Pattern Recognition 45 (2012) 863–875

Contents lists available at ScienceDirect

Pattern Recognition

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

Design of reject rules for ECOC classification systems

P. Simeone, C. Marrocco, F. Tortorella *

DAEIMI, Università degli Studi di Cassino, Via G. Di Biasio 43, 03043 Cassino (FR), Italy

ARTICLE INFO

Article history:
Received 26 July 2010
Received in revised form
25 July 2011
Accepted 1 August 2011
Available online 10 August 2011

Keywords:
ECOC
Reject option
Multiple classifiers systems

ABSTRACT

ECOC is a widely used and successful technique, which implements a multi-class classification system by decomposing the original problem into several two-class problems. In this paper, we study the possibility to provide ECOC systems with a tailored reject option carried out through different schemes that can be grouped under two different categories: an external and an internal approach. The first one is based on the reliability of the entire system output and does not require any change in its structure. The second scheme, instead, estimates the reliability of the internal dichotomizers and implies a slight modification in the decoding stage. Experimental results on popular benchmark data sets are reported to show the behavior of the different schemes.

1. Introduction

The most immediate way to face a classification task with n multiple classes is to build a single monolithic classifier capable of producing multiple outputs. A viable alternative is to break the problem into a set of dichotomies; nowadays, this is a widely used solution because of the stronger theoretical roots and better comprehension in characterizing two-class classifiers, such as perceptrons or support vector machines. The simplest decomposition schemes designed for such an aim are *one vs. all*, which defines n binary problems each discriminating one class from all the remaining classes, and *one vs. one* [15], which splits the n class problem into a set of $n(n-1)/2$ binary problems each discriminating one class from another class. In the last years, another more general method has emerged as a well-established technique for many applications in the field of Pattern Recognition and Data Mining: *error correcting output coding* (ECOC) [9]. The rationale is to create a certain number L of different binary problems that aggregate the original n classes into two classes in different ways according to an $n \times L$ "coding matrix" $\mathbf{C}$. This matrix associates a binary string of length L to each class of the problem. The binary problems are defined by the columns in $\mathbf{C}$. When a new sample is classified by the L dichotomizers, a new binary string is obtained, which has to be matched with the existing n binary class codes, using a suitable decoding technique. Although the original motivation for this method was founded on the error correcting capabilities of the codes used to group the classes, it has also been proved that ECOC provides a reliable probability estimation and concurrent reduction of both bias and variance [18], thus motivating its good generalization capabilities. For such reasons, it has been successfully applied to a wide range of real applications, such as text and digit classification [14,35], face recognition, and verification [32,17] or fault detection [25].

Over the last years, many studies had focused their attention on several aspects of this subject. The design of suitable coding and decoding approach was investigated in many ways [1,33]. In particular, Masulli and Valentini [20] analyzed the factors influencing the effectiveness of ECOC classifiers, and studied the dependencies between the coding matrix structure and the correlation of the classifiers trained on its columns. Some of the most recent works were also deeply focused on the codes itself. Crammer and Singer [7] implemented a technique to design codes given the set of the employed binary classifiers. In [23], an approach for designing effective codes from data has been proposed, while in [11], the code dependence from subclass problems has been analyzed. Recently, some new techniques [27] have been proposed, where the iterative codes have been used to improve the overall classification performance of the ECOC systems, while in [10], a ternary code has been used to model the analyzed problem.

Similar to other classifier architectures, when used in real applications, ECOC-based systems are subject to produce classification errors that could have serious consequences, usually expressed by means of an error cost. In some cases, such a cost can be so high that it is convenient to reject the sample (i.e., to suspend the decision and call for a further test) instead of risking a wrong decision. Obviously, this choice also involves a non-negligible cost given by the charge of employing a more powerful system or requiring the decision of a human expert. Thus, a rule is needed to find the optimal trade-off between errors and rejects

* Corresponding author. Tel.: +39 0776 2993605; fax: +39 0776 2994333.
E-mail address: tortorella@unicas.it (F. Tortorella).

0031-3203/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2011.08.001

Table 1
Example of a coding matrix for a 5-classes problem.

Classes	Codewords														
ω_1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1
ω_2	−1	−1	−1	−1	−1	−1	−1	−1	+1	+1	+1	+1	+1	+1	+1
ω_3	−1	−1	−1	−1	+1	+1	+1	+1	−1	−1	−1	−1	+1	+1	+1
ω_4	−1	−1	+1	+1	−1	−1	+1	+1	−1	−1	+1	+1	−1	−1	+1
ω_5	−1	+1	−1	+1	−1	+1	−1	+1	−1	+1	−1	+1	−1	+1	−1

for the application at hand. The use of a reject option was first implemented by Chow [5], who proposed an optimal classification rule with a reject option. In this case, the optimal trade-off between errors and rejects can be accomplished if the *a posteriori* probabilities for each class are known.

Different approaches have also been proposed to incorporate the reject option directly into the classification scheme. In [6,26], some criteria for evaluating the classification reliability of neural classifiers have been proposed, while in [28,3], a reject option has been built for support vector machines. The benefits of a reject rule have also been studied in the case of linearly combined classifiers [13]. Unlike other classification techniques, ECOC lacks of a methodology for the application of a reject mechanism. In the literature, there is only one paper by Zhou et al. [34] that discusses the possibility of a rejection scheme for an ECOC system but this is just defined to increase the reliability on decision of the testing samples and it is not presented as a systematic approach. The behavior of ECOC systems with a reject option has also been studied in our previous papers [19,24] where a preliminary analysis of such a problem has been presented. In this paper, we have sensibly extended our previous investigation and introduced some new strategies to improve ECOC with such feature. ECOC classification systems consist of different levels of decision: an internal level where a decision for the dichotomizers output is taken and an external level where all these collected outputs are submitted to the decoding stage. Considering such a structure, in this paper, we have analyzed two different possibilities to apply a reject option. The first and more immediate one is to modify the final decoding stage, where the multi-class decision is accomplished and where it is possible to evaluate a reject threshold on the system output. The second solution is to work where the base classifiers take their binary decisions and to apply a reject rule to each of them. Different decision techniques have been proposed depending on the various reject options applicable to the dichotomizers. The effectiveness of the schemes has been verified on several benchmark data.

The paper is organized as follows: in Section 2, we analyze the ECOC and the possible decoding rules that can be adopted. Successively, Sections 3 and 4 tackle the problem of a reject rule in the ECOC systems and describe the different proposed approaches. The experimental results are reported in Section 5, while in Section 6, some conclusions are drawn and possible future developments are presented.

2. The ECOC approach

ECOC is commonly used to address the multi-class problem by means of decomposition in binary problems. A bit string of length L (*codeword*) is associated with each class ω_i with $i=1,\ldots,n$ to have every class represented by a different codeword. If n is the number of the original classes, the set of codewords is arranged in an $n \times L$ coding matrix $\mathbf{C}=\{c_{hk}\}$, where $c_{hk} \in \{-1,+1\}$. The columns of $\mathbf{C}$ define L binary problems, each requiring a specific dichotomizer. An example of coding matrix with $n=5$ and $L=15$ is shown in Table 1.

The classification task is performed by feeding each sample $\mathbf{x}$ to all the dichotomizers and collecting their outputs in a vector (*output vector*) that is then compared with the codewords of the coding matrix with a proper decoding procedure. Such procedures can be divided into hard decoding and soft decoding techniques, depending on the way in which the outcomes of dichotomizers are managed.

2.1. Hard decoding

In this case, crisp decisions are made on the outputs of the dichotomizers; the goal of hard decoding is to correct binary errors induced in the hard decision process. To this aim, let us consider $\mathbf{o}$ as the output vector containing the binary decisions coming from the dichotomizers, i.e., $\mathbf{o}=\{o_1(\mathbf{x}),o_2(\mathbf{x}),\ldots,o_L(\mathbf{x})\}$ with $o_j(\mathbf{x}) \in \{-1,1\}$. To have a perfect prediction by the ECOC system, the output vector should totally match one of the original codewords. However, even though some dichotomizers output wrong prediction, this does not necessarily lead to a wrong prediction, which cannot be recovered. A common criterion to classify a sample $\mathbf{x}$ is to choose the class with the “closest” codeword to the output vector.

In this case, a major role is played by the Hamming distance D_H between two words, which is given by the number of position where the bit patterns of the two words differ, i.e.:

$$D_H(\mathbf{c}_i,\mathbf{c}_j)=\sum_{h=1}^{L}\frac{|c_{ih}-c_{jh}|}{2}=\mathrm{card}(\{h : c_{ih}\neq c_{jh}\}). \tag{1}$$

The minimum Hamming distance $d_{min}=\min_{i,j} D_H(\mathbf{c}_i,\mathbf{c}_j)$ between any pair of codewords is a measure of the quality of the code.[1] In particular it is possible to correctly decode any word which contains v erroneous bits if

$$v < \lfloor (d_{min}-1)/2 \rfloor. \tag{2}$$

In this way, a single bit error does not influence the result, as can happen when using the usual one-per-class coding, where the Hamming distance between each pair of strings is 2. The sample $\mathbf{x}$ is assigned to the class ω_k corresponding to the codeword with the minimum Hamming distance from the output vector:

$$\omega_k=\arg\min_h D_H(\mathbf{c}_h,\mathbf{o}). \tag{3}$$

2.2. Soft decoding

A disadvantage of the hard decoding technique is that it completely ignores the magnitude of the soft outputs, which represents an indicator of the reliability of the decision taken by the dichotomizer. Therefore, a common strategy is to consider the real-valued output of a dichotomizer $f_h(\mathbf{x})$ normalized in the interval $[-1,+1]$, and to collect the results into a real-valued output vector $\mathbf{f}$.

