A Genetic Inspired Optimization for ECOC

Miguel Ángel Bautista[1,2], Sergio Escalera[1,2], Xavier Baró[2,3], and Oriol Pujol[1,2]

[1]Dept. Matemàtica Aplicada i Anàlisi, Universitat de Barcelona, Gran Via 585, 08007 Barcelona, Spain.
[2]Centre de Visió per Computador, Campus UAB, Edifici O, 08193 Bellaterra, Barcelona, Spain.
[3]EIMS, Universitat Oberta de Catalunya, Rambla del Poblenou 156, 08018, Barcelona.
{mbautista,opujol,sescalera}@ub.edu
xbaro@uoc.edu

Abstract. In this work, we propose a novel Genetic Inspired Error Correcting Output Codes (ECOC) Optimization, which looks for an efficient problem-dependent encoding of the multi-class task with high generalization performance. This optimization procedure is based on novel ECOC-Compliant crossover, mutation, and extension operators, which guide the optimization process to promising regions of the search space. The results on several public datasets show significant performance improvements as compared to state-of-the-art ECOC strategies.

Keywords: Error-Correcting Output Codes, Genetic Optimization, Ensemble learning

1 Introduction

A challenging task in Pattern Recognition is to develop efficient methodologies to process huge amount of data. Concretely, classification procedures present a lack of options when the number of categories is arbitrarily large. In this scope, the Error Correcting Output Codes (ECOC) framework has shown great performance results. At the ECOC *coding* step, a set of binary partitions of the original problem are encoded in a matrix of codewords (one code per class, univocally defined) which are learnt by binary classifiers. Then, at the ECOC *decoding* step a final decision is obtained by comparing the set of binary predictions with every class code, and choosing the class with the code at minimum 'distance'. Standard ECOC coding strategies need between N and $\binom{N}{2}$ classifiers to deal with a $N-$class problem (using the One vs. All and the One vs. One coding designs, respectively). This implies a scalability problem when dealing with a large number of classes. Recently, some works applied Genetic Algorithms (GA) to find a sub-optimal ECOC configuration. The underlying idea of GA is to reproduce the natural evolution by means of computer programs, using a chromosome based representation of the problems, and implementing from a functional point of view the processes involved in nature (crossover and mutation). Various works

have treated the optimization of ECOC matrices with GA [2,6,5]. Nevertheless, they fail in taking into account the ECOC constraints, implying an unnecessary enlargement of the search space.

In this work, we propose a novel framework for treating the optimization of an ECOC matrix inspired on GA. In this framework the operators have been completely redefined in order to avoid non-valid individual generation, and thus, minimizing the search space in relation to previous works. In addition, the code length is reduced to be sub-linear in the number of categories, building both reduced and high-performance codes. This novel procedure is tested on several public datasets, obtaining significant performance improvements compared to state-of-the-art ECOC approaches.

The paper is organized as follows: Section 2 presents the novel genetic approach. Section 3 shows the experimental results and Section 4 concludes the paper .

2 ECOC-Compliant Genetic Algorithm

In this section we review the ECOC framework, its properties, and present the Genetic-ECOC.

2.1 ECOC framework

The ECOC framework is composed of two different steps: *coding* and *decoding* [1]. At the coding step an ECOC coding matrix $M_{N\times n} \in \{-1, +1, 0\}$ is constructed, where N denotes the number of classes in the problem and n the number of bi-partitions defined to discriminate the N classes. In this matrix, the rows (also known as *codewords*) are univocally defined, since these are the identifiers of each category in the multi-class problem. On the other hand, the columns of M denote the set of bi-partitions, dichotomies, or meta-classes to be learnt by each base classifier h^j (also known as dichotomizer). Hence, classifier h^j is responsible for learning the bi-partition denoted on the $j-$th column of M [1]. From the learning point of view, the performance of the ECOC ensemble will increase as more bi-partitions are taken into account. However, by taking into account the problem idiosyncrasies the system is able to obtain great performance by using few bi-partitions.

At the decoding step a new sample s is classified according to the N possible categories. In order to perform the classification task, each dichotomizer predicts a binary value for s whether it belongs to one of the bi-partitions defined by the corresponding dichotomy. Once the set of predictions $x(s) \in \mathbb{R}^n$ is obtained, it is compared to the codewords of M using a distance metric δ, known as the decoding function.

[1] For notation purposes we will refer to the entry of M at the i-th row and the j-th column as $M_{i,j}$

2.2 ECOC Coding Matrix Properties

We define an ECOC coding matrix $M_{N\times n} \in \{-1, +1, 0\}$ to be constrained by,

$$\min(\delta_{AHD}(y^i, y^k)) \geq 1, \forall i, k : i \neq k,\ i, k \in [1, \ldots, N] \tag{1}$$

$$\min(\delta_{HD}(d^j, d^l)) \geq 1, \forall j, l : j \neq l,\ j, l \in [1, \ldots, n] \tag{2}$$

$$\min(\delta_{HD}(d^j, -d^l)) \geq 1, \forall j, l : j \neq l,\ j, l \in [1, \ldots, n] \tag{3}$$

where δ_{AHD} and δ_{HD} are the Attenuated Hamming Distance (AHD) and the Hamming Distance (HD) are defined as in [4].

2.3 Genetic Inspired ECOC Optimization

In this section we present the novel Genetic-ECOC.

Problem encoding In order to consider the ECOC properties and obtain smart heuristics to guide the optimization process, a novel representation of ECOC individuals is proposed. ECOC individuals are represented as structures $I =< M, C, H, P, E, \delta >$, where the fields are defined as follows,

• The **coding matrix**, $M_{N\times n} \in \{-1, +1, 0\}$ where $n \geq \lceil \log_2 N \rceil$. For the initial population we fix $n = \lceil \log_2 N \rceil$, where n can grow along generations.

•The **confusion matrix**, $C_{N\times N}$, over the validation subset. Let c^i and c^j be two classes of our problem, then the entry of C at the i-th row and the j-th column, defined as $C_{i,j}$, contains the number of examples of class c^i classified as examples of class c^j.

• The **set of dichotomizers** $H =< h^1, \ldots, h^n >$.

• The **performance of each dichotomizer**, $P \in \mathbb{R}^n,\ P = [p^1, \ldots, p^n]$. This vector contains the proportion of correctly classified examples over a validation subset for each dichotomizer in H.

• The **error rate**, E, over a validation subset. This scalar is the proportion miss-classified samples of the validation subset using the Loss-Weighted decoding [4]. Let the set of samples in the validation subset be $V =< (s_1, l(s_1)), \ldots, (s_v, l(s_v)) >$, then E is defined as,

$$E = \sum_{j=1}^{v} I(\Delta(M, x^{s_j}), l(s_j))/v, \tag{4}$$

$$\Delta(M, x) = \underset{i}{\operatorname{argmin}}\ \delta(y_i, x),\ i \in \{1, \ldots, N\} \tag{5}$$

Fitness function The fitness function measures the environmental adaptation of each individual, and thus, is the one to be optimized. Individuals are evaluated according to the performance they obtain in the validation subset. Let E_{I_K} be the error rate of individual I_K and let n_{I_K} be the length of the coding matrix M of I_K, then, we define the fitness function as $F_f(I_k) = E_{I_k} + \lambda n_{I_k}$. [2]

[2] This expression (similar to the one showed by regularized classifiers), serves us to control the learning capacity of the ECOC matrix in order to not over-fit the training data.

ECOC Crossover and Mutation Operators In this section we introduce the novel ECOC crossover and mutation operators. These operators do not only take into account the restrictions of the ECOC framework (see Equations 1, 2, and 3) but also are carefully designed in order to avoid a premature convergence to local minima without generating non-valid individuals, and thus, converging to satisfying populations in fewer generations. In this sense, the crossover and mutation operators have two variants. The *Generic* one, which provides us with a tool to avoid premature convergence, and the *Specific* one, which guides the optimization to promising regions of the search space.

• **ECOC crossover algorithm**

Assume a N-class problem to be learnt and let I_F and I_M be two individuals encoded as shown in Section 2.3. Then, the crossover algorithm will generate a new individual I_S which coding matrix $M^{I_S}_{N\times n},\ n = \min(n_{I_F}, n_{I_K})$ contains dichotomies of each parent. Therefore, the key aspect of this recombination is the selection of which dichotomies of each parent are suitable to be combined. We introduce a dichotomy selection algorithm that chooses those n dichotomies that hold the constraints shown in Equations 1, 2, and 3. The dichotomy selection algorithm generates a dichotomy selection order $\tau^I \in \mathbb{R}^n$ for each parent I. Moreover, the selection algorithm checks if the separation between codewords is congruent with the number of dichotomies left to be added. In this sense, the $(k-i)$-th extension dichotomy will be only added if it splits the existing codewords to define $|Y| = r \leq 2^{(k-i)}$ codes at $\delta_{AHD}(y^a, y^b) = 0\ \forall\ y^a, y^b \in Y : a \neq b$, where k is the final length of the ECOC matrix. The Generic and Specific version of the ECOC crossover algorithm depend on how τ is defined. In the Generic version, τ is randomly generated, while in the Specific version τ is a classifier performance ranking.

In the crossover example shown in Figure 1 two individuals I_M and I_F are combined to produce a new offspring I_S. The crossover algorithm generates a dichotomy selection order τ for each parent. The first parent from which a dichotomy is taken is I_M, and d_3 is valid since $r \leq 2^{(3-1)} = 4$, and it only defines three codes without separation (y^1, y^2, and y^5). Once this step is performed, the parent is changed, and the following dichotomy will be extracted from I_F based on its selection order τ^{I_F}. In this case, d_4 is valid since $r \leq 2^{(3-2)} = 2$ and d_3 of I_M together with d_4 of I_F define only two equivalent codewords (y^1 and y^5). In the following iteration, the parent is changed again, and thus, I_M is used. Since $\delta_{AHD}(y^1, y^5) = 0$, d^1 can not be considered as an extension dichotomy, and therefore, the next dichotomy to use is d_2, which satisfies Equation 1 defining a valid ECOC coding matrix.

• **ECOC Mutation Algorithm**

Picture an individual I encoded as shown in Section 2.3 to be transformed by means of the mutation operator. This operator will select a set of positions $\mu =< M_{i,j}, \ldots, M_{k,l} >,\ i,k \in \{1,\ldots,N\},\ j,l \in \{1,\ldots,n\}$ of M^I to be mutated. The value of these positions is changed constrained to values in the set $\{-1, +1, 0\}$. In the Generic version, the set of positions μ are those valued 0. Once μ is defined, the positions are randomly recoded to one of the three possible values

```
Data: I_F, I_M
Result: I_S
n := min(M^{I_F}, M^{I_M}) // Minimum code length among parents
τ^{I_F} ∈ ℝ^n = selorder(I_F) // Dichotomy selection order of I_F
τ^{I_M} ∈ ℝ^n = selorder(I_M);
cp := I_F // Current parent to be used
M^{I_S} := ∅ // Coding matrix of the offspring
for i ∈ {1, ..., n} do
    for j ∈ {1, ..., n_cp} : τ_j^{cp} ≠ ∅ do
        f := 0 // Valid dichotomy search flag
        if calcRepetitions (M^{I_S}, d^{τ_j^{cp}}) ≤ 2^{(k-i)} then
            d^i := d^{τ_j^{cp}} // Inheritance of dichotomies
            h^i := h^{τ_j^{cp}} // Inheritance of dichotomizer
            p^i := p^{τ_j^{cp}} // Inheritance of performance
            τ_j^{cp} := ∅ // Avoid using a dichotomy twice
            f := 1 // Valid dichotomy found
            break;
        end
    end
    if !f then
        d^i := generateCol(M^{I_S}) // If non ECOC matrix can be built
        h^i := ∅;
        p^i := ∅;
    end
    if cp = I_F then
        cp := I_M // Dichotomy inheritance parent switch
    else
        cp := I_F;
    end
end
```

Algorithm 1: ECOC Crossover.

in $\{-1, +1, 0\}$. In the Specific mutation algorithm, the set of positions μ is chosen taking into account the confusion matrix C. Once these classes are obtained, the algorithm will mutate the bits valued 0 of its codewords $\{y^i, y^j\}$ in order to increment the distance $\delta_{AHD}(y_i, y_j)$. The specific ECOC mutation algorithm is shown in Algorithm 2.

In Figure 2 an example of the specific mutation algorithm is shown. Let I_T be an individual encoded as shown in Section 2.3. The confusion matrix C_{I_T} has its non-diagonal maximum at $C_{4,3} + C_{3,4}$. Then codewords y^4 and y^3 are going to be mutated. The 0 valued bits of this codewords are changed in order to increment $\delta_{AHD}(y^4, y^3)$, and thus, incrementing also the correction capability between them. At the following iteration $C_{4,3}$ is not taken into consideration and the procedure will be repeated with y^5 and y^4 which are the following classes that show confusion in C.

Problem-Dependent Extension Operator We propose an operator to extend ECOC designs based on the confusion matrix, focusing the extension of dichotomies on those categories which are difficult to be split. This methodology defines two types of extensions, the One vs. One extension (Generic extension) and the Sparse extension (Specific extension), which have the same probability of being executed along the optimization process. In the former, the ECOC coding

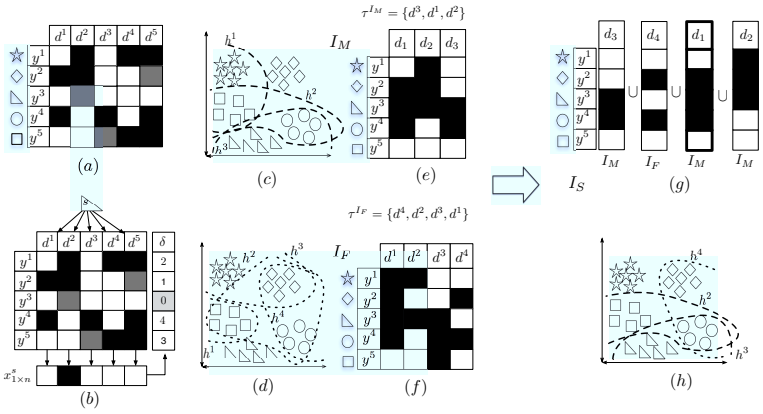

Fig. 1. (a). An example of an ECOC coding matrix. (b) Example of the decoding process. (c) Feature space and trained classifiers