the second one classifies c_2 and c_3 against c_1 and c_4, etc., as follows:

$$h_1(x) = \begin{cases} +1, & \text{if } x \in \{c_3\} \\ -1, & \text{if } x \in \{c_1, c_2, c_4\} \end{cases}, \ldots, \quad h_5(x) = \begin{cases} +1, & \text{if } x \in \{c_2, c_4\} \\ -1, & \text{if } x \in \{c_1, c_3\}. \end{cases} \tag{1}$$

During the decoding process, applying the n binary classifiers, a code x is obtained for each data sample ρ in the test set. This code is compared to the base codewords (y_i, $i \in [1, .., N]$) of each class defined in the matrix M. And the data sample is assigned to the class with the *closest* codeword. In Fig. 1a, the new code x is compared to the class codewords $\{y_1, \ldots, y_4\}$ using the Hamming [1] and the euclidean Decoding [8]. Note that if the distance between two codewords is d in the binary case, $\frac{d}{2} - 1$ bit errors can be corrected—since, through $\frac{d}{2} - 1$ discrepancies between the test sample and its class occurrence, the test sample will still be assigned to its true class. In the examples, the test sample is classified to class c_2, correcting one bit error.

In the ternary symbol-based ECOC, the coding 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 given classifier. A ternary coding design is shown in Fig. 1b. The matrix is coded using seven dichotomizers $\{h_1, \ldots, h_7\}$ for a 4-class problem $\{c_1, \ldots, c_4\}$ with respective codewords

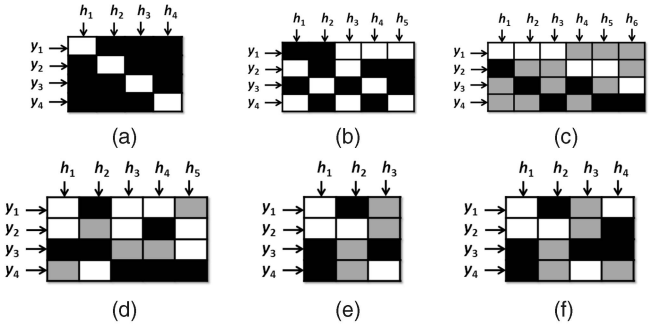


Fig. 2. Coding designs for a 4-class problem: (a) one-versus-all, (b) dense random, (c) one-versus-one, (d) sparse random, (e) DECOC, and (f) ECOC-ONE.

$\{y_1, \ldots, y_4\}$. The white regions are coded by 1 (considered as one class by the respective dichotomizer h_j), the dark regions by -1 (considered as the other class), and the gray regions correspond to the zero symbol (classes that are not considered by the respective dichotomizer h_j). For example, the first classifier is trained to discriminate c_3 against c_1 and c_2 without taking into account class c_4, the second one classifies c_2 against c_1, c_3, and c_4, etc. In this case, the Hamming and euclidean decoding classify the test data sample by class c_1. Note that a test codeword cannot contain the zero value since the output of each dichotomizer is $h_j \in \{-1, +1\}$.

2.1 Coding Designs

In this section, we review the state of the art on coding designs. We divide the designs based on their membership to the binary or the ternary ECOC frameworks.

2.1.1 Binary Coding

The standard binary coding designs are the one-versus-all [1] strategy and the dense random strategy [10]. In one-versus-all, each dichotomizer is trained to distinguish one class from the rest of classes. Given N classes, this technique has a codeword length of N bits. An example of a one-versus-all ECOC design for a 4-class problem is shown in Fig. 2a. 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 have suggested a length of $n = 10 \log N$ [5]. For the set of generated dense random matrices, the optimal one should maximize the Hamming Decoding measure between rows and columns (also considering their complementary), taking into account that each column of the matrix M must contain the two different symbols $\{-1, +1\}$. An example of a dense random ECOC design for a 4-class problem and five dichotomizers is shown in Fig. 2b. The complete coding approach requires the complete set of classifiers to be measured $(2^{N-1} - 1)$, which usually is computationally unfeasible in practice [5].

2.1.2 Ternary Coding

The standard ternary coding designs are the one-versus-one strategy [8] and the sparse random strategy [5]. The one-versus-one strategy considers all possible pairs of classes. Thus, its codeword length is $\frac{N(N-1)}{2}$ (see Fig. 2c). The sparse random strategy is similar to the dense random design, but it includes the third symbol zero with another probability to appear, given by $P(0) = 1 - P(-1) - P(1)$. Studies have suggested a sparse code length of $15 \log N$ [5] (see Fig. 2d).

Due to the huge number of bits involved in the traditional coding strategies, new problem-dependent designs have been proposed [10], [12], [27]. The new techniques are based on exploiting the problem domain by selecting the representative binary problems that increase the generalization performance while keeping the code length small. The Discriminant ECOC (DECOC) of [10] is based on the embedding of discriminant tree structures derived from the problem domain. As a result, the length of the codeword is only $(n - 1)$ (see Fig. 2e). Finally, the recently proposed ECOC-ONE strategy [12] is based on the extension of ECOC configurations. The method uses a coding process that trains relevant binary problems guided by a validation subset. A length of $2N$ bits for the codeword has been suggested. The design of Fig. 2f is obtained by extending the DECOC design of Fig. 2e with just one extra dichotomizer.

2.2 Decoding Designs

In this section, we review the state of the art on decoding designs.

2.2.1 Binary Decoding

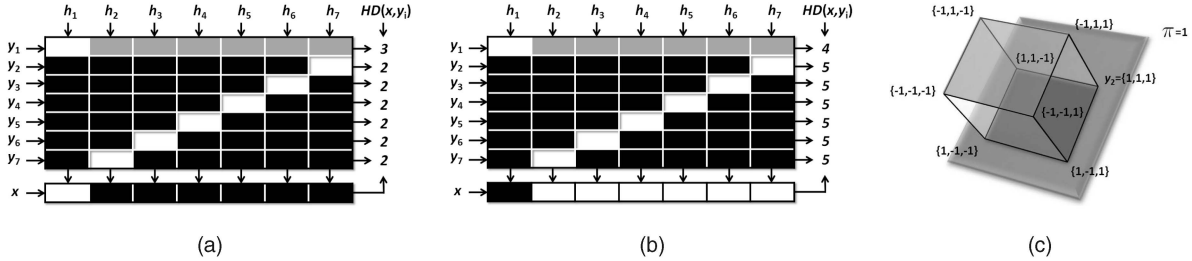
The most frequently applied binary decoding designs are: Hamming decoding [1], Inverse Hamming decoding [7], and euclidean decoding [10].

- *Hamming decoding:* The initial proposal to decode is the Hamming decoding measure. This measure is defined as $HD(x, y_i) = \sum_{j=1}^{n}(1 - \text{sign}(x^j \cdot y_i^j))/2$. This decoding strategy is based on the 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, and two possible symbols can be found at each position of the sequence [6].[3]
- *Inverse Hamming decoding:* The Inverse Hamming decoding [7] is defined as follows: Let Δ be the matrix composed of the Hamming decoding measures between the codewords of M. Each position of Δ is defined by $\Delta(i_1, i_2) = HD(y_{i_1}, y_{i_2})$. Δ can be inverted to find the vector containing the N individual class-likelihood functions by means of

$$IHD(x, y_i) = \max(\Delta^{-1} D^T), \tag{2}$$

where the values of $\Delta^{-1} D^T$ can be seen as the proportionality of each class codeword in the test codeword, and D is the vector of Hamming decoding values of the test codeword x for each of the base codewords y_i. The practical behavior of the IHD is shown to be very close to the behavior of the HD strategy [5].

3. Note that if $y \in \{-1, 0, +1\}$, the test codeword $x \in \{-1, +1\}$, and $sign(0) = 0$, the HD formulation is equivalent to the $L1$ norm $L1 = \sum_{j=1}^{n} |x^j - y^j|$, but including the factor $1/2$.



(a) (b) (c)

Fig. 3. (a) Ternary coding matrices for a 7-class problem codified using seven dichotomizers $\{h_1, .., h_7\}$. (b) A new test codeword x is classified using the Hamming decoding. (c) Cube of codewords of length 3.

- *Euclidean decoding:* Another well-known decoding strategy is the euclidean decoding. This measure is defined as

$$ED(x, y_i) = \sqrt{\sum_{j=1}^{n} (x^j - y_i^j)^2}.$$

2.2.2 Ternary Decoding

Concerning the ternary decoding, the state-of-the-art strategies are: Attenuated euclidean decoding [12], loss-based decoding [5], and the probabilistic-based decoding [17].

- *Attenuated Euclidean decoding:* This technique is an adaptation of the euclidean decoding. The formulation is redefined taking into account the factors $|y_i^j||x^j|$. It makes the measure to be unaffected by the positions of the codeword y_i that contain the zero symbol ($|y_i^j| = 0$). Note that, in most of the cases, $|x^j| = 1$.[4] Then, the euclidean decoding measure is redefined as follows:

$$AED(x, y_i) = \sqrt{\sum_{j=1}^{n} |y_i^j||x^j|(x^j - y_i^j)^2}. \quad (3)$$

- *Loss-based decoding:* The loss-based decoding strategy [5] chooses the label ℓ_i that is most consistent with the predictions f (where f is a real-valued function f: $\rho \to \mathcal{R}$), in the sense that, if the data sample ρ was labeled ℓ_i, the total loss on example (ρ, ℓ_i) would be minimized over choices of $\ell_i \in \ell$, where ℓ is the complete set of labels. Formally, given a Loss-function model, the decoding measure is the total loss on a proposed data sample (ρ, ℓ_i):

$$LB(\rho, y_i) = \sum_{j=1}^{n} L(y_i^j \cdot f^j(\rho)), \quad (4)$$

where $y_i^j \cdot f^j(\rho)$ corresponds to the *margin* and L is a loss function that depends on the nature of the binary classifier. The two most common loss functions are $L(\theta) = -\theta$ (linear loss-based decoding (LLB)) and $L(\theta) = \mathrm{e}^{-\theta}$ (exponential loss-based decoding (ELB)). The final decision is achieved by assigning a label to example ρ according to the class c_i that obtains the minimum score.

4. If x^j corresponds to a position of the test codeword, it only takes the values -1 or $+1$.

- *Probabilistic-based decoding:* Recently, the authors of [17] proposed a probabilistic-based decoding strategy based on the continuous output of the classifier to deal with the ternary decoding. The measure is given by

$$PD(y_i, x) = -log\left(\prod_{j \in [1,..,n]: M(i,j) \neq 0} P(x^j = M(i,j)|f^j) + K\right), \quad (5)$$

where K is a constant factor that collects the probability mass dispersed on the invalid codes, and the probability $P(x^j = M(i,j)|f^j)$ is estimated by means of

$$P(x^j = y_i^j|f^j) = \frac{1}{1 + \mathbf{e}^{y_i^j(\upsilon^j f^j + \omega^j)}},$$

where vectors υ and ω are obtained by solving an optimization problem [17].

3 A Framework for a General Representation of Decoding Strategies

In this section, first we analyze the ternary ECOC framework. We show examples where the use of the traditional decoding strategies is inconsistent to deal with a successful classification. Second, we give a general representation of decoding strategies and we group them based on the properties they fulfill. The properties are analyzed for the state-of-the-art decoding strategies on the new representation. Finally, two novel decoding strategies are proposed to deal with a successful decoding.

3.1 Ternary Decoding Analysis

In order to work with the large set of binary problems of the ternary ECOC framework, we need to know how to decode a ternary ECOC matrix $M \in \{-1, 0, +1\}$. Although standard decoding strategies are currently applied over 3-symbol matrices, it seems reasonable to analyze if the traditional decoding rules are correctly used in the ternary case. To show the behavior of the standard Hamming decoding strategy in the ternary ECOC framework, we designed the example of Fig. 3a. In this example, a ternary coding matrix for a 7-class problem $\{c_1, .., c_7\}$ is codified by means of seven dichotomizers $\{h_1, .., h_7\}$.

Now, let us observe the test codeword x of Fig. 3a obtained by applying the seven dichotomizers $\{h_1, .., h_7\}$ of the coding

matrix M to a new data sample ρ. The values of the test codeword correspond to $x = \{1, -1, -1, -1, -1, -1, -1\}$. As commented before, the test codeword cannot contain the zero symbol since each classifier should vote in either way. In the example, the Hamming decoding takes as input the test codeword x and each class codeword y_i, $i \in \{1, .., 7\}$. The decoding measure obtained for each class is shown on the right of the matrix. The output of the HD strategy assigns a higher decoding value to class c_1 in comparison to the other classes, and thus, any of the classes $\{c_2, .., c_7\}$ can be selected as a first choice.

In order to analyze this example, let us have a look at the subset of codewords represented in the *coding space* of Fig. 3c. 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, the codeword $y_1 = \{1, 0, 0\}$ represented by the *plane* $\pi : y_1^1 = 1$ in the figure 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^e is a possible representation[5] of the codeword y_1. Now observe the codeword $y_2 = \{1, 1, 1\}$ shown in the figure. Note that y_2 corresponds to one of the four representations of y_1 ($y_2 \in Y_1^e$). Then, in the figure, y_2 corresponds to a point in the *plane* π. Taking into account this decomposition, the test codeword x of Fig. 3a is a possible representation of codeword y_1 of class c_1. Thus, it seems reasonable to classify x as c_1. However, in the example of Fig. 3a, c_1 is the last choice. One can see this effect occurs because the decoding value increases with the number of positions that contain the zero symbol when we use the HD strategy. Let us introduce a term to denote this phenomenon.

Definition 1. ***Decoding bias*** *is the value introduced by the comparison of a position coded by $\{-1, +1\}$ with a position containing the zero symbol.*

Now observe the example of Fig. 3b. A new test codeword $x = \{-1, 1, 1, 1, 1, 1, 1\}$ is evaluated in the same ECOC design. In this case, the classification decision obtained by the HD is class c_1 with a minimum decoding value of four, while the decoding value of the rest of classes is five. Observe that the only trained classifier that takes into account c_1 is h_1. However, if we use the HD, we are deciding class c_1 according to the information obtained from the classifiers that have not considered class c_1 in their learning process. Therefore, all the information provided by class c_1 is contained in the first position of its codeword y_1.

In the example of Fig. 3b, either considering or not the zero positions to decode, when we use the HD, the decision in both cases is class c_1. This effect can be explained by the fact that the quantity of codeword positions codified by $\{-1, +1\}$ introduces a second *bias* that makes the measures between codewords noncomparable. It is produced because the decoding process for each codeword works in a different range of values. This effect leads to another definition.

Definition 2. ***A dynamic range bias*** *corresponds to the difference among the ranges of values associated to the decoding process of each codeword.*

Observe that this range depends on the number of positions codified by zero. In Fig. 3b, the codeword y_1 works on a different *dynamic range* than the rest of codewords $\{y_2, .., y_7\}$. In the example of Fig. 3b, the decoding process of codeword y_1 takes a minimum value of three when the first bit matches, and a maximum value of four when the first bit fails. It means that the *dynamic range* for the codeword y_1 applying the Hamming decoding is $[3, 4]$. On the other hand, codewords $\{y_2, .., y_7\}$ can take a minimum value of zero at the decoding process when all bits match, and a maximum value of seven when all bits fail, obtaining a *dynamic range* of $[0, 7]$. When we consider the first position of y_1 to decode, a failure in that position should have the same influence as the failure at all positions containing $\{-1, +1\}$ symbols on the rest of codewords (independently of the number of zero bits). In the same way, a match on that position also must represent the same information than to match all the positions containing $\{-1, +1\}$ symbols on the rest of codewords. Then, the codewords take values from the same *dynamic range*, and the results are comparable.

3.2 Decoding Decomposition

Based on the three possible symbols of the ternary ECOC framework, we define the following terms: Let b be the value produced when a bit of the test codeword with a $\{-1, +1\}$ value is compared to a zero symbol, a the value produced by a match in a position of a codeword containing a $\{-1, +1\}$ value, and e the value introduced by an error in a position of a codeword containing a $\{-1, +1\}$ value. Then, we introduce the following definition to represent decoding strategies:

Definition 3. ***A basic decoding decomposition*** *of a pair[6] of codewords is defined as follows:*

$$d = \sum_{k \in I_b} b_k + \sum_{i \in I_a} a_i + \sum_{j \in I_e} e_j \tag{6}$$

where I_b, I_a, and I_e are the sets of indexes of a codeword corresponding to the zero positions, matches on $\{-1, +1\}$ values, and failures on $\{-1, +1\}$ values, respectively. Let $|I_b| = z, |I_a| = \alpha$, and $|I_e| = \beta$ be the number of zeros, number of matches between two codewords, and number of failures between two codewords, respectively. In this sense, the length of a codeword is $n = z + \alpha + \beta$. Note that the value b corresponds to the bias induced by a zero position applying a particular decoding strategy.

As a zero symbol means that the corresponding classifier is not trained over a class, considering the *decision* of this classifier to estimate the similarity of the new test example to that class does not make sense. Thus, we define the first property as follows:

5. Possible representation means that any test example of class c_1 could give a codeword from Y_1^e.

6. When we speak about a pair of codewords, we understand that one is the test codeword and the other one corresponds to a class codeword.

TABLE 1
Types of Decoding Strategies

	$b \neq 0$	$b = 0$
Different *dynamic ranges*	Type 0	Type I
Same *dynamic ranges*	Type II	Type III

Property 1. *The bias induced by a zero position applying a particular decoding strategy is zero ($b = 0$).*

Moreover, we argue that to obtain comparable results between class codewords, each codeword of the coding matrix M should take values in the same *range*.

Definition 4. *The dynamic range (DR) associated to each codeword is determined as follows:*

$$DR = [min(K_1, K_2), max(K_1, K_2)], K_1 = \sum_{i \in I_a} |a_i|,$$
$$|I_a| = \alpha = n - z, K_2 = \sum_{j \in I_e} |e_j|, \quad |I_e| = \beta = n - z.$$

If K_1 and K_2 are constant factors for all pairs of codewords, the *dynamic range* is maintained for all classes, and the decoding measures are comparable, then we define the second property as follows:

Property 2. *K_1 and K_2 are constant factors for all pairs of codewords.*

Based on the previous properties, we define the four types of decoding strategies shown in Table 1.

3.3 Analysis of State-of-the-Art Decoding Strategies

Following the notation in (6), we split the state-of-the-art decoding strategies according to the decomposition of (6) and analyze the two previous properties in each case. The analysis is performed over the decoding strategies reviewed in the previous section: Hamming decoding, Inverse Hamming decoding, euclidean decoding, attenuated euclidean decoding, loss-based decoding, and the probabilistic-based decoding approach of [17].

- *Hamming decoding:* We can easily find a correspondence between the original formulation of the HD and the representation of (6). HD always induces a value of $\frac{1}{2}$ for b_k, $k \in I_b$. A match does not influence the measure ($a_i = 0$, $i \in I_a$), and a failure on a position increases the measure in $e_j = 1$, $j \in I_e$. Then, the new representation can be defined as follows:

$$HD(x, y) = \sum_{k \in I_b} b_k + \sum_{j \in I_e} e_j = \frac{z}{2} + \beta. \tag{7}$$

  Analyzing the Hamming decoding in the ternary case, one can observe that the zero positions introduce a *bias* of $\frac{z}{2}$. Moreover, the prediction is influenced by the value of z, which makes codewords to take values from different *dynamic ranges* for different number of zero positions. In this sense, HD corresponds to the strategies of Type 0.
- *Inverse Hamming decoding:* Looking at (2), the term Δ^{-1} of the IHD corresponds to a constant factor only dependent on the trained class codes. Therefore, we can focus on the term D^T to find a correspondence with (6). The term $\Delta_1^{-1} D^T$ stands for the IHD for the first codeword of the coding matrix M. Note that Δ_1^{-1} does not depend on the test codeword x. Then, if the components of the first row of Δ^{-1} correspond to $\{W_1, .., W_N\}$, the result of the product $\Delta_1^{-1} D^T$ can be defined as

$$IHD(x, y_1) = \Delta_1^{-1} D^T = \sum_{j=1}^{N} W_j \cdot HD(x, y1)$$
$$= \sum_{j=1}^{N} W_j \Big(\frac{z_j}{2} + \beta_j \Big),$$

  which implies

$$IHD(x, y_1) = z_1 \frac{1}{2} \left(W_1 + \sum_{i=2}^{N} \frac{W_i z_i}{z_1} \right) + \beta_1 \left(W_1 + \sum_{i=2}^{N} \frac{W_i \beta_i}{\beta_1} \right). \tag{8}$$

  This expression exactly corresponds to the representation of (6) for a codeword y_1, $b_k := b^{IHD}$, where $b^{IHD} = -1(W_1 + \sum_{i=2}^{N} \frac{W_i z_i}{z_1})$, $k \in I_b$, $e_j = -1(W_1 + \sum_{i=2}^{N} \frac{W_i \beta_i}{\beta_1})$, $j \in I_e$, and $a_i = 0$, $i \in I_a$, being the weights W dependent on the design of the coding matrix M. Note that different *biases* are induced by different number of zeros and different *dynamic ranges* are also obtained for different values of z and β. Thus, the IHD corresponds to the Type 0 strategies.
- *Euclidean decoding:* The parameters in this case are: $b_k = 1$, $k \in I_b$, $a_i = 0$, $i \in I_a$, and $e_j = 4$, $j \in I_e$, obtaining the following representation:

$$ED(x, y) = \sum_{k \in I_b} b_k + \sum_{j \in I_e} e_j = \sum_{k \in I_b} 1 + \sum_{j \in I_e} 4 = z + 4\beta. \tag{9}$$

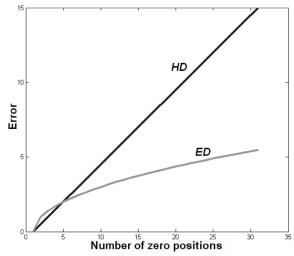


Fig. 4. Errors induced by the zero symbol for the HD and ED decoding strategies.

Compared to the error induced by the zero symbol in the HD strategy, one can observe that, in this case, b_k, $k \in I_b$ is less significative in comparison with e_j, $j \in I_e$. In Fig. 4, one can see this behavior for different number of zero positions. When the number of zeros increases, the error accumulated by the ED is less significative than

the HD error. This is one of the main reasons why the ED usually improves the performance of the HD when applied to ternary symbol-based ECOC [10]. This strategy also is of Type 0.

- *Attenuated euclidean decoding:* This technique is an adaptation of the euclidean decoding that takes into account Property 1 [12]. In this case, the difference between the ED and the AED is the value of b_k, $k \in I_b$, fixed to zero by AED. The new representation is $AED(x,y) = \sum_{j \in I_e} e_j = \sum_{j \in I_e} 4 = 4\beta$. Note that the weighting parameter of AED (3) avoids the *bias* produced by the zero symbol. Nevertheless, different *dynamic ranges* are obtained for different values of β. Thus, this strategy corresponds to Type I.
- *Loss-based decoding:* We introduce the loss-based decoding in the representation of (6) for the linear and the exponential loss-based functions. Using a loss function, the final measure is obtained by means of an additive model where the matches introduce negative weights. In particular, the continuous LLB_C parameters considering the margin of the output of the classifier are as follows: $b_k = 0$, $k \in I_b$, $a_i = -|f^i(\rho)|$, $i \in I_a$, and $e_j = |f^j(\rho)|$, $j \in I_e$, where $|f^j(\rho)|$ stands for the absolute value of the output of $f^j(\rho)$, giving

$$\begin{aligned} LLB_C(\rho, y) &= \sum_{i \in I_a} a_i + \sum_{j \in I_e} e_j \\ &= -\sum_{i \in I_a} |f^i(\rho)| + \sum_{j \in I_e} |f^j(\rho)|, \end{aligned} \tag{10}$$

and the following parameters in the discrete case LLB_D: $b_k = 0$, $k \in I_b$, $a_i = -1$, $i \in I_a$, and $e_j = 1$, $j \in I_e$, giving

$$\begin{aligned} LLB_D(x, y) &= \sum_{i \in I_a} a_i + \sum_{j \in I_e} e_j \\ &= \sum_{i \in I_a} (-1) + \sum_{j \in I_e} 1 = -\alpha + \beta. \end{aligned} \tag{11}$$

In the case of the exponential loss-based function, in the continuous case ELB_C, considering the margin of the output of the classifier, we have: $b_k = 1$, $k \in I_b$, $a_i = 1/\mathbf{e}^{|f^i(\rho)|}$, $i \in I_a$, and $e_j = \mathbf{e}^{|f^j(\rho)|}$, $j \in I_e$, obtaining

$$\begin{aligned} ELB_C(\rho, y) &= \sum_{k \in I_b} b_k + \sum_{i \in I_a} a_i + \sum_{j \in I_e} e_j \\ &= \sum_{k \in I_b} 1 + \sum_{i \in I_a} 1/\mathbf{e}^{|f^i(\rho)|} + \sum_{j \in I_e} \mathbf{e}^{|f^j(\rho)|} \end{aligned} \tag{12}$$

and the following parameters in the discrete case ELB_D: $b_k = 1$, $k \in I_b$, $a_i = 1/\mathbf{e}$, $i \in I_a$, and $e_j = \mathbf{e}$, $j \in I_e$, obtaining

$$\begin{aligned} ELB_D(x, y) &= \sum_{k \in I_b} b_k + \sum_{i \in I_a} a_i + \sum_{j \in I_e} e_j \\ &= \sum_{k \in I_b} 1 + \sum_{i \in I_a} 1/\mathbf{e} + \sum_{j \in I_e} \mathbf{e} = z + \frac{\alpha}{\mathbf{e}} + \beta\mathbf{e}. \end{aligned} \tag{13}$$

In LLB cases, the matches and the failures have the same influence for the same magnitude of the margin of the output of the classifiers, while, in the ELW cases, a failure is $\mathbf{e}^{2|f^j(\rho)|}$ times more significant than a match for a same magnitude of the margin of the output of the classifiers. One can see that $b_k = 0, k \in I_b$, in the LLB cases, but in both the LLB and ELB cases, different *dynamic ranges* are obtained for different values of α and β for LLB, and z, α, and β for ELB. Thus, ELB is of Type 0 and LLB of Type I.

- *Probabilistic-based decoding:* From the initial definition of this strategy (5), we can fix the parameters $K = \omega = 0$ and $\upsilon = -1$ in order to simplify the study of the technique. Applying minus to change the decision rule from likelihood to the measure of (6), we obtain the following representation:

$$PD(\rho, y) = \log\left(\prod_{i \in I_a}\left(\frac{1}{1 + 1/\mathbf{e}^{|f^i(\rho)|}}\right)\prod_{j \in I_e}\left(\frac{1}{1 + \mathbf{e}^{|f^j(\rho)|}}\right)\right). \tag{14}$$

We can easily change this representation into the form of (6) by defining: $b_k = 0$, $k \in I_b$, $a_i = \log(\frac{1}{1+1/\mathbf{e}^{|f^i(\rho)|}})$, $i \in I_a$, and $e_j = \log(\frac{1}{1+\mathbf{e}^{|f^j(\rho)|}})$, $j \in I_e$ in the continuous case PD_C, which implies

$$\begin{aligned} PD_C(\rho, y) &= \sum_{i \in I_a} a_i + \sum_{j \in I_e} e_j \\ &= \sum_{i \in I_a} \log\left(\frac{1}{1 + 1/\mathbf{e}^{|f^i(\rho)|}}\right) \\ &\quad + \sum_{j \in I_e} \log\left(\frac{1}{1 + \mathbf{e}^{|f^j(\rho)|}}\right) \end{aligned}$$

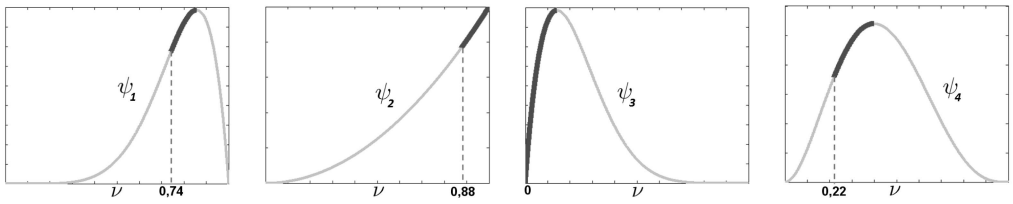
and the following parameters for the discrete case:

$$\begin{aligned} PD_D\colon b_k = 0, k \in I_b, a_i = \log\left(\frac{1}{1 + 1/\mathbf{e}}\right), \\ i \in I_a, e_j = \log\left(\frac{1}{1 + \mathbf{e}}\right), j \in I_e, \end{aligned}$$

which implies

$$\begin{aligned} PD_D(x, y) &= \sum_{i \in I_a} a_i + \sum_{j \in I_e} e_j \\ &= \sum_{i \in I_a} \log\left(\frac{1}{1 + 1/\mathbf{e}}\right) + \sum_{j \in I_e} \log\left(\frac{1}{1 + \mathbf{e}}\right) \\ &= \alpha \log\left(\frac{1}{1 + 1/\mathbf{e}}\right) + \beta \log\left(\frac{1}{1 + \mathbf{e}}\right). \end{aligned}$$

This strategy was proposed to deal with the ternary decoding. In particular, it satisfies that $b_k = 0$, $k \in I_b$, since the induced *bias* by the zero symbol is null (note that in (5), the positions in the coding matrix M containing a zero symbol are not considered at the decoding step). However, note that there is no upper bound for the decoding value obtained ($[0, \infty)$), and that this value is influenced by the number of positions coded by -1 and $+1$ values, obtaining different *dynamic ranges* for different values of α and β. Thus, the Probabilistic-based decoding belongs to Type I strategies.



(a) (b) (c) (d)

Fig. 5. Pessimistic Score decoding for the test codeword x and the matrix M for the four classes of Fig. 1b. (a) Class c_1, (b) class c_2, (c) class c_3, and (d) class c_4. The probability for the second class allows a successful classification in this case.

3.4 β-Density and Loss-Weighted Decoding

Based on the presented grouping of strategies, none of the decoding techniques in the literature belongs to either Type II or Type III. In this section, we introduce two novel decoding strategies of Type III. First, we propose a methodology based on the discrete output of the classifiers, called Pessimistic β-Density Distribution decoding. After that, we extend its behavior using a continuous extension.

3.4.1 Pessimistic β-Density Distribution Decoding ($\beta - DEN$)

The simplest way to avoid the *bias* of the zero symbol is to ignore the positions coded by zero. This yields a measure that counts the number of coincidences between the input codeword and the class codeword. In order to make all the codewords to work in the same *dynamic range*, the measure is normalized by the total number of positions coded by $\{-1, +1\}$, obtaining $d(x, y_i) = \frac{\alpha_i}{\alpha_i + \beta_i}$. The main drawback of this definition is that it is not robust when there is a small number of coded positions in the codeword. In order to alleviate this problem, we introduce a prior bias, known as the Laplace correction. With this correction, the new decoding score, called Laplacian decoding (LAP), is defined as follows:

$$LAP(x, y_i) = \frac{\alpha_i + 1}{\alpha_i + \beta_i + K}, \tag{15}$$

where K is an integer value that codifies the number of classes considered by the classifier—two in this case.

Based on this formulation, we can define a suboptimal method, called Pessimistic β-Density Distribution decoding. The method is based on estimating the probability density functions between two codewords. The main goal of this strategy is to model at the same time the accuracy and uncertainty based on a pessimistic score in order to obtain more reliable predictions. We use an extension of the continuous binomial distribution, the β-distribution, defined as $\psi_i(\nu, \alpha_i, \beta_i) = \frac{1}{K}\nu^{\alpha_i}(1-\nu)^{\beta_i}$, where ψ_i is the β-Density Distribution between a codeword x and a class codeword y_i for class c_i and $\nu \in [0, 1]$. Note that the maximum of the function ψ_i corresponds to the previous discrete approximation of (15) without the prior K

$$\frac{\partial \psi_i}{\partial \nu} = \frac{1}{K}\left(\nu^{\alpha_i - 1}(1-\nu)^{\beta_i - 1)}\right)(\alpha_i(1-\nu) - \beta_i\nu),$$
$$\alpha_i(1-\nu) - \beta_i\nu = 0, \quad \nu = \frac{\alpha_i}{\alpha_i + \beta_i}.$$

Now, we can make use of the integral around the maximum $\nu = \frac{\alpha_i}{\alpha_i + \beta_i}$ of ψ_i to obtain a measure of the confidence in the classification prediction. In this sense, given a test codeword x and the set of functions $\psi(\nu, \alpha, \beta) = [\psi_1(\nu, \alpha_1, \beta_1), \ldots, \psi_N(\nu, \alpha_N, \beta_N)]$, the class c_i is assigned to x if it achieves the highest score s_i, defined as the pessimistic score satisfying the following equivalence:

$$s_i \ : \quad \int_{\nu_i - s_i}^{\nu_i} \psi_i(\nu, \alpha_i, \beta_i) d\nu = u, \tag{16}$$

where u is a threshold parameter. After a preliminary set of experiments, we fixed $u = \frac{1}{3}$. Note that u governs the uncertainty influence in the final score. Fig. 5 shows the estimated density functions $[\psi_1, \psi_2, \psi_3, \psi_4]$ for the design shown in Fig. 1b. Observe that, in the design of Fig. 1b, the HD and the ED decoding strategies classify the test codeword x by class c_1, although, according to the present discussion, the decision should be class c_2. In Fig. 5, one can see that the β-DEN decoding classifies the test data sample to its correct class c_2, obtained by Fig. 5b. It can be shown that, when a function ψ_i is estimated by a combination of values α_i and β_i, the sharpness is higher if it is generated by a majority of one of both types. Moreover, this sharpness depends on the number of code positions different to zero and the balance between the number of matches and failures. In this way, the pessimistic score reflects the confidence in the expectation of the probability density function.

Now, let us analyze the β-Density to obtain the representation in the form of (6). We can apply a negative logarithmic function to the β-Density formulation to split it. We obtain the parameters: $b_k = 0$, $k \in I_b$, $a_i = -\log(\nu)$, $i \in I_a$, and $e_j = -\log(1-\nu)$, $j \in I_e$, and the following representation:

$$\beta - DEN(x, y) = \sum_{i \in I_a} a_i + \sum_{j \in I_e} e_j = \sum_{i \in I_a}(-\log(\nu))$$
$$+ \sum_{j \in I_e}(-\log(1-\nu)) = -\alpha\log(\nu) - \beta\log(1-\nu).$$

Note that, in the β-DEN decoding, the zero symbol has no influence and the *dynamic range* for all the codewords takes values in the same interval $[0, 1]$, being a strategy of Type III.

3.4.2 Loss-Weighted Decoding (LW)

We define a novel Loss-Weighted decoding based on a combination of normalized probabilities to adapt the ternary ECOC decoding to the strategies of Type III. The properties are encoded in a matrix that is used to weight the decoding process. The weight matrix codifies Properties I and II, being independent of the coding and decoding strategy applied.

$$M=\begin{bmatrix}1&1&-1&0\\1&-1&0&0\\1&1&1&-1\end{bmatrix} \quad H=\begin{bmatrix}0.955&0.955&1.000&0.000\\0.900&0.800&0.000&0.000\\1.000&0.905&0.805&0.805\end{bmatrix} \quad M_W=\begin{bmatrix}0.328&0.328&0.344&0.000\\0.529&0.471&0.000&0.000\\0.285&0.257&0.229&0.229\end{bmatrix}$$

(a) (b) (c)

Fig. 6. (a) Coding matrix M of four hypotheses for a 3-class problem. (b) Performance matrix H. (c) Weight matrix M_W.

Moreover, as not all of the hypotheses have the same performance on learning the data samples, the accuracy of each binary problem is used to adjust the final decision.

We define the weight matrix M_W by assigning to each position of the codeword codified by $\{-1,+1\}$ a weight value of $\frac{1}{n-z}$. As $\alpha+\beta=n-z$, by excluding the zero positions, the previous process codifies the same *dynamic range* for all codewords, fulfilling Property 2. Moreover, the *bias* of the third symbol is avoided by assigning a weight of zero to those positions of the weight matrix M_W that contain a zero in the coding matrix M. In this way, $\sum_{j=1}^{n} M_W(i,j)=1,\ \forall i=1,\ldots,N$, fulfilling Property 1 and satisfying Type III properties.

We assign to each position (i,j) of a performance matrix H a continuous value that corresponds to the performance of the dichotomizer h_j classifying the samples of class c_i as follows:

$$\begin{aligned} H(i,j) &= \frac{1}{m_i}\sum_{k=1}^{m_i}\varphi(h^j(\rho_k^i),i,j),\\ \varphi(x^j,i,j) &= \begin{cases}1, & \text{if} \quad x^j=y_i^j,\\ 0, & \text{otherwise.}\end{cases}\end{aligned} \tag{17}$$

Note that (17) makes H to have zero probability at those positions corresponding to unconsidered classes.

We normalize each row of the matrix H so that M_W can be considered as a discrete probability density function

$$M_W(i,j)=\frac{H(i,j)}{\sum_{j=1}^{n}H(i,j)}, \quad \forall i\in[1,\ldots,N], \quad \forall j\in[1,\ldots,n].$$

In Fig. 6, a weight matrix M_W for a 3-multiclass problem of four hypotheses is estimated. Fig. 6a shows the coding matrix M. The matrix H of Fig. 6b represents the accuracy of the hypotheses classifying the instances of the training set. The normalization of H results in a weight matrix M_W shown in Fig. 6c.

Once we obtain the weight matrix M_W, we introduce the weight matrix in the Loss-based decoding. The decoding estimation is obtained by means of an LB decoding model $L(\theta)$, where θ corresponds to $y_i^j\cdot f(\rho,j)$ (similar to the Loss-based decoding), weighted using M_W,[7]

$$LW(\rho,i)=\sum_{j=1}^{n}M_W(i,j)L\big(y_i^j\cdot f(\rho,j)\big). \tag{18}$$

The summarized algorithm is shown in Table 2.

Note that the weight matrix M_W encoding the ternary decoding properties is independent of the coding and decoding strategies applied. In this sense, it can be potentially applied to any existing decoding strategy. For the present formulation, we choose the loss-based decoding as the base decoding strategy to apply the weight matrix M_W since LB was one of the firsts attempts toward ternary decoding. In this sense, different weighted decodings can be formulated.[8] Moreover, note that depending on the problem we are working on, not only the continuous output of the base classifier could be useful to weight the matrix M_W, but also prior information about the classes distribution (or class frequencies instead) as well as other useful information can also be included.

TABLE 2
Loss-Weighted Algorithm

Loss-Weighted strategy: Given a coding matrix M,
1) Calculate the performance matrix H:

$$H(i,j)=\frac{1}{m_i}\sum_{k=1}^{m_i}\varphi(h^j(\rho_k^i),i,j) \quad \text{based on:} \tag{19}$$

$$\varphi(x^j,i,j)=\begin{cases}1, & \text{if} \quad x^j=y_i^j,\\ 0, & \text{otherwise.}\end{cases} \tag{20}$$

2) Normalize H: $\sum_{j=1}^{n}M_W(i,j)=1, \quad \forall i=1,...,N$:

$$M_W(i,j)=\frac{H(i,j)}{\sum_{j=1}^{n}H(i,j)}, \ \forall i\in[1,...,N], \ \forall j\in[1,...,n] \tag{21}$$

3) Given a test data sample ρ, decode based on:

$$LW(\rho,i)=\sum_{j=1}^{n}M_W(i,j)L(y_i^j\cdot f(\rho,j)) \tag{22}$$

In order to obtain the formulation of LW in the representation of (6), we consider the use of the linear and the exponential loss functions with discrete and continuous possible outputs of the classifiers.

In the case of the linear loss-weighted, using the continuous output of the classifier LLW_C, we obtain the values $b_k=0, k\in I_b, a_i=-M_W(\_,i)|f^i(\rho)|, i\in I_a, M_W(\_,i)\in[0,1]$, where "_" stands for the row whose corresponding codeword is being compared and $e_j=M_W(\_,j)|f^j(\rho)|$, $j\in I_e$, $M_W(\_,j)\in[0,1]$. Then, the new representation is as follows:

$$\begin{aligned} LLW_C(\rho,y) &= \sum_{i\in I_a}a_i+\sum_{j\in I_e}e_j\\ &= -\sum_{i\in I_a}M_W(\_,i)|f^i(\rho)|+\sum_{j\in I_e}M_W(\_,j)|f^j(\rho)| \end{aligned} \tag{23}$$

and the following parameters considering a discrete output of the classifier LLW_D: $b_k=0$, $k\in I_b$, $a_i=-M_W(\_,i)$, $i\in I_a$, $M_W(\_,i)\in[0,1]$, and $e_j=M_W(\_,j)$, $j\in I_e$, $M_W(\_,j)\in[0,1]$, giving

$$LLW_D(x,y)=\sum_{i\in I_a}a_i+\sum_{j\in I_e}e_j=-\sum_{i\in I_a}M_W(\_,i)+\sum_{j\in I_e}M_W(\_,j). \tag{24}$$

7. Note that different Loss functions as well as discrete and continuous outputs of the classifiers can also be applied.

8. We have performed some experiments applying the weight matrix over other decoding strategies, such as the Weight-euclidean decoding, obtaining significant performance improvements.

TABLE 3
Decoding Parameters in the Decomposition of (6)

Strategy	b	a	e
HD	1/2	0	1
IHD	b^{IHD}	0	$-1\left(W_1+\sum_{i=2}^{N}\frac{W_i\beta_i}{\beta_1}\right)$
ED	1	0	4
AED	0	0	4
LLB_C	0	$-\lvert f(\rho)\rvert$	$\lvert f(\rho)\rvert$
LLB_D	0	-1	1
ELB_C	1	$1/\mathbf{e}^{\lvert f(\rho)\rvert}$	$\mathbf{e}^{\lvert f(\rho)\rvert}$
ELB_D	1	$1/\mathbf{e}$	$\mathbf{e}$
PD_C	0	$\log\left(\frac{1}{1+\mathbf{e}^{\lvert f(\rho)\rvert}}\right)$	$\log\left(\frac{1}{1+1/\mathbf{e}^{\lvert f(\rho)\rvert}}\right)$
PD_D	0	$\log\left(\frac{1}{1+\mathbf{e}}\right)$	$\log\left(\frac{1}{1+1/\mathbf{e}}\right)$
$\beta-DEN$	0	$\log(\nu)$	$\log(1-\nu)$
LLW_C	0	$-M_W(\_,i)\lvert f(\rho)\rvert$	$M_W(\_,j)\lvert f(\rho)\rvert$
LLW_D	0	$-M_W(\_,i)$	$M_W(\_,j)$
ELW_C	0	$\frac{M_W(\_,i)}{\mathbf{e}^{\lvert f(\rho)\rvert}}$	$M_W(\_,j)\mathbf{e}^{\lvert f(\rho)\rvert}$
ELW_D	0	$\frac{M_W(\_,i)}{\mathbf{e}}$	$M_W(\_,j)\mathbf{e}$

If we take as baseline the discrete representation of (24) and consider the ideal case where each dichotomizer learns the training data with zero error, we obtain: $b_k=0$, $k\in I_b$, $a_i=-\frac{1}{n-z}$, $i\in I_a$, and $e_j=\frac{1}{n-z}$, $j\in I_e$, which implies

$$\begin{aligned}LLW_D(x,y)&=\sum_{i\in I_a}a_i+\sum_{j\in I_e}e_j=-\sum_{i\in I_a}\frac{1}{n-z}+\sum_{j\in I_e}\frac{1}{n-z}\\&=-\frac{\alpha}{n-z}+\frac{\beta}{n-z}.\end{aligned}\tag{25}$$

Then, we can observe that in the discrete LLW_D, the zero symbol is not considered. Moreover, independently of the number of positions coded by $\{-1,+1\}$, if all of these positions match, then $\alpha=n-z$, and the parameter $-\frac{\alpha}{n-z}$ of (25) is maintained constant to $K_1=-1$ for all of the codewords. In the case that all positions coded by $\{-1,+1\}$ correspond to failures, $\beta=n-z$, obtaining a constant value $K_2=1$. Therefore, all of the codewords take values in the interval $[-1,1]$. The same occurs with the continuous LLW_C since the previous behavior is only affected by the factor introduced by the margin of the output of the classifier.

Applying the same formalism in the case of the continuous Exponential Loss-Weighted ELW_C, we obtain the following parameters: $b_k^{(0)}=0$, $k\in I_b$, $a_i=\frac{M_W(\_,i)}{\mathbf{e}^{|f^i(\rho)|}}$, $i\in I_a$, $M_W(\_,i)\in[0,1]$, and $e_j=M_W(\_,j)\mathbf{e}^{|f^j(\rho)|}$, $j\in I_e$, $M_W(\_,j)\in[0,1]$, obtaining

$$\begin{aligned}ELW_C(\rho,y)&=\sum_{i\in I_a}a_i+\sum_{j\in I_e}e_j\\&=\sum_{i\in I_a}\frac{M_W(\_,i)}{\mathbf{e}^{|f^i(\rho)|}}+\sum_{j\in I_e}M_W(\_,j)\mathbf{e}^{|f^j(\rho)|}\end{aligned}\tag{26}$$

and the following parameters considering a discrete output of the classifier ELW_D: $b_k^{(0)}=0$, $k\in I_b$, $a_i=\frac{M_W(\_,i)}{\mathbf{e}}$, $i\in I_a$, $M_W(\_,i)\in[0,1]$, and $e_j=M_W(\_,j)\mathbf{e}$, $j\in I_e$, $M_W(\_,j)\in[0,1]$, obtaining

TABLE 4
Decoding Strategies Grouped by Type and Discrete/Continuous Domains

Type	Discrete	Continuous
Type 0	*HD, IHD, ED*	-
Type I	*AED*	*LB, PD*
Type III	$\beta-DEN$, *LAP, LW*	*LW*

$$\begin{aligned}ELW_D(x,y)&=\sum_{i\in I_a}a_i+\sum_{j\in I_e}e_j\\&=\sum_{i\in I_a}\frac{M_W(\_,i)}{\mathbf{e}}+\sum_{j\in I_e}M_W(\_,j)\mathbf{e}.\end{aligned}\tag{27}$$

If we take as baseline the discrete representation of (27), and consider the ideal case where each dichotomizer learns the training data with zero error, we obtain: $b_k=0$, $k\in I_b$, $a_i=\frac{1}{(n-z)\mathbf{e}}$, $i\in I_a$, and $e_j=\frac{\mathbf{e}}{n-z}$, $j\in I_e$, which implies

$$\begin{aligned}ELW_D(x,y)&=\sum_{i\in I_a}a_i+\sum_{j\in I_e}e_j=\sum_{i\in I_a}\frac{1}{(n-z)\mathbf{e}}+\sum_{j\in I_e}\frac{\mathbf{e}}{n-z}\\&=\frac{\alpha}{(n-z)\mathbf{e}}+\frac{\beta\mathbf{e}}{n-z}.\end{aligned}\tag{28}$$

In the ELW cases, the zero symbol is not considered. If all positions coded by $\{-1,+1\}$ correspond to matches, $\alpha=n-z$, which makes all of the codewords to obtain the constant value $K_1=\frac{1}{\mathbf{e}}$. On the other hand, if all positions coded by $\{-1,+1\}$ correspond to failures, then $\beta=n-z$ implies a constant factor $K_2=\mathbf{e}$, which makes all codewords to obtain values in the same *dynamic range* $[\frac{1}{\mathbf{e}},\mathbf{e}]$. Thus, all the LW variants correspond to Type III strategies.

3.5 Taxonomy of Decoding Strategies

Table 3 summarizes the values of the parameters obtained using the representation of (6) for all the decoding strategies. The decoding strategies of Table 3 are sorted from Type 0 to Type III designs. At the first column of the table, the values of b different from zero point out the methods that introduce a *bias* for the zero symbol. Note that four of the traditional approaches do not avoid this *bias*. The columns of values a and e stand for the values introduced by a match and a failure at the decoding step, respectively. Note that none of the traditional decoding strategies presented in the literature belongs to Type II and Type III strategies since the *dynamic ranges* differ for different number of positions coded by zero. Only the $\beta-DEN$ and LW decoding variants normalize the *dynamic ranges* to work in the same range of values for all codewords. Note that, if we substitute in (6), the parameters b, a, and e of each decoding strategy, we obtain an equivalent decoding evaluation than using its original formulation.

Based on the previous types of decoding strategies and with the use of discrete or continuous outputs of the classifiers, six different groups of decodings are shown in Table 4. The Laplacian decoding LAP has also been included as the simplest choice of Type III strategy. Some strategies, such as ED, AED, LB, and PD can also be used in both discrete and continuous domains, though there are no evidences of their use in the literature. Note that none of

the decoding strategies presented in the literature belongs to Type II strategies since there is no method that maintains the *dynamic range* for all codewords at the same time that includes *bias* for the zero symbol.

In the next section, we perform several experiments to test the proposed methodology. Based on the present formulation, our working hypothesis is that when the decoding strategies avoid the *bias* produced by the zero symbol and all the codewords work in the same *dynamic range*, fulfilling Properties 1 and 2, the performance of the ECOC designs is improved. Therefore, we apply the decoding strategies on the state-of-the-art coding designs and test their behavior over different categorization problems.

4 Results

Before the results are presented, we discuss our validation methodology regarding the data, comparatives, measurements, and experiments.

- *Data*: The data used for the experiments consists of 16 multiclass data sets from the UCI Machine Learning Repository database [29] and video sequences from a Mobile Mapping system [22].
- *Comparatives*: For the comparatives, we used the decoding strategies analyzed in the previous section: Hamming decoding, euclidean decoding, inverse Hamming decoding, attenuated euclidean decoding, loss-based decoding with linear and exponential loss functions, probabilistic-based decoding, Laplacian decoding as the simplest choice of Type III strategy, pessimistic β-density distribution decoding, and four variants of the loss-weighted decoding strategy: the linear loss-weighted, with discrete and continuous outputs of the classifier, and the exponential loss-weighted with discrete and continuous outputs of the classifier. For all of the experiments, the state-of-the-art decoding strategies are applied using the default and optimized parameters given by the authors.

  Furthermore, all of the decoding strategies are applied on the state-of-the-art ECOC coding designs: one-versus-one [8], one-versus-all [1], dense random [5], sparse random [5], DECOC [10], and ECOC-ONE [12] designs. The parameters of the coding strategies are the predefined or the default values given by the authors. The dense and sparse matrices are selected from a set of 20,000 generated random matrices whose corresponding codewords have a length of N, where N corresponds to the number of classes, in order to provide a fair comparison with the one-versus-all, DECOC, and ECOC-ONE strategies using a similar number of dichotomizers.
- *Measurements*: To measure the performance of the different strategies, we apply a stratified sampling and tenfold cross-validation, and test for confidence interval at 95 percent with a two-tailed t test. We also use the Nemenyi test to look for significant statistical differences between the methods' performances at 95 percent [30]. The base classifiers used for the experiments are Gentle Adaboost with 40 runs of decision stumps [4], the Linear OSU[9] implementation of Support Vector Machines (SVM), and a tuned Support Vector Machine with Radial Basis Function kernel [2], [3].
- *Experiments*: The experiments are divided as follows: First, we evaluate the classification on 16 UCI data sets, and second, we address a real 9-class speed traffic sign categorization problem. We compare the performance of the different decoding strategies based on each coding design. Finally, we also compare this behavior using a tuned base classifier.

TABLE 5
UCI Repository Data Sets Characteristics

Problem	#T	#A	#C	Problem	#T	#A	#C
Dermathology	366	34	6	Yeast	1484	8	10
Iris	150	4	3	Satimage	6435	36	7
Ecoli	336	8	8	Letter	20000	16	26
Wine	178	13	3	Pendigits	10992	16	10
Glass	214	9	7	Segmentation	2310	19	7
Thyroid	215	5	3	OptDigits	5620	64	10
Vowel	990	10	11	Shuttle	14500	9	7
Balance	625	4	3	Vehicle	846	18	4

#T: number of training samples, #A: number of attributes, and #C: number of classes.

4.1 UCI Classification

The first experiment consists of classifying 16 multiclass UCI repository databases. The details of the data sets are shown in Table 5. In this experiment, the 13 decoding strategies are applied on the six coding designs, which corresponds to a total of 2,496 tenfold experiments. Table 6 summarizes the performance results obtained over the UCI data sets for the Gentle Adaboost base classifier. For each data set and decoding strategy, the best performance and its corresponding coding design are shown. The best performance per data set is also highlighted in boldface. Note that the best results are obtained by the Type III strategies in most cases, and specially by the ELW_C decoding technique applied over the one-versus-one and ECOC-ONE coding designs.[10]

In order to summarize these results, we estimated the ranking of each particular decoding strategy for the two different base classifiers. Thus, each decoding strategy has been applied on six codings $\times 16$ data sets. Using these 96 experiments for each decoding, the rankings, considering the intersection of confidence intervals on one hand and without considering the confidences on the other hand, are shown in Fig. 7a for Gentle Adaboost and Linear SVM, respectively.[11] The rankings are obtained estimating each particular ranking r_i^j for each problem i and each decoding j, and computing the mean ranking R for each

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

10. Performance details of all the experiments of this paper can be found at http://www.maia.ub.es/~sergio/.

11. When we consider the intersection of confidence intervals, two methods obtain the same rank position if their corresponding ranges, defined by their mean performance and confidence interval, intersect in a common subinterval of values.

TABLE 6
UCI Performance Results

	HD	IHD	ED	AED	LLB	ELB	PD	LAP	DEN	LLW_D	LLW_C	ELW_D	ELW_C
Derma	92.04 E-ONE	92.37 DENSE	92.04 E-ONE	92.04 E-ONE	**95.13** 1VSALL	**95.13** 1VSALL	95.11 1VSALL	92.04 E-ONE	92.04 E-ONE	92.04 E-ONE	**95.13** 1VSALL	91.81 E-ONE	95.11 1VSALL
Iris	94.00 1VS1	94.00 1VS1	94.00 E-ONE	95.33 E-ONE	95.33 DENSE	95.33 DENSE	95.33 DENSE	95.33 E-ONE	95.33 E-ONE	95.33 E-ONE	**96.00** E-ONE	**96.00** E-ONE	**96.00** E-ONE
Ecoli	78.01 DECOC	78.12 DECOC	78.23 DECOC	79.74 E-ONE	81.38 1VS1	80.53 1VS1	**81.47** 1VS1	79.46 E-ONE	79.74 E-ONE	81.38 E-ONE	79.74 1VS1	80.98 E-ONE	80.98 E-ONE
Wine	94.35 E-ONE	94.35 E-ONE	95.49 E-ONE	**96.05** E-ONE	**96.05** 1VSALL	95.49 SPARSE	95.49 1VS1	**96.05** E-ONE	**96.05** E-ONE	**96.05** E-ONE	**96.05** E-ONE	**96.05** E-ONE	**96.05** E-ONE
Glass	66.69 1VS1	67.14 1VS1	66.69 1VS1	66.69 1VS1	64.01 E-ONE	64.59 E-ONE	66.21 1VS1	66.69 1VS1	66.69 1VS1	67.16 1VS1	66.69 E-ONE	**67.62** 1VS1	66.69 E-ONE
Thyroid	**92.10** E-ONE	**92.10** E-ONE	**92.10** E-ONE	**92.10** E-ONE	**92.10** DENSE	**92.10** DENSE	**92.10** 1VS1	**92.10** E-ONE	**92.10** E-ONE	**92.10** E-ONE	**92.10** SPARSE	**92.10** E-ONE	**92.10** DENSE
Vowel	61.30 E-ONE	60.82 E-ONE	62.83 E-ONE	64.36 E-ONE	55.63 E-ONE	56.48 E-ONE	58.48 1VS1	65.36 E-ONE	65.36 E-ONE	66.79 E-ONE	69.53 E-ONE	68.45 E-ONE	**71.77** E-ONE
Balance	78.97 E-ONE	78.97 E-ONE	80.15 SPARSE	80.15 SPARSE	75.93 1VS1	77.86 1VS1	**82.22** E-ONE	78.97 SPARSE	78.96 SPARSE	80.94 E-ONE	76.09 E-ONE	80.94 DENSE	77.55 E-ONE
Yeast	49.64 E-ONE	49.94 E-ONE	50.88 E-ONE	51.98 E-ONE	50.74 1VS1	51.81 1VS1	49.85 E-ONE	52.04 E-ONE	52.04 E-ONE	52.04 E-ONE	51.88 E-ONE	52.04 E-ONE	**52.17** E-ONE
Satimage	83.25 1VS1	83.19 1VS1	83.25 1VS1	84.06 E-ONE	83.51 1VS1	84.24 E-ONE	83.31 1VS1	84.15 E-ONE	84.15 E-ONE	84.88 E-ONE	85.25 E-ONE	85.03 E-ONE	**85.37** E-ONE
Letter	88.31 E-ONE	87.68 1VS1	88.96 1VS1	89.03 E-ONE	88.54 1VS1	88.76 1VS1	87.65 1VS1	90.12 E-ONE	90.32 E-ONE	91.09 E-ONE	90.86 E-ONE	91.12 E-ONE	**91.92** E-ONE
Pendigits	96.72 1VS1	97.02 1VS1	97.34 1VS1	97.34 1VS1	96.78 1VS1	96.87 1VS1	96.98 1VS1	97.34 1VS1	97.34 1VS1	97.88 1VS1	97.98 1VS1	97.96 1VS1	**98.01** 1VS1
Segment	96.10 1VS1	96.23 1VS1	96.13 E-ONE	96.44 E-ONE	95.54 1VSALL	95.80 1VS1	96.06 1VS1	96.44 E-ONE	96.44 E-ONE	96.48 E-ONE	96.76 E-ONE	96.65 E-ONE	**97.01** E-ONE
Optdigits	96.50 1VS1	96.85 1VS1	96.58 E-ONE	96.50 1VS1	94.03 E-ONE	94.83 E-ONE	96.26 1VS1	96.60 E-ONE	96.60 E-ONE	96.90 E-ONE	96.83 E-ONE	**99.82** E-ONE	97.05 E-ONE
Vehicle	72.34 E-ONE	72.34 E-ONE	72.34 E-ONE	72.34 E-ONE	72.34 1VS1	72.80 E-ONE	72.70 E-ONE	72.34 E-ONE	72.34 E-ONE	72.70 E-ONE	72.70 E-ONE	72.70 E-ONE	**73.15** E-ONE
Shuttle	99.84 1VS1	99.83 1VSALL	99.84 E-ONE	99.84 E-ONE	**99.88** 1VSALL	**99.88** E-ONE	99.15 E-ONE	99.84 SPARSE	99.84 SPARSE	**99.88** SPARSE	99.85 E-ONE	**99.88** SPARSE	99.86 E-ONE

decoding as $R_j = \frac{1}{J}\sum_i r_i^j$, where J is the total number of problems (six codings $\times 16$ databases). Note that, either for the Gentle Adaboost base classifiers or the Linear SVM classifier, the ranking positions of each decoding strategy is the same in most cases. When the confidence interval is considered, the ranking differences are less significant. This is produced by the fact that the performance of some strategies falls into the range of performances defined by the confidence interval of other methods. However, still in those cases, the relative positions among techniques are maintained too. The general behavior of this graphics shows that Type III strategies, and, in particular, the four variants of the Loss-Weighted decoding, attain the best performance in the experiments, followed by Type I strategies, and finally by Type 0 strategies. Note that, in the variants of LW, for both Gentle Adaboost and Linear SVM, the differences between LLW with discrete and continuous outputs of the classifier are not significant, but in the case of ELW, the performance is improved by considering the continuous outputs of both base classifiers.

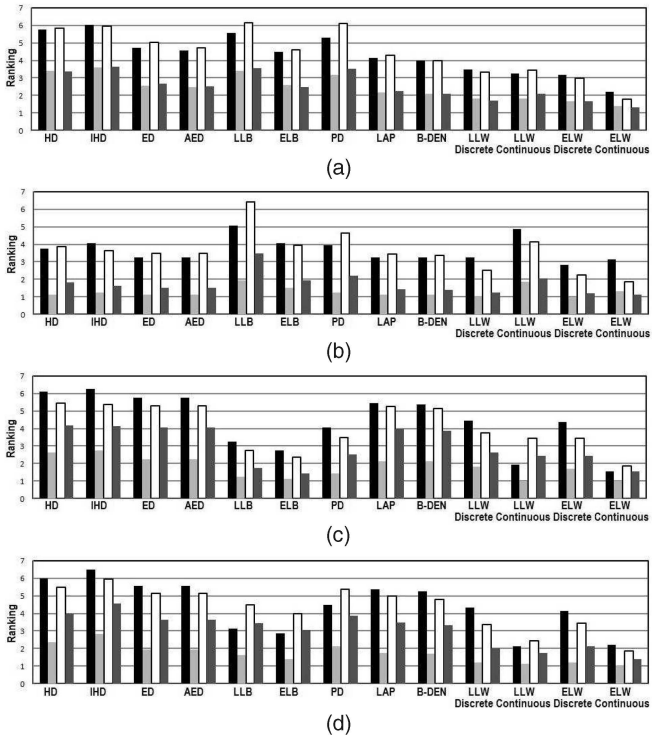


Fig. 7. Ranking for the decoding strategies over coding designs and UCI data sets: Gentle Adaboost without (in black) and considering (in light gray) the intersection of the confidence intervals, and Linear SVM without (in white) and considering (in dark gray) the intersection of the confidence intervals, respectively. (a) All coding design ranks, (b) one-versus-one design ranks, (c) one-versus-all design ranks, and (d) dense random design ranks.

Now, we analyze if the results of the different strategies are statistically significant. To check for the statistically significant methods, we use the Nemenyi test—two techniques are significantly different if the corresponding average rankings differ by at least the critical difference value (CD)

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6J}}, \tag{29}$$

where q_α is based on the Studentized range statistic divided by $\sqrt{2}$, k is the number of methods in the comparative, and J is the total number of performed experiments. In our case, when comparing 13 methods with a confidence value $\alpha = 0.05$, $q_{0.05} = 1.771$. Substituting in (29), we obtain a critical difference value of 0.995. Looking at the rankings of each decoding strategy shown in the first and third column of each group in Fig. 7a, one can observe that any variant of the Loss-Weighted strategy has a difference superior to the critical value of Type 0 and Type I strategies, only intersecting with the Laplacian and $\beta - DEN$ Type III strategies. Thus,

TABLE 7
Ranking Positions of the Decoding Strategies on the UCI Experiments Grouped by Type

	Gentle Adaboost			Linear SVM		
	Type 0	Type I	Type III	Type 0	Type I	Type III
Discrete	5.5000	4.9844	3.3715	5.6042	5.3880	3.2951
Continuous	3.0799	2.7839	**1.7813**	3.2778	3.0469	**1.8681**

we can argue that the LW variants are significantly better than any Type 0 and Type I strategies at 95 percent of the confidence interval in the present experiments. Remember that all of the decoding strategies in the literature belong to these two types. In the case of the Type 0 and Type I strategies, although Type I strategies tend to have inferior ranking (thus, better position) than the Type 0 methods, there are combinations of methods for which the difference is inferior to the critical value, and therefore, we cannot argue that, in those cases, the Type I strategies are significantly better than the methods of Type 0.

Finally, the mean ranking positions grouping the techniques in their respective types are shown in Table 7. One can observe that the ranking performance in all cases is better when satisfying the decoding properties, as claimed in the previous section. Besides, the novel Type III strategies obtain results statistically significantly better than the rest of the state-of-the-art strategies.

Now, we analyze the previous behavior of decoding strategies for each particular coding design. Figs. 7b, 7c, 7d, 8a, 8b, and 8c show the ranking of the decoding strategies using the same settings as the previous analysis for the one-versus-one, one-versus-all, dense random, sparse random, DECOC, and ECOC-ONE designs, respectively. In the case of the sparse random, DECOC, and ECOC-ONE designs of Fig. 8, the advantage of Type III strategies is clearer. Note

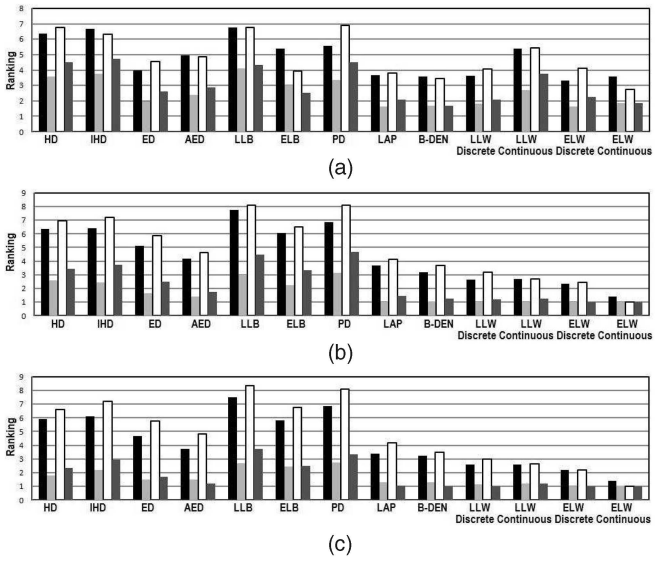


Fig. 8. Ranking for the decoding strategies over coding designs and UCI data sets: Gentle Adaboost without (in black) and considering (in light gray) the intersection of the confidence intervals, and Linear SVM without (in white) and considering (in dark gray) the intersection of the confidence intervals, respectively. (a) Sparse random design ranks, (b) DECOC design ranks, and (c) ECOC-ONE design ranks.

Fig. 9. (a) Samples from the road video sequences. (b) Speed data set samples.

that these three coding designs usually have high sparseness degree (high percentage of zero symbols in the coding matrix M). The difference among the decoding performances is produced because, when we increase the percentage of zero symbols, both biases produced by the third symbol also increase and the performance for the traditional decoding strategies is more affected.

On the other hand, the rankings of the one-versus-one, one-versus-all, and dense random designs of Fig. 7 do not significantly differ. This is due to the null sparseness degree for these three types of designs. Thus, the two biases produced by the zero symbol do not appear. A particular case occurs in the one-versus-one design, where though the sparseness degree seems high, the amount of positions containing the zero symbol and the $\{-1, +1\}$ values coincide for all codewords, and thus, the biases produced by all the codewords are the same. However, though the ranking positions do not significantly differ, one can observe that the new ternary decoding strategies are also selected as the first choice in the binary coding designs.

4.2 Real Multiclass Traffic Sign Categorization

For this experiment, we use the video sequences obtained from the Mobile Mapping System [22] to test the decoding strategies on a real traffic sign categorization problem. In this system, the position and orientation of the different traffic signs are 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. Fig. 9 shows examples of video sequences and samples of the speed database used in the experiments. The database contains a total of 2,500 samples divided in nine classes. Each sample is composed of 1,200 pixel-based features after smoothing the image and applying histogram equalization. For this experiment, we apply tenfold cross validation on the set of all the coding and decoding designs.

The rankings obtained from the experiments are shown in Fig. 10. Note that the different variants of loss-weighted strategy obtain the best positions in this real experiment. In particular, the exponential loss-weighted decoding using the continuous output of the base classifiers attains the best positions either when we use Gentle Adaboost or the Linear SVM. The rest of Type III strategies obtain good performance too. This behavior is more significant if we observe the rankings without considering the confidence interval of Fig. 10, corresponding to the second and fourth column of each group.

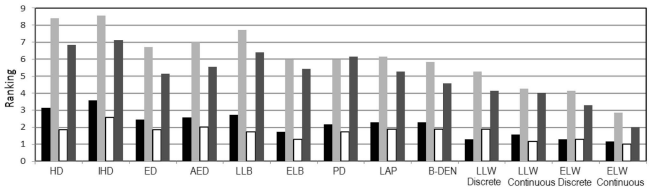


Fig. 10. Ranking of the decoding strategies for the different coding designs applied on the speed database: Gentle Adaboost considering (in black) and without (in light gray) considering the intersection of the confidence intervals, and Linear SVM considering (in white) and without considering (in dark gray) the intersection of the confidence intervals, respectively.

4.3 Discussions

Finally, it is important to discuss the behavior of the decoding strategies when a more complex base classifier is applied. In the previous experiments, the parameters for the Linear SVM classifier were fixed by default to compare the performance of the decoding strategies. In those cases, the performance results obtained by the one-versus-one and one-versus-all strategies were different. The individual problems considered by the one-versus-one strategy used to be significantly smaller than the one-versus-all strategy, and thus, they are easier to be learned using a simple base classifier. However, complex classifiers and optimization can make the results obtained by the two former strategies comparable [11]. In this sense, we include a brief experiment considering an SVM with Radial Basis Function kernel optimized via cross validation applied over the one-versus-one, one-versus-all, and sparse random strategies using the set of decoding strategies and UCI data sets to look if the behavior of the decoding strategies is maintained applying a more complex base classifier. For this experiment, the σ and regularization parameters were tested from 0.1 increasing per 0.05 up to 1 and from 1 increasing per 5 up to 150, respectively. The UCI data sets used correspond to the eight data sets described in the first column of Table 5: Dermathology, Iris, Ecoli, Wine, Glass, Thyroid, Vowel, and Balance. The rankings obtained by the decoding strategies on each of the previous ECOC designs for the optimized *RBF SVM* experiments are shown in Fig. 11. The results show that the behavior of the decoding strategies is maintained with respect to the previous results in Fig. 7. One can see that, for the one-versus-one and one-versus-all ECOC designs, the rankings among the decoding strategies are very similar in most cases. It is produced due to the null sparseness degree of both strategies. However, the Type III strategies are also selected as the first choice in most cases. In the case of the sparse random coding with tuned base classifier, the advantage of Type III methods is clearer.

5 Conclusion

In this paper, we analyzed the decoding step of the ternary symbol-based ECOC framework. We showed that the zero symbol produces serious inconsistences when using the traditional decoding strategies. We formulated a new taxonomy in order to represent decoding strategies. As a consequence, two novel strategies fulfilling the presented properties were proposed. The validation of the methodology

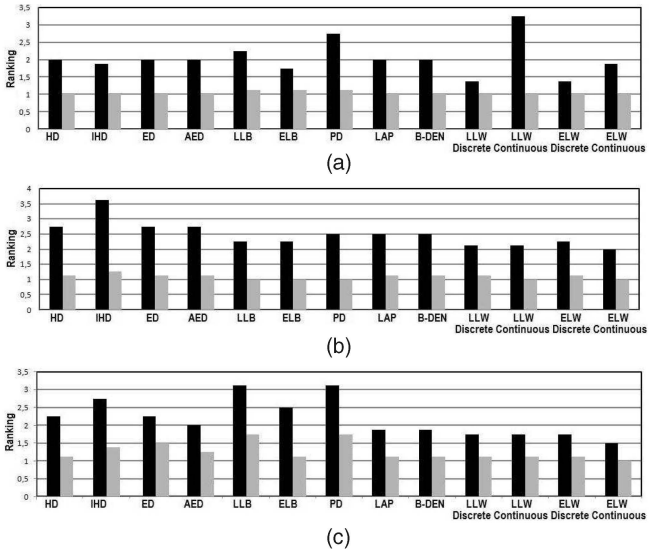


Fig. 11. Ranking for the decoding strategies over one-versus-one, one-versus-all, and sparse random coding designs and UCI data sets: without (in black) and considering (in gray) the intersection of the confidence intervals, respectively. (a) One-versus-one design ranks for $RBF\ SVM$, (b) one-versus-all design ranks for $RBF\ SVM$, (c) sparse random design ranks for $RBF\ SVM$.

was performed over a wide set of the UCI Machine Learning Repository data sets using the state-of-the-art coding and decoding strategies as well as Adaboost and Support Vector Machines as the base classifiers. We also validated the proposed strategies on a real traffic sign categorization problem in uncontrolled environments. We showed that, when the decoding strategies avoid the *bias* produced by the zero symbol and all the codewords work in the same *dynamic range*, significant performance improvement is obtained on the ECOC designs.

Acknowledgments

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

References

[1] N.J. Nilsson, *Learning Machines.* McGraw-Hill, 1965.
[2] V. Vapnik, *The Nature of Statistical Learning Theory.* Springer, 1995.
[3] OSU-SVM-TOOLBOX, http://svm.sourceforge.net, 2009.
[4] J. Friedman, T. Hastie, and R. Tibshirani, "Additive Logistic Regression: A Statistical View of Boosting," *The Annals of Statistics,* vol. 38, pp. 337-374, 1998.
[5] E. Allwein, R. Schapire, and Y. Singer, "Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers," *J. Machine Learning Research,* vol. 1, pp. 113-141, 2002.
[6] T. Dietterich and G. Bakiri, "Solving Multiclass Learning Problems via Error-Correcting Output Codes," *J. Artificial Intelligence Research,* vol. 2, pp. 263-286, 1995.
[7] T. Windeatt and R. Ghaderi, "Coding and Decoding for Multi-Class Learning Problems," *Information Fusion,* vol. 4, pp. 11-21, 2003.
[8] T. Hastie and R. Tibshirani, "Classification by Pairwise Grouping," *Proc. Conf. Neural Information Processing Systems,* vol. 26, pp. 451-471, 1998.
[9] K. Crammer and Y. Singer, "On the Learnability and Design of Output Codes for Multi-Class Problems," *Machine Learning,* vol. 47, pp. 201-233, 2002.

[10] O. Pujol, P. Radeva, and J. Vitrià, "Discriminant ECOC: A Heuristic Method for Application Dependent Design of Error Correcting Output Codes," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 6, pp. 1001-1007, June 2006.
[11] R. Rifkin and A. Klautau, "In Defense of One-vs-All Classification," *J. Machine Learning Research,* vol. 5, pp. 101-141, 2004.
[12] O. Pujol, S. Escalera, and P. Radeva, "An Incremental Node Embedding Technique for Error Correcting Output Codes," *Pattern Recognition,* to appear.
[13] T. Windeatt and G. Ardeshir, "Boosted ECOC Ensembles for Face Recognition," *Proc. Int'l Conf. Visual Information Eng.,* pp. 165-168, 2003.
[14] J. Kittler, R. Ghaderi, T. Windeatt, and J. Matas, "Face Verification Using Error Correcting Output Codes," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* vol. 1, pp. 755-760, 2001.
[15] R. Ghani, "Combining Labeled and Unlabeled Data for Text Classification with a Large Number of Categories," *Proc. Int'l Conf. Data Mining,* pp. 597-598, 2001.
[16] J. Zhou and C. Suen, "Unconstrained Numeral Pair Recognition Using Enhanced Error Correcting Output Coding: A Holistic Approach," *Proc. Int'l Conf. Document Analysis and Recognition,* vol. 1, pp. 484-488, 2005.
[17] A. Passerini, M. Pontil, and P. Frasconi, "New Results on Error Correcting Output Codes of Kernel Machines," *IEEE Trans. Neural Networks,* vol. 15, no. 1, pp. 45-54, Jan. 2004.
[18] O. Dekel and Y. Singer, "Multiclass Learning by Probabilistic Embeddings," *Proc. Conf. Neural Information Processing Systems,* vol. 15, 2002.
[19] N. Ishii, E. Tsuchiya, Y. Bao, and N. Yamaguchi, "Combining Classification Improvements by Ensemble Processing," *Proc. ACIS Int. Conf. Software Eng. Research, Management, and Applications,* pp. 240-246, 2005.
[20] W. Utschick and W. Weichselberger, "Stochastic Organization of Output Codes in Multiclass Learning Problems," *Neural Computation,* vol. 13, no. 5, pp. 1065-1102, 2004.
[21] T. Dietterich and E. Kong, "Error-Correcting Output Codes Corrects Bias and Variance," *Proc. 21th Int'l Conf. Machine Learning,* S. Prieditis and S. Russell, eds., pp. 313-321, 1995.
[22] J. Casacuberta, J. Miranda, M. Pla, S. Sanchez, A. Serra, and J. Talaya, "On the Accuracy and Performance of the Geomobil System," *Proc. Congress of Int'l Soc. for Photogrammetry and Remote Sensing,* 2004.
[23] R. Schapire and Y. Singer, "Improved Boosting Algorithms Using Confidence-Rated Prediction," *Machine Learning,* vol. 37, no. 3, pp. 297-336, 1999.
[24] J. Zhu, S. Rosset, H. Zou, and T. Hastie, "Multi-Class Adaboost. A Multiclass Generalization of the Adaboost Algorithm, Based on a Generalization of the Exponential Loss," 2005.
[25] T. Kikuchi and S. Abe, "Error Correcting Output Codes vs. Fuzzy Support Vector Machines," *Proc. Conf. Artificial Neural Networks in Pattern Recognition,* 2003.
[26] F. Ricci and D. Aha, "Error-Correcting Output Codes for Local Learners," *Proc. European Conf. Machine Learning,* pp. 280-291, 1998.
[27] S. Escalera, O. Pujol, and P. Radeva, "Boosted Landmarks of Contextual Descriptors and Forest-ECOC: A Novel Framework to Detect and Classify Objects in Clutter Scenes," *Pattern Recognition Letters,* vol. 28, no. 13, pp. 1759-1768, 2007.
[28] R.S. Smith and T. Windeatt, "Decoding Rules for Error Correcting Output Code Ensembles," *Multiple Classifier Systems,* pp. 53-63, Springer, 2005.
[29] A. Asuncion and D.J. Newman, UCI Machine Learning Repository, Dept. of Information and Computer Science, Univ. of California, Irvine, http://www.ics.uci.edu/mlearn/MLRepository.html, 2007.
[30] J. Demsar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *J. Machine Learning Research,* vol. 7, pp. 1-30, 2006.

Sergio Escalera received the BS and MS degrees from the Universitat Autònoma de Barcelona in 2003 and 2005, respectively, and the PhD degree in computer science in 2008 under the guidance of Dr. Petia Radeva and Dr. Oriol Pujol. He is a collaborator professor at the Universitat de Barcelona. His research interests include statistical and visual pattern recognition.

Oriol Pujol received the PhD degree in computer science from the Universitat Autònoma de Barcelona in 2004. Currently, he is an associate professor in the Applied Mathematics and Analysis Department at the Universitat de Barcelona. His main research interests include statistical machine learning techniques for object recognition and medical imaging analysis.

Petia Radeva received the PhD degree from the Universitat Autònoma de Barcelona on development of physics-based models applied to image analysis. Currently, she is an associate professor in the Applied Mathematics and Analysis Department at the Universitat de Barcelona. Her present research interests include the development of physics-based and statistical approaches for object recognition, medical image analysis, and industrial vision.