[1] In general, the coding matrix is built so as to have the largest d_{min} possible. This criterion is not respected when the coding matrix implements a particular decomposition, such as *one vs. one* or *one vs. all*. In this last case, e.g., $d_{min}=2$.

In this case, the decoding procedure can be accomplished in a way similar to hard decoding by measuring the distance between the output vector and a codeword through L_1 or L_2 norm distances [20]. Another possible approach is the *loss-based decoding* proposed in [1], which we will assume for the rest of the paper because it contains, as particular cases, the decoding schemes based on the L_1 and L_2 norms.

To introduce the loss-based decoding technique, we have to refer to the concept of *margin* [1]. If we assume that y is the correct label of a sample $\mathbf{x}$, the margin associated with the prediction of the dichotomizer f_h on the sample $\mathbf{x}$ is given by $yf_h(\mathbf{x})$. As a consequence, the sample margin is negative if f_h provides a wrong prediction for $\mathbf{x}$. More precisely, while the sign of the margin indicates the correctness of the classifier, the magnitude estimates the confidence of the classifier in making its prediction on the sample $\mathbf{x}$. The margin is instrumental in many learning algorithms that aim at maximizing the sample margins on a training set; in fact, there are theoretical justifications [29] as to why maximizing the margin may be beneficial for classifier generalization capability. However, as margin maximization in its general form can be intractable, such learning algorithms typically circumvent the computational complexities by introducing a *loss function* evaluated on the margin $\mathcal{L}(yf_h(\mathbf{x}))$, which can be more easily minimized through suitable algorithms.

In the case of ECOC, the margins associated with the choice of a particular codeword $\mathbf{c}_i$ are given by $c_{ih}f_h(\mathbf{x})$ with $h=1,\ldots,L$; if we know the loss function $\mathcal{L}(\cdot)$ used for training the dichotomizers,[2] we can use it for evaluating the global loss associated with such a codeword:

$$D_{\mathcal{L}}(\mathbf{c}_i,\mathbf{f}) = \sum_{h=1}^{L} \mathcal{L}(c_{ih}f_h(\mathbf{x})). \tag{4}$$

Such an expression is suitable for combining margin values into a *loss-based distance*, which gives a level of confidence on the output word [1]. To predict the label for the k-th class, analogously to previous cases, this rule can be used:

$$\omega_k = \arg\min_i D_{\mathcal{L}}(\mathbf{c}_i,\mathbf{f}). \tag{5}$$

As said earlier, Eq. (5) can also be used when the loss function of the dichotomizers is not known and the L_1 or L_2 norm distance is used. In fact, it is easy to see that the loss-based distance in (4) reduces to L_1 or L_2 distance provided that the loss $\mathcal{L}(z)=|1-z|$ or $\mathcal{L}(z)=(1-z)^2$ is used, respectively.

3. The external reject option

A common approach to decrease the costs of a classification system consists of turning as many errors as possible into rejects. In a real application, in fact, the cost of an error is typically higher than a reject cost; thus, an effective reject option is a general benefit for the original multi-class classification problem. Generally, a reject option is accomplished by evaluating the reliability of the decision taken by the classifier and rejecting such a decision if the reliability is lower than some given threshold. A complete description of the classification system with the reject option is given by the *error–reject curve* (shortly, *ER curve*) that plots the error rate $E(t)$ against the reject rate $R(t)$ when varying the threshold t on the reliability estimate (see Fig. 1).

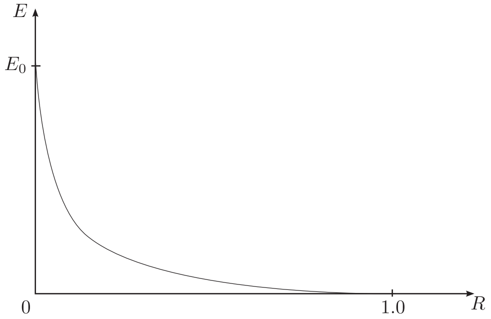

Fig. 1. A typical error–reject curve. E_0 is the error at 0-reject, i.e., the error rate provided by the classifier when the reject rule is not active. When the reject rate is 1.0 the error rate becomes null since all the inputs are rejected.

The simplest approach is to consider the entire ECOC system as a monolithic classification system. The main idea is to apply a reject option that modifies the decoding stage and evaluates the reliability of the multi-class decision through the possible decision rules that could be adopted; we can consider such a scheme as *external* because it works at the output of the whole classification system ignoring the internal structure.

Let us first consider an ECOC with hard decoding: in this case, the reliability can be assessed only by looking at the Hamming distance between the binary output vector $\mathbf{o}$ and the codewords in $\mathbf{C}$. In particular, the greater the Hamming distance is between the output vector and the nearest codeword, the greater is the probability of an erroneous decision and thus the lesser is the reliability of the decision. More precisely, the condition in Eq. (2) is not verified if the distance between the output vector and the correct codeword (i.e., the number of erroneous bits in the output vector) is higher than $\lfloor (d-1)/2 \rfloor$, where d is the Hamming distance between the codeword and its nearest neighbor in $\mathbf{C}$.

In other words, the reliability of the decision can be estimated by looking at the distance between the output vector and the nearest codeword. In this way, we can define a reject rule for the hard decoding based on the Hamming distance that introduces a reject region between each codeword and its nearest neighbor in $\mathbf{C}$ and abstains from the decision when the distance between the output vector and its nearest codeword is too high. If t_e is the reject threshold and ω_k is the class chosen according to Eq. (3), then the reject rule is

$$r(\mathbf{o},t_e) = \begin{cases} \omega_k & \text{if } D_H(\mathbf{c}_k,\mathbf{o}) < t_e, \\ \textit{reject} & \text{if } D_H(\mathbf{c}_k,\mathbf{o}) \geq t_e. \end{cases} \tag{6}$$

To better explain these distance-based reject rules, let us consider Fig. 2 where we have reported two output vectors $\mathbf{o}_1$ and $\mathbf{o}_2$ obtained by two samples belonging to the class ω_p with respect to the two nearest codewords in $\mathbf{C}$. In this case the first sample will be correctly decoded while the vector $\mathbf{o}_2$ will be assigned to the wrong class ω_q (Fig. 2a). When introducing a reject rule, a decision for the vector $\mathbf{o}_2$ will not be taken to avoid an error (Fig. 2b).

The block diagram in Fig. 3 illustrates a hard decoding scheme with the external reject option.

In this framework, we can also consider the approach described in [34] that employed a slightly different reject rule. In this method, not only the Hamming distance between the output vector and the nearest codeword is estimated, but also the difference between the output vector and the second closest codeword. In this case, the reject rule, that we will refer as *relevant codeword difference decoding*, becomes

$$r(\mathbf{o},t_r) = \begin{cases} \omega_k & \text{if } D_H(\mathbf{c}_k^2,\mathbf{o}) - D_H(\mathbf{c}_k,\mathbf{o}) > t_r, \\ \textit{reject} & \text{if } D_H(\mathbf{c}_k^2,\mathbf{o}) - D_H(\mathbf{c}_k,\mathbf{o}) \leq t_r, \end{cases} \tag{7}$$

where $\mathbf{c}_k^2$ is the second closest codeword to the output vector. In this case, the block diagram is the same as that illustrated in Fig. 3, with the obvious substitution of t_e with the threshold t_r.

[2] For example, $\mathcal{L}(z)=e^{-z}$ for Adaboost and $\mathcal{L}(z)=\max\{1-z,0\}$ for the SVM with linear kernel.

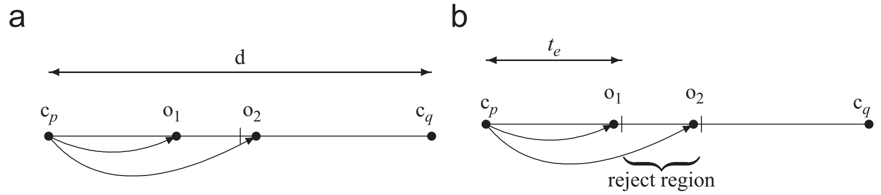

Fig. 2. An example of the distance-based decoding method with the standard approach (a) and with an external reject option (b).

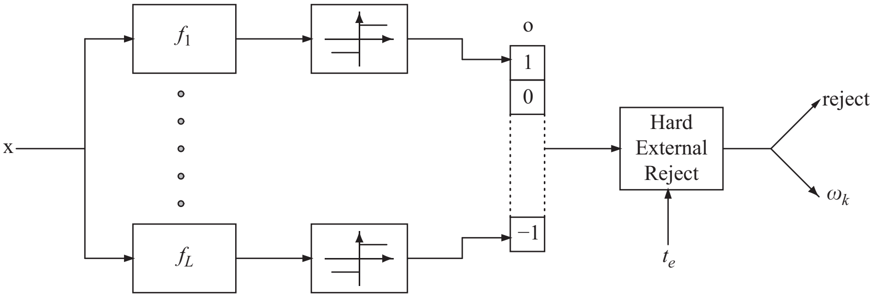

Fig. 3. The block diagram for hard decoding with external reject rule.

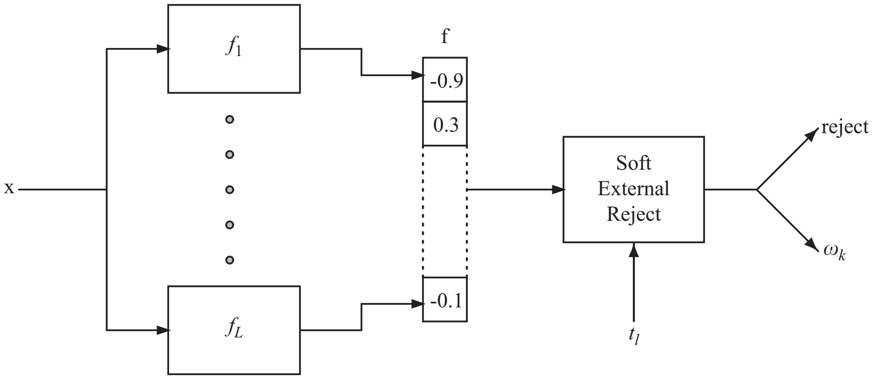

Fig. 4. The block diagram for soft decoding with external reject rule.

In the case of soft decoding, the reject option can exploit the loss function associated with the employed dichotomizer and evaluate the reliability of the decision by looking at the loss distance related to the decision. Assuming a loss value normalized in the range [0,1], such a criterion can be formalized as

$$r(\mathbf{f},t_l) = \begin{cases} \omega_k & \text{if } D_{\mathcal{L}}(\mathbf{c}_k,\mathbf{f}) < t_l, \\ \textit{reject} & \text{if } D_{\mathcal{L}}(\mathbf{c}_k,\mathbf{f}) \geq t_l, \end{cases} \tag{8}$$

where ω_k is the class chosen according to Eq. (5) and $t_l \in [0,1]$. The resulting scheme is illustrated in Fig. 4.

It is worth mentioning that the external schemes do not require any assumption either on the dichotomizers or on the coding matrix; thus, they can be applied in both the decoding techniques previously described without modifying the internal organization of the ECOC system. Finally, for both the schemes of external reject option, it is very simple to build the ER curve by varying the threshold (whether t_e or t_l) and observing the errors and rejects obtained.

4. The internal reject option

The previous section described a general approach to provide an ECOC system with a reject option uniquely based on the observation of the output of the ensemble of classifiers. A more in-depth analysis can be carried out if we explicitly take into account the structure of the ECOC system and estimate the reliability of the output of each dichotomizer. This implies that the dichotomizers provide a real value (e.g., in the range $[-1,+1]$) on which a threshold is set in the case of hard decoding, while it is used as such in soft decoding.

In this situation, a new scheme (*internal* scheme) can be introduced, which can evaluate the reliability of the outputs coming from the dichotomizers and rejects the decisions not sufficiently reliable. The idea is to single out the unreliable elements (binary decisions) in the output vector and process them in an appropriate way before arriving at the decoding stage. As a consequence, the decoding rules have to be modified to take their decisions only on the basis of the bits that are evaluated as sufficiently reliable. Two different possibilities are considered. The first one refers to hard decoding and is derived from the Coding Theory. In particular, the rejected outputs are considered as "erasures" in a codeword transmitted over a Binary Erasure Channel [21]. This is a typical model in Coding Theory, which furnishes a tailored hard decoding technique called *erasure filling*. The second approach is based on the loss-based distance and extends the soft decoding technique previously shown.

4.1. Designing the reject option for the dichotomizers

Before going into details about the decoding techniques, let us consider how to design the reject option for the group of dichotomizers in the ECOC architecture. Let us assume that each dichotomizer $f_h(\mathbf{x})$ outputs a real value in the range $[-1,+1]$ and that, to take a decision about the class of $\mathbf{x}$, the value $f_h(\mathbf{x})$ is compared with a threshold τ_h. In other words, $\mathbf{x}$ is assigned the class $+1$ if $f_h(\mathbf{x}) \geq \tau_h$ otherwise, the class -1 is chosen. It is worth noting that irrespective of the value of the decision threshold τ_h, the majority of unreliable decisions correspond to the outcome values near the threshold, where the distributions of the two classes overlap. In other words, the samples for which the output of the dichotomizer falls in this region are characterized by some ambiguity in the allocation, because their corresponding outcomes are very similar and thus quite difficult to distinguish. A way to obtain more reliable results is to employ a decision rule with two thresholds, τ_{h1} and τ_{h2} with $\tau_{h1} \leq \tau_{h2}$, such that

$$r(f_h,\tau_{h1},\tau_{h2}) = \begin{cases} +1 & \text{if } f_h(\mathbf{x}) > \tau_{h2}, \\ -1 & \text{if } f_h(\mathbf{x}) < \tau_{h1}, \\ \textit{reject} & \text{if } f_h(\mathbf{x}) \in [\tau_{h1},\tau_{h2}]. \end{cases} \tag{9}$$

The idea is to encapsulate the class overlap region into the *reject interval* $[\tau_{h1},\tau_{h2}]$, so as to turn many of the errors due to the class overlap into rejects. Generally, the optimal values for the thresholds (τ_{h1},τ_{h2}) should be chosen to satisfy two contrasting requirements: enlarging the reject region to eliminate more errors and limiting the reject region to preserve as many correct classifications as possible. In our case, we cannot choose the same pair of thresholds for all the dichotomizers because each of them has different distributions for the output score, and a unique choice would involve abnormal results for most of them (an example is shown in Fig. 5).

Accordingly, we imposed all dichotomizers to work at a chosen rejection rate ρ and used the method described by Pietraszek in [22]. It requires estimating the ROC curve of each dichotomizer and calculating the pair of thresholds (τ_{h1},τ_{h2}) such that f_h abstains for no more than ρ at the lowest possible error rate. The rationale is to make all the dichotomizers work almost at the same level of reliability.

4.2. Erasure filling decoding

When dealing with hard decoding, the activation of the internal reject option creates "no decision" bits where a reject has been produced. Thus, the output vector contains, besides the usual output values, another special symbol (indicated as $*$) for the rejects. A resume is given by the following equation:

$$o_h^{(\rho)}(\mathbf{x}) = \begin{cases} +1 & \text{if } f_h(\mathbf{x}) > \tau_{h2}, \\ -1 & \text{if } f_h(\mathbf{x}) < \tau_{h1}, \\ * & \text{if } f_h(\mathbf{x}) \in [\tau_{h1},\tau_{h2}], \end{cases} \tag{10}$$

where the exponent in the symbol $o_h^{(\rho)}$ remembers that the output is dependent on the value of the parameter ρ. The decoding stage now has to correct the possible errors and fill the erasures. One way to decode is to guess the erased symbols and then to exploit the error correction capability of the code; if all the resulting errors are less than half of the minimum distance, then they can be corrected and the right class can be recovered. On this basis, the Coding Theory provides a simple erasure filling procedure [21] summarized as in Algorithm 1.

Algorithm 1. Erasure decoding.

```
1: Place $-1$ in all erased positions and decode to the closest
   codeword $\mathbf{c}^{(-1)}$;
2: Place $+1$ in all erased positions and decode to the closest
   codeword $\mathbf{c}^{(+1)}$;
3: Choose the $\mathbf{c}^{(j)}$, with $j=-1,+1$, closest to the received
   codeword in the unerased positions.
```

The first two steps of the algorithm are meant to solve the rejects/erasures, while the last one exploits the error correction capability of the code. The rationale is to replace all the erasures with -1 and decode; if at least half of the erasures are correctly replaced and the condition in Eq. (2) is verified, then the decoding will be correct. On the other hand, if most of the erasures should have been filled with 1, then the decoding may introduce many further errors. In this case, replacing all the erasures with 1 will give a more correct decoding. Therefore, the procedure requires two filling-decode rounds: if the obtained codewords are different, the closer to the unerased part of the output vector is chosen.

In the presence of the erasures, the minimum distance between the codewords is decreased accordingly, and this obviously affects the error correcting capability. In particular, to

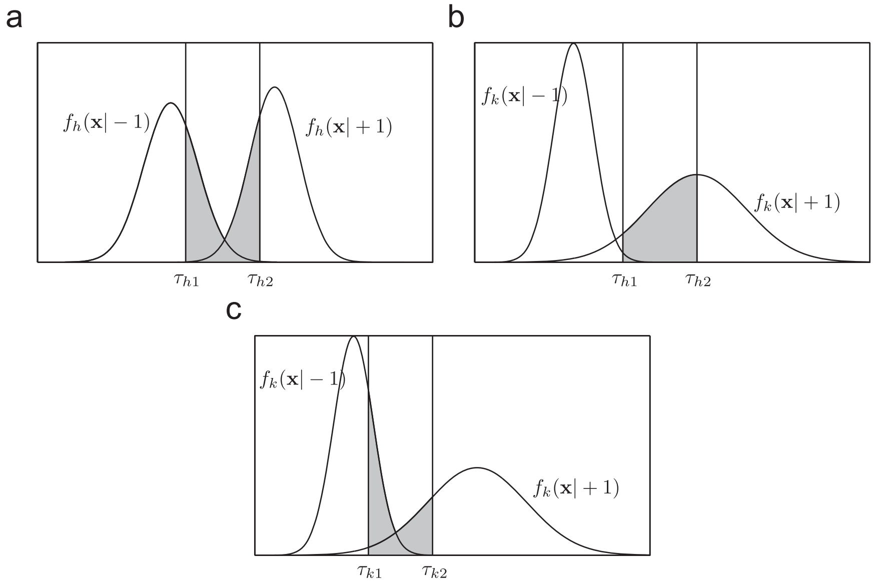

Fig. 5. Choosing the proper reject interval. (a) The distributions of the outputs of the dichotomizer f_h for the two classes and the pair of thresholds (τ_{h1},τ_{h2}). (b) The same thresholds applied to a different dichotomizer f_k produce abnormal results. (c) The proper thresholds (τ_{k1},τ_{k2}) evaluated for f_k.

have a correct decision, the number of errors (v) and erasures (μ) should verify the following condition:

$$2v+\mu<d_{min}. \tag{11}$$

This equation has a clear meaning [21]: it is twice difficult to correct an error than an erasure, because the position of the erasures in the output word is known, while it is impossible to know where the errors occurred. To analyze the effects of Eq. (11), Fig. 6 shows the plane representing the numbers of errors (vertical axis) against erasures (horizontal axis) in an output word. Two regions can be distinguished to illustrate what can happen for an L-length codeword. The output vectors falling into the white region ("full recovery" zone) are those that verify the condition in Eq. (11) and thus can be completely recovered. On the other hand, the output vectors falling in the gray region ("uncertainty" zone) exceed the minimum Hamming distance d_{min}, and this does not make it profitable to apply the erasure decoding, because the decision is very likely to be wrong. In such cases, it is advisable to reject the sample, i.e., to suspend the decision of the whole system. However, the diagram in Fig. 6 cannot be practically used for detecting the output vectors to be rejected because the exact number of erroneous bits and the exact position of the output vector on the plane are unknown. A suitable approximation is to abstain from decision when the number of erasures is greater than or equal to d_{min} (i.e., on the right-hand side of the dotted line in Fig. 6). In other words, we are actually assuming that the unerased bits are sufficiently reliable to guarantee that there are no errors among them (Fig. 7).

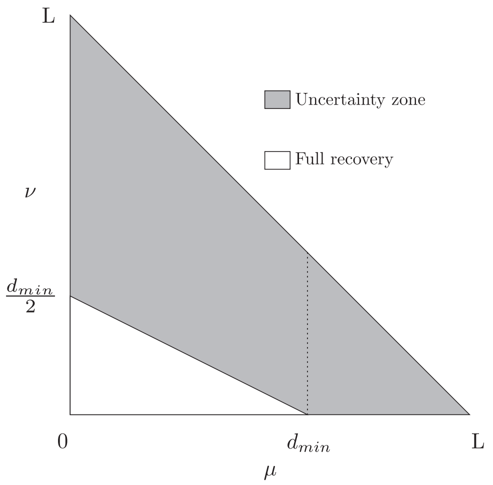

Fig. 6. The effects of Eq. (11) as a function of the number of errors (v) and erasures (μ).

On this basis, the step 3 in Algorithm 1 can be substituted by the following rule:

$$r(\mathbf{o}^{(\rho)},\mathbf{C})=\begin{cases}\omega_k^{(-1)} & \text{if } D_H^*(\mathbf{o}^{(\rho)},\mathbf{c}^{(-1)})<D_H^*(\mathbf{o}^{(\rho)},\mathbf{c}^{(+1)})\wedge\mu<d,\\ \omega_k^{(+1)} & \text{if } D_H^*(\mathbf{o}^{(\rho)},\mathbf{c}^{(+1)})<D_H^*(\mathbf{o}^{(\rho)},\mathbf{c}^{(-1)})\wedge\mu<d,\\ \textit{reject} & \text{if } \mu\geq d,\end{cases} \tag{12}$$

where D_H^* is the Hamming distance calculated on the unerased bits. Even in this case, the ER curve can be simply drawn by varying the parameter ρ and observing the errors and rejects obtained at the final decoding stage of the system.

4.3. *Trimmed loss decoding*

The reject option for the dichotomizers described in Section 4.1 can also be applied in the case of soft decoding. The only difference with respect to the hard decoding case is in the value produced by the classifier in the case of a reject, which is now assumed to be 0:

$$f_h^{(\rho)}(\mathbf{x})=\begin{cases}0 & \text{if } f_h(\mathbf{x})\in[\tau_{h1},\tau_{h2}],\\ f_h(\mathbf{x}) & \text{otherwise,}\end{cases} \tag{13}$$

where we assumed the same notation in Eq. (10).

It is worth noting that the null value is also a possible outcome for the dichotomizer without the reject option. It corresponds to the particular case when the sample falls on the decision boundary and thus it is assigned neither to the positive nor negative label. In this case, the loss calculated on the margin is $\mathcal{L}(c_{ih}f(\mathbf{x}))=\mathcal{L}(0)$, irrespective of the value of c_{ih}, while it is higher or lower than the "don't care" loss value $\mathcal{L}(0)$ if $f_h(\mathbf{x})\neq 0$ (depending on whether $c_{ih}f(\mathbf{x})$ is positive or negative). The reject option actually extends such behavior to all the samples whose outcome $f_h(\mathbf{x})$ falls within the reject interval $[\tau_{h1},\tau_{h2}]$, as shown in Fig. 8. In this way, a part of the values assumed by the loss is cut away and is not considered in the final decision procedure, which is called *trimmed loss decoding*. The block diagram of this internal rejection scheme is given in Fig. 9.

The final loss value is now modified by the presence of zero values in the output word $\mathbf{f}^{(\rho)}$. More precisely, it is given by

$$D_{\mathcal{L}}(\mathbf{c}_i,\mathbf{f}^{(\rho)})=\sum_{h\in I_{nz}}\mathcal{L}(c_{ih}f_h(\mathbf{x}))+|I_z|\cdot\mathcal{L}(0), \tag{14}$$

where I_{nz} and I_z are the sets of indexes of nonzero values and zero values in the output word, respectively. In practice, the loss is given by two contributions, where the second one is independent of the codeword that is compared with the output word. This modifies the range of the loss values, as shown in Fig. 10.

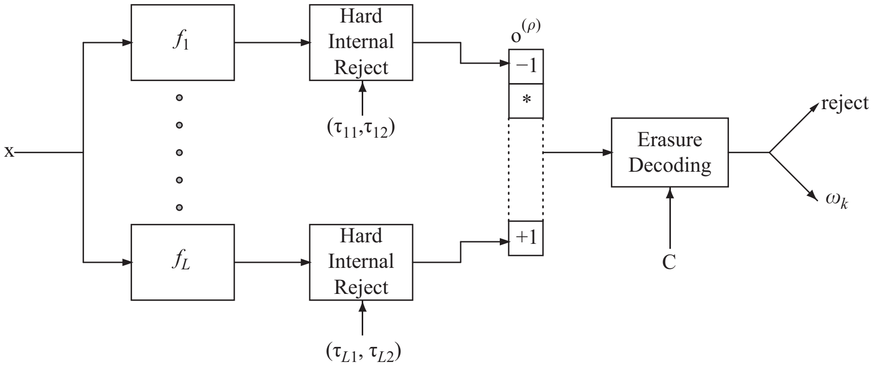

Fig. 7. The block diagram for the hard internal reject rule followed by the erasure filling decoding. The thresholds (τ_{h1},τ_{h2}) for $h=1,\ldots,L$ are evaluated for each dichotomizer according to [22].

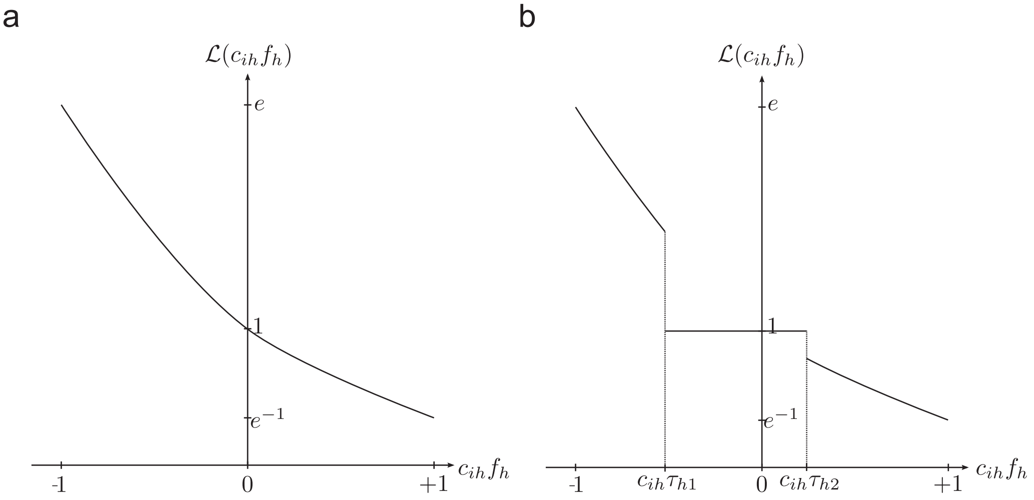

Fig. 8. (a) The loss function evaluated on the margin $c_{ih}f_h$, where we assume $c_{ih}=+1$ and $\mathcal{L}(z)=e^{-z}$. (b) The effects on the loss function produced by the introduction of the pair of thresholds (τ_{h1},τ_{h2}).

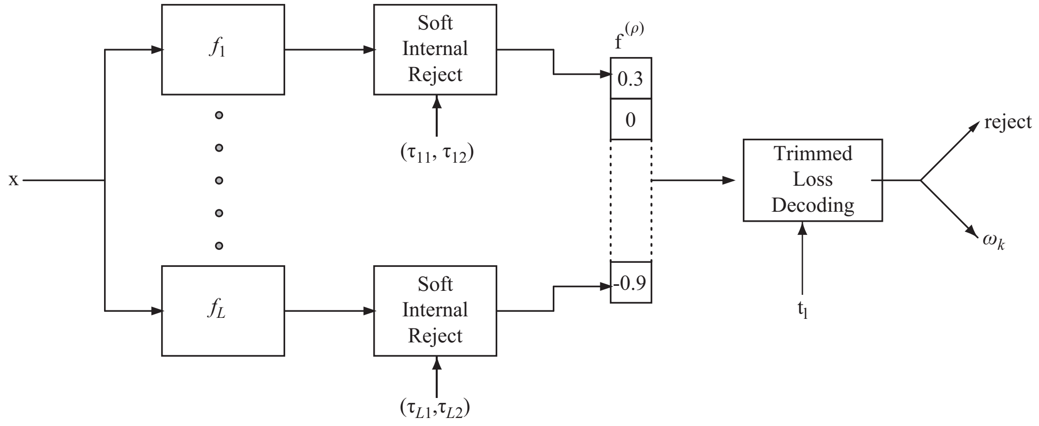

Fig. 9. The block diagram for the soft internal reject rule followed by the trimmed loss decoding. The thresholds (τ_{h1},τ_{h2}) for $h=1,\ldots,L$ are evaluated for each dichotomizer according to [22].

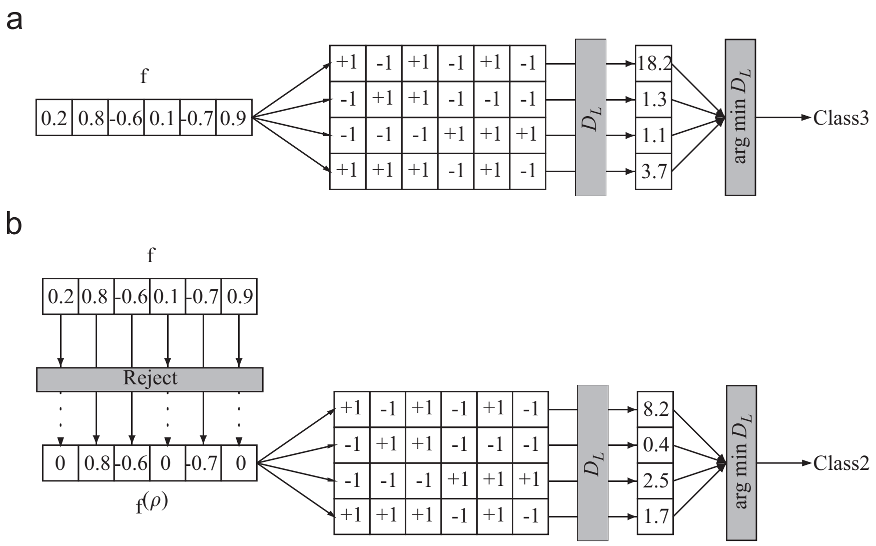

Fig. 10. Loss distance-based decision on the whole codeword (a) and on a trimmed output word (b) using an exponential loss function.

Chosen a threshold value t_l, a rejection rule can be immediately defined as

$$r(\mathbf{f}^{(\rho)},t_l)=\begin{cases}\omega_k & \text{if } D_{\mathcal{L}}(\mathbf{c}_k,\mathbf{f}^{(\rho)})<t_l,\\ \textit{reject} & \text{if } D_{\mathcal{L}}(\mathbf{c}_k,\mathbf{f}^{(\rho)})\geq t_l,\end{cases} \tag{15}$$

where ω_k is the class chosen according to Eq. (5).

The resulting decision rule depends on two different thresholds (ρ and t_l), while the previously described rules depend only on one parameter. In the previous cases, the reject option generates a unique curve in which each point is a function of only one threshold, while now we have a family of ER curves, each produced by a particular value of ρ (Fig. 11). However, which one should be assumed as the ER curve of the whole system?

To have some insight about this question, let us consider two ER curves corresponding to two different values ρ_1 and ρ_2 of the internal reject threshold. They can be arranged into two different ways: one of the curves can be completely below the other one (see Fig. 12a) or they can intersect (see Fig. 12b). In the first case, the lower curve (and the corresponding ρ) must be preferred, because it achieves a better error rate at the same reject rate or a better reject rate at the same error rate. The second case shows different regions in which one of the curves is better than the other one, and thus there is no curve (and an internal reject threshold value) that is definitely optimal. Therefore, to obtain an optimal ECOC system under all circumstances, the ER curve should include the locally optimal parts of the two curves. This is obtained if we assume the convex hull of the two curves as the ER curve of the ECOC system (see Fig. 12c).

This can be easily extended to the curves related to all the values considered for ρ so as to assume the convex hull of all the curves as the ER curve of the ECOC system (see Fig. 13).

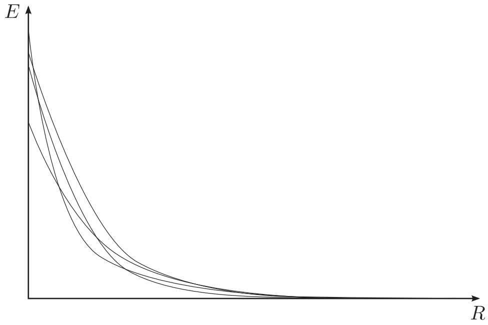

Fig. 11. The ER curves generated by various values for ρ.

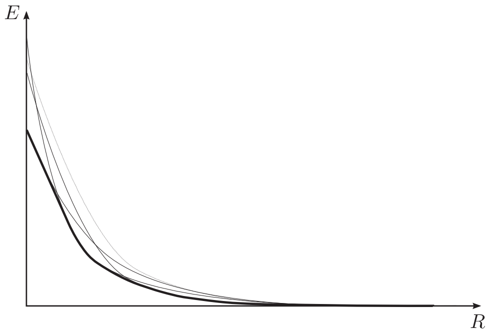

Fig. 13. The convex hull of the ER curves shown in Fig. 11 assumed as the ER curve of the ECOC system. The dotted curve is dominated by the other ER curves and thus, it does not give any contribute.

5. Experimental results

To test the performance of the proposed reject rules, 10 multi-class data sets publicly available at the UCI machine learning repository [2] have been used. To avoid any bias in the comparison, 12 runs of a multiple hold-out procedure have been performed on all the data sets. In each run, the data set has been split into three subsets: a training set (containing the 70% of the samples of each class), a validation set and a test set (each containing the 15% of the samples of each class). The training

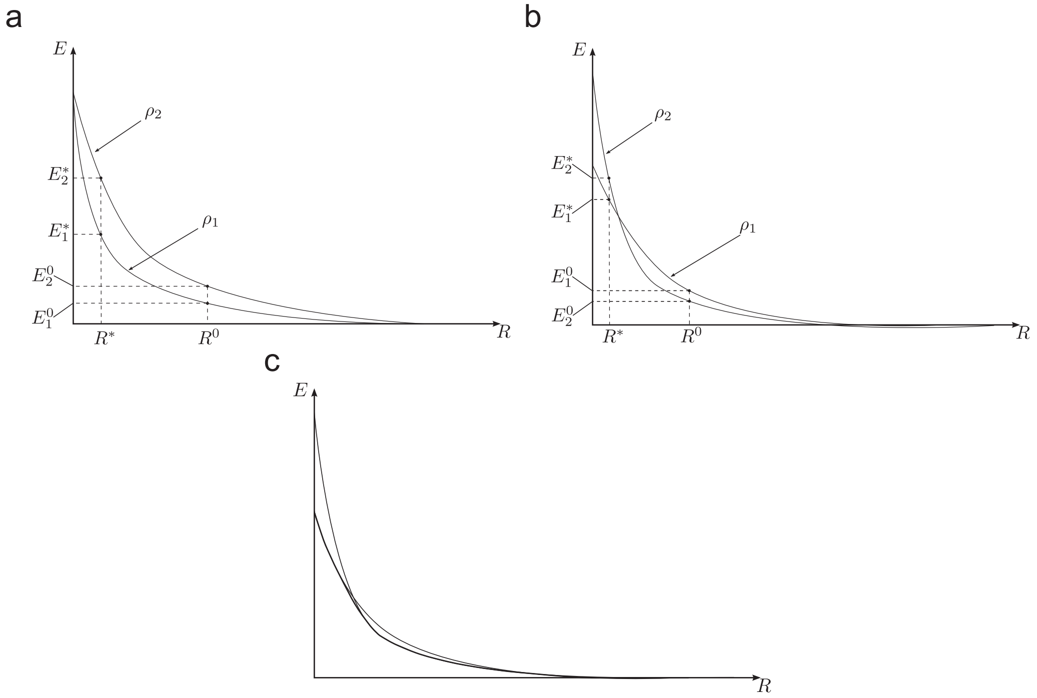

Fig. 12. (a) The ER curve produced by ρ_1 dominates the curve produced by ρ_2: for different values of R, E_1 is always less than E_2. (b) There is no ER curve dominating the other: for R^* it is $E_1^*<E_2^*$, while for R^0, $E_1^0>E_2^0$. (c) The convex hull of the ER curves shown in (b), which includes the locally optimal parts of the two curves.

set has been used to train the base classifiers, the validation set to normalize the outputs into the range $[-1, 1]$ and to calculate the thresholds (τ_{h1},τ_{h2}), and the test set to evaluate the performance of the classification system. A short description of the data sets has been reported in Table 2. In the same table, we also report the number of columns of the coding matrix chosen for each data set according to [9].

Three different classifiers have been used as base dichotomizer: Modest Adaboost (MA) [30] and support vector machine (SVM) with both linear and RBF kernel. MA was built using a decision tree with maximum depth equal to 10 as weak learner and 50 boosting steps. SVM was implemented through the libSVM [4] software library using a linear kernel and an RBF kernel. All the parameters, i.e., the γ value for the RBF kernel and the C values for both kernels, were set according to the optimizing procedure of cross validation for model selection of the software library. The exponential loss $\mathcal{L}(z)=e^{-z}$ for the MA and the "hinge" loss $\mathcal{L}(z)=\max\{1-z,0\}$ for the SVM were adopted.

As an example of the obtained results for the three classifiers, we show in Figs. 14 and 15 the ER curves calculated by averaging the ER curves obtained in the 12 runs of the multiple hold-out procedure. The range for the reject rate on the x-axis has been limited to [0,0.30] since higher reject rates are typically not of interest in real applications.

Fig. 14 reports a comparison of the three external reject rules: the hard decoding (HD), the soft decoding (SD) and the relevant codeword difference decoding (RCD), proposed by Zhou et al. in [34]. Since HD is based on the Hamming distance, the threshold has been varied in the range $[0,L/2]$ with step 1. For RCD the difference between the two Hamming distances has been normalized with respect to the codeword length L and the threshold varied between [0,1] with step 0.05. For the SD the loss output has been normalized in the range [0, 1] and, consequently, the thresholds were varied in this interval with the step 0.01. As expected, in the majority of cases SD outperforms RCD and HD. RCD behaves similar to HD, even though it frequently shows slightly better performance. Actually, while HD simply decides for the nearest codeword, RCD is able to detect and correctly manage situations of ambiguous decision where the output word has similar distance from two distinct codewords. The overall better performance of SD shows that the knowledge of architectural details of the dichotomizers (i.e., the nature of their outputs and the knowledge of their loss functions) can be fruitfully exploited to improve the performance of the ECOC classification system.

Fig. 15 focuses on the internal reject rules, indicated respectively with EFHD (erasure filling with hard decoding) and TLD (trimmed loss decoding). In both cases, we have varied the parameter ρ from 0 to 1 with step 0.05. As in the SD case, the loss output was normalized in the range [0,1] and the external threshold was varied with step 0.01 into the same range. It is evident that the EFHD technique always performs worse than TLD for each data set and classifier (except on Yeast data set using an RBF SVM where the performance are quite similar, see Fig. 15f). Also this set of experiments confirms that the quality of the ECOC classification system improves if we know the learning criteria adopted by the dichotomizers and we are able to exploit such knowledge.

To make a complete comparison among the different strategies, we have finally considered as performance figure the area under the ER curve evaluated on the whole range [0,1] of the reject rate, which provides the average error rate when varying the reject rate. This is a single numerical value which provides a synthetic and clear measure of a classifier with the reject option: the lower its value is, the better is the overall classification performance.

Tables 3, 4 and 5 report the results for MA, linear SVM and RBF SVM classifier respectively, obtained with the multiple hold out procedure before described. Each cell contains the values of the mean area under the ER curve and its standard deviation for each data set and reject rule.

First, we have tested the significance of differences between multiple means using the Friedman test [12], a nonparametric equivalent of the ANOVA that is a more general test even if less powerful than ANOVA when its assumptions are met [8]. The Friedman test ranks the reject rules for each run separately according to their score (the highest mean area under the ER curve gets the rank 1, the second highest the rank 2, and so on); in the event of tied scores, the average of the involved ranks is assigned to all scores tied for a given rank. In our case, the null hypothesis for the Friedman test corresponds to a nonstatistically significant difference among the mean areas of the employed methods. When the null hypothesis is rejected (i.e., not all the rules are equivalent), we subsequently apply a post hoc test to identify the differing rules. In particular, Holm's step-down procedure [16] has been employed to find out which classifiers exhibit a statistically different behavior. According to the Friedman–Holm test, TLD was the best method in 8 of the 30 analyzed cases. In all the other cases the pair TLD-SD had always statistically significant better performance than all the other approaches. Only in one case, i.e., Ecoli when using a linear SVM, HD-TLD was the best pair. In all these cases, we have applied a further test (the Wilcoxon signed ranks-test [31]) to break the tie between the two best methods and verify if there is a statistical difference between them. The Wilcoxon test assessed the superiority of TLD in all the analyzed cases with respect to SD but no statistical difference has been found with HD on Ecoli data set with a linear SVM. The outcomes of the statistical tests are shown in the reported tables where a bold value on each row indicates the method that obtains the statistically lowest mean area according to the Friedman–Holm test and/or to the Wilcoxon test. All the tests have been performed with a significance level equal to 0.05. Once again, the results indicate that the soft decoding techniques work better than hard decoding, i.e., using the soft outputs of the dichotomizers improves the quality of the classification.

Some considerations may be made about how to employ the described methods when dealing with different applicative scenarios. Actually, the whole choice depends on the architecture of dichotomizer we can apply in the particular application at hand. Depending on their outputs, we can have two kinds of dichotomizers: a crisp dichotomizer which outputs only a class label (e.g., a syntactic or rule-based classifier) or a soft dichotomizer that assigns a real-valued confidence degree to the input sample. It is apparent that the soft dichotomizers contain a higher level of information about the decision taken and that it can be reduced to a crisp classifier by imposing a threshold on the output.

The nature of the dichotomizer should thus guide the choice of the technique to be used. The crisp dichotomizer cannot use the

Table 2
The data sets used in the experiments.

Data set	Classes	Features	Length (L)	Samples
Abalone	29	8	30	4177
Ecoli	8	7	62	341
Glass	6	9	31	214
Letter	26	16	63	20 000
OptDigits	10	62	31	5620
Pendigits	10	12	31	10 992
SatImage	6	36	31	6435
Segmentation	7	18	63	2310
Vowel	11	10	14	435
Yeast	10	8	31	1484

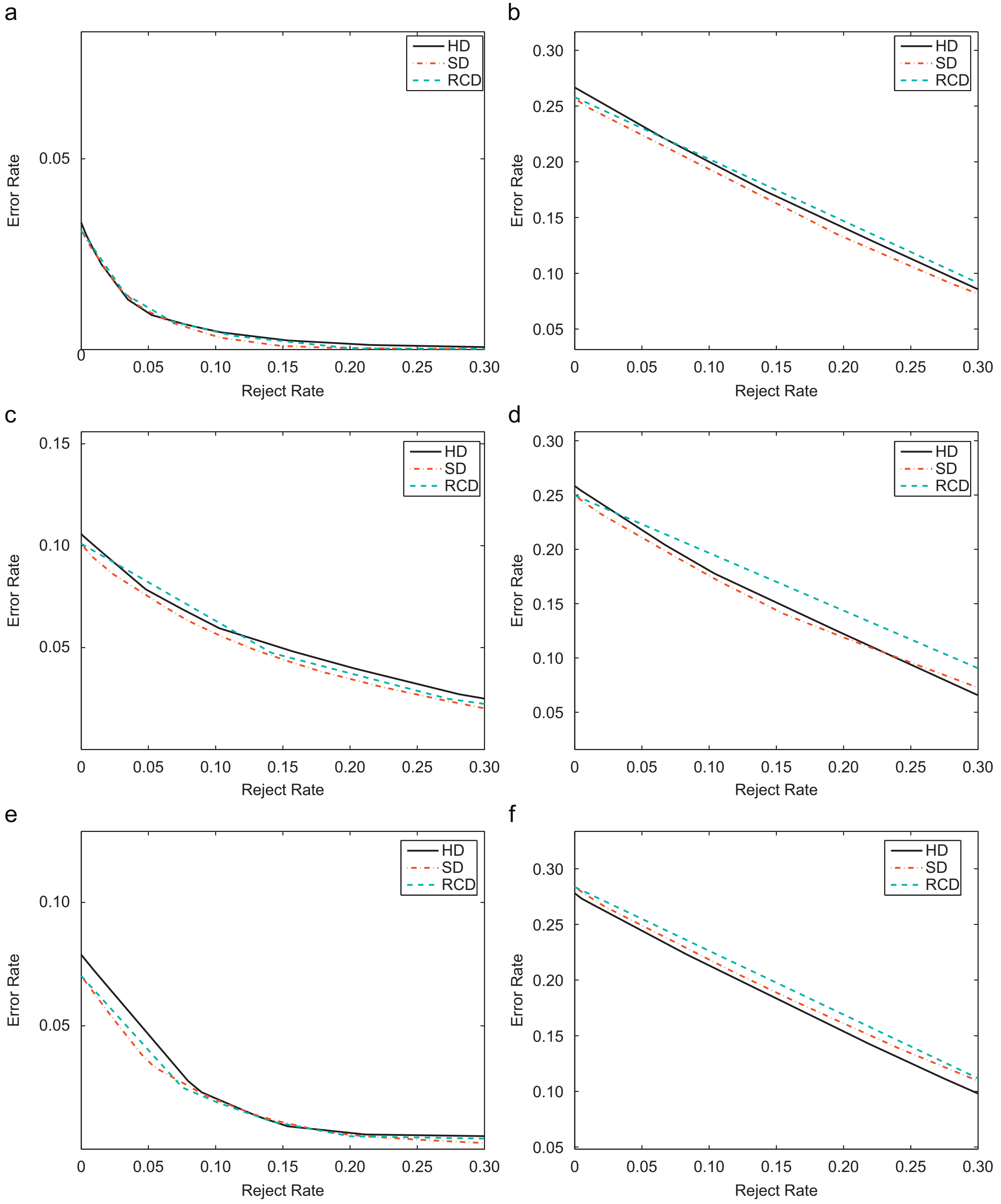

Fig. 14. ER curves for the external reject rules on Segmentation (a, c, e) and Yeast (b, d, f) data sets using Modest Adaboost (a, b), linear SVM (c, d) and RBF SVM (e, f).

internal reject option and thus the choice is limited to HD and RCD (which are both based on hard decoding with external reject option), even though the experiments have shown that RCD works slightly better than HD, specially with more proficient dichotomizers. On the other hand, soft dichotomizers can in principle employ all the described techniques, but the experiments have shown that the soft decoding techniques work better than hard decoding. In other words, the accuracy of the whole ECOC system is significantly improved when all the information available about the decision process is used.

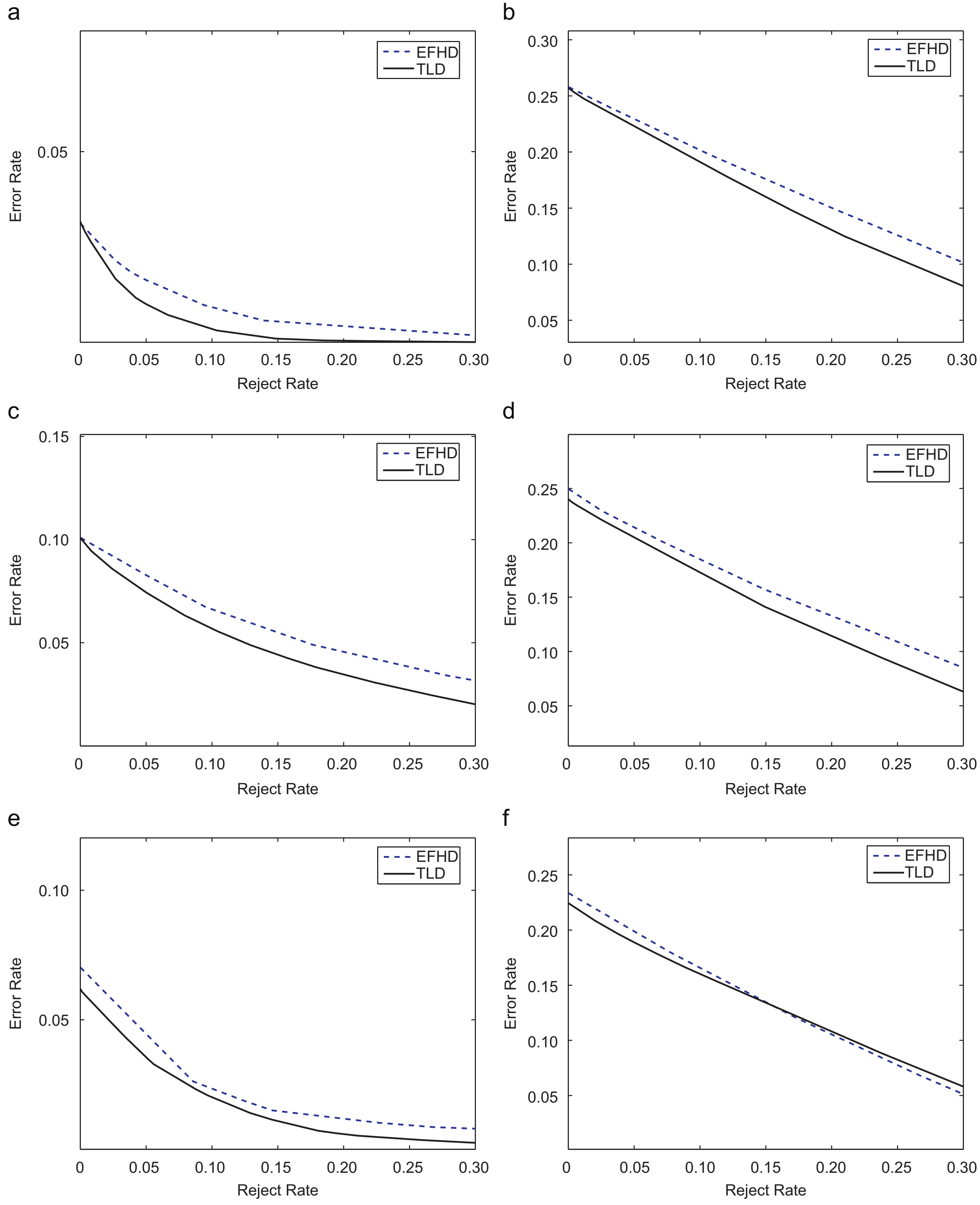

Fig. 15. ER curves for the internal reject rules on Segmentation (a, c, e) and Yeast (b, d, f) data sets using Modest Adaboost (a, b), linear SVM (c, d) and RBF SVM (e, f).

As a last remark, the experiments have clearly shown that internal reject is useful when coupled with soft decoding but not with hard decoding. Even though this could lead to conclude that the internal reject should be applied only when a soft decoding technique is used, we have also to notice that the particular recovery method of the erased bits does not always work properly

Table 3
Mean area under the ER curve and standard deviation (MA as base classifier).

Data sets	HD	SD	RCD	EFHD	TLD
Abalone	0.359 (0.010)	0.348 (0.009)	0.362 (0.005)	0.369 (0.011)	**0.342** (**0.010**)
Ecoli	0.032 (0.016)	0.036 (0.015)	0.036 (0.015)	0.035 (0.016)	**0.026** (**0.011**)
Glass	0.094 (0.026)	0.090 (0.019)	0.092 (0.020)	0.101 (0.025)	**0.069** (**0.017**)
Letter	0.061 (0.005)	0.052 (0.007)	0.064 (0.007)	0.065 (0.006)	**0.050** (**0.006**)
OptDigits	0.006 (0.001)	0.004 (0.001)	0.010 (0.003)	0.008 (0.002)	**0.004** (**0.001**)
Pendigits	0.003 (0.001)	0.003 (0.001)	0.003 (0.001)	0.004 (0.001)	**0.002** (**0.001**)
SatImage	0.025 (0.005)	0.021 (0.003)	0.023 (0.003)	0.024 (0.002)	**0.021** (**0.003**)
Segmentation	0.002 (0.001)	0.001 (0.001)	0.002 (0.001)	0.003 (0.002)	**0.001** (**0.001**)
Vowel	0.052 (0.011)	0.037 (0.010)	0.040 (0.011)	0.064 (0.011)	**0.032** (**0.010**)
Yeast	0.169 (0.015)	0.160 (0.015)	0.166 (0.012)	0.170 (0.016)	**0.151** (**0.012**)

Table 4
Mean area under the ER curve and standard deviation (linear SVM as base classifier).

Data sets	HD	SD	RCD	EFHD	TLD
Abalone	0.368 (0.011)	0.352 (0.012)	0.371 (0.008)	0.397 (0.010)	**0.344** (**0.007**)
Ecoli	**0.024** (**0.014**)	0.031 (0.011)	0.033 (0.012)	0.037 (0.013)	**0.023** (**0.009**)
Glass	0.174 (0.024)	0.177 (0.021)	0.191 (0.019)	0.184 (0.026)	**0.135** (**0.022**)
Letter	0.169 (0.010)	0.158 (0.008)	0.203 (0.006)	0.209 (0.004)	**0.147** (**0.006**)
OptDigits	0.006 (0.001)	0.005 (0.001)	0.006 (0.001)	0.010 (0.002)	**0.005** (**0.001**)
Pendigits	0.020 (0.003)	0.018 (0.003)	0.025 (0.004)	0.033 (0.005)	**0.017** (**0.004**)
SatImage	0.037 (0.006)	0.032 (0.003)	0.035 (0.003)	0.040 (0.004)	**0.031** (**0.003**)
Segmentation	0.019 (0.004)	0.018 (0.004)	0.019 (0.004)	0.022 (0.004)	**0.015** (**0.004**)
Vowel	0.284 (0.024)	0.225 (0.019)	0.267 (0.021)	0.240 (0.019)	**0.213** (**0.019**)
Yeast	0.183 (0.011)	0.187 (0.010)	0.190 (0.011)	0.192 (0.012)	**0.171** (**0.013**)

Table 5
Mean area under the ER curve and standard deviation (RBF SVM as base classifier).

Data sets	HD	SD	RCD	EFHD	TLD
Abalone	0.339 (0.011)	0.328 (0.007)	0.355 (0.005)	0.376 (0.008)	**0.322** (**0.008**)
Ecoli	0.033 (0.021)	0.034 (0.018)	0.034 (0.016)	0.038 (0.018)	**0.027** (**0.013**)
Glass	0.136 (0.039)	0.112 (0.023)	0.106 (0.023)	0.124 (0.024)	**0.099** (**0.028**)
Letter	0.029 (0.004)	0.026 (0.004)	0.026 (0.003)	0.030 (0.005)	**0.024** (**0.005**)
OptDigits	0.003 (0.001)	0.001 (0.001)	0.002 (0.001)	0.002 (0.001)	**0.001** (**0.000**)
Pendigits	0.002 (0.001)	0.001 (0.001)	0.001 (0.001)	0.002 (0.001)	**0.001** (**0.000**)
SatImage	0.035 (0.003)	0.020 (0.004)	0.021 (0.004)	0.023 (0.004)	**0.019** (**0.003**)
Segmentation	0.007 (0.002)	0.005 (0.002)	0.006 (0.001)	0.008 (0.004)	**0.004** (**0.001**)
Vowel	0.006 (0.004)	0.003 (0.002)	0.003 (0.002)	0.004 (0.002)	**0.002** (**0.001**)
Yeast	0.173 (0.019)	0.176 (0.018)	0.180 (0.017)	0.170 (0.019)	**0.166** (**0.016**)

and produces output words very different from the codewords used for representing the classes. As a consequence, it is advisable to consider and test further recovery methods.

6. Conclusions

In this paper, we have discussed two different ways to enrich an ECOC classification system with a reject option. Each technique has been fully explored in conjunction with both the possible decoding techniques (hard and soft decoding) used in the ECOC systems. Accordingly, we have defined four different methods that can be used for real applications.

The main difference is in the simplicity and ease of use of the external methods against the flexibility of internal methods. External techniques only require an intervention at the final stage of the ECOC system, while internal techniques involve more parameters to be managed. However, the geometrical method described in Section 4.3 provides a suitable tool to make the internal techniques particularly effective. This is particularly useful because the experiments showed that the more complex technique attains higher reduction of error rate. The trimmed loss technique generally gave the best overall results because it allows to single out and suitably manage the individual errors of the base classifiers. As a consequence, the system has the possibility to face the uncertainty of wrong predictions in a more precise and effective way than external techniques.

Directions for future work include investigating the relation of the rejection rule with the characteristics of the coding, taking special account of the new problem or data-dependent designs. Moreover, we believe that a more thorough analysis of the techniques provided by Coding Theory could be advisable. In fact, there are some theoretical frameworks that could allow us to exploit the characteristics of particular coding/decoding techniques, which guarantee a more effective recovery of the erased bits.

References

[1] E.L. Allwein, R.E. Schapire, Y. Singer, Reducing multiclass to binary: a unifying approach for margin classifiers, Journal of Machine Learning Research 1 (2000) 113–141.
[2] A. Asuncion, D.J. Newman, UCI Machine Learning Repository, 2007.
[3] P.L. Bartlett, M.H. Wegkamp, Classification with a reject option using a hinge loss, Journal of Machine Learning Research 9 (2008) 1823–1840.
[4] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (27) (2011) 1–27.

[5] C. Chow, On optimum recognition error and reject tradeoff, IEEE Transactions on Information Theory 16 (1) (1970) 41–46.

[6] L.P. Cordella, C. De Stefano, F. Tortorella, M. Vento, A method for improving classification reliability of multilayer perceptrons, IEEE Transactions on Neural Networks 6 (5) (1995) 1140–1147.

[7] K. Crammer, Y. Singer, On the learnability and design of output codes for multiclass problems, in: N. Cesa-Bianchi, S.A. Goldman (Eds.), COLT, Morgan Kaufmann, 2000, pp. 35–46.

[8] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

[9] T.G. Dietterich, G. Bakiri, Solving multiclass learning problems via error-correcting output codes, Journal of Artificial Intelligence Research 2 (1995) 263–286.

[10] S. Escalera, O. Pujol, P. Radeva, On the decoding process in ternary error-correcting output codes, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (1) (2010) 120–134.

[11] S. Escalera, D.M.J. Tax, O. Pujol, P. Radeva, R.P.W. Duin, Subclass problem-dependent design for error-correcting output codes, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (6) (2008) 1041–1054.

[12] M. Friedman, A comparison of alternative tests of significance for the problem of m rankings, Annals of Mathematical Statistics 11 (1940) 86–92.

[13] G. Fumera, F. Roli, Analysis of error-reject trade-off in linearly combined multiple classifiers, Pattern Recognition 37 (6) (2004) 1245–1265.

[14] R. Ghani, Combining labeled and unlabeled data for text classification with a large number of categories, in: IEEE International Conference on Data Mining, 2001, pp. 597–598.

[15] T. Hastie, R. Tibshirani, Classification by pairwise coupling, in: M.I. Jordan, M.J. Kearns, S.A. Solla (Eds.), Advances in Neural Information Processing Systems, vol. 10, The MIT Press, 1998.

[16] S. Holm, A simple sequentially rejective multiple test procedure, Scandinavian Journal of Statistics 6 (1979) 65–70.

[17] J. Kittler, R. Ghaderi, T. Windeatt, J. Matas, Face verification via error correcting output codes, Image and Vision Computing 21 (13–14) (2003) 1163–1169.

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

[19] C. Marrocco, P. Simeone, F. Tortorella, A framework for multiclass reject in ECOC classification systems, in: B.K. Ersbøll, K.S. Pedersen (Eds.), SCIA, Lecture Notes in Computer Science, vol. 4522, Springer, 2007, pp. 313–323.

[20] F. Masulli, G. Valentini, An experimental analysis of the dependence among codeword bit errors in ECOC learning machines, Neurocomputing 57 (2004) 189–214.

[21] R.H. Morelos-Zaragoza, The Art of Error Correcting Coding, John Wiley & Sons, 2006.

[22] T. Pietraszek, On the use of ROC analysis for the optimization of abstaining classifiers, Machine Learning 68 (2) (2007) 137–169.

[23] O. Pujol, P. Radeva, J. Vitrià, Discriminant ECOC: a heuristic method for application dependent design of error correcting output codes, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (6) (2006) 1007–1012.

[24] P. Simeone, C. Marrocco, F. Tortorella, Exploiting system knowledge to improve ECOC reject rules, in: ICPR, IEEE, 2010, pp. 4340–4343.

[25] S. Singh, A. Kodali, K. Choi, K.R. Pattipati, S.M. Namburu, S.C. Sean, D.V. Prokhorov, L. Qiao, Dynamic multiple fault diagnosis: mathematical formulations and solution techniques, IEEE Transactions on Systems, Man, and Cybernetics, Part A 39 (1) (2009) 160–176.

[26] C. De Stefano, C. Sansone, M. Vento, To reject or not to reject: that is the question-an answer in case of neural classifiers, IEEE Transactions on Systems, Man, and Cybernetics, Part C 30 (1) (2000) 84–94.

[27] E. Tapia, P. Bulacio, L. Angelone, Recursive ECOC classification, Pattern Recognition Letters 31 (3) (2010) 210–215.

[28] F. Tortorella, Reducing the classification cost of support vector classifiers through an ROC-based reject rule, Pattern Analysis & Applications 7 (2) (2004) 128–143.

[29] V.N. Vapnik, Statistical Learning Theory, Wiley, 1998.

[30] A. Vezhnevets, V. Vezhnevets, Modest adaboost—teaching adaboost to generalize better, in: Graphicon-2005, 2005.

[31] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bulletin 1 (6) (1945) 80–83.

[32] T. Windeatt, G. Ardeshir, Boosted ECOC ensembles for face recognition, in: Proceedings of the International Conference on Visual Information Engineering, 2003, pp. 165–168.

[33] T. Windeatt, R. Ghaderi, Coding and decoding strategies for multi-class learning problems, Information Fusion 4 (1) (2003) 11–21.

[34] J. Zhou, H. Peng, C.Y. Suen, Data-driven decomposition for multi-class classification, Pattern Recognition 41 (January) (2008) 67–76.

[35] J. Zhou, C.Y. Suen, Unconstrained numeral pair recognition using enhanced error correcting output coding: a holistic approach, in: ICDAR, IEEE Computer Society, 2005, pp. 484–488.

Paolo Simeone received the M.Sc. degree in Telecommunications Engineering in 2005 and the Ph.D. degree in Information and Electrical Engineering in 2009, both from University of Cassino, Italy. Currently, he is in a postdoctoral position. His research interests include statistical learning, multiple classifiers systems and cost sensitive classification. He is a member of the International Association for Pattern Recognition (IAPR).

Claudio Marrocco received the M.Sc. degree in Telecommunications Engineering in 2003 and the Ph.D. degree in Information and Electrical Engineering in 2007, both from University of Cassino, Italy. In 2009 he joined the Department of Automation, Electromagnetism, Information Engineering and Industrial Mathematics at the University of Cassino where currently he is an assistant professor in the Faculty of Engineering. He authored more than 20 publications and served as referee for many international journals. His research interests include statistical learning, multiple expert systems and medical image analysis and classification. He is a member of the International Association for Pattern Recognition (IAPR).

Francesco Tortorella is a full professor of Computer Science at the University of Cassino, Italy. He has authored over 80 research papers in international journals and conference proceedings. His current research interests include classification techniques, statistical learning, medical image analysis and interpretation. Prof. Tortorella is a member of the IAPR and a senior member of the IEEE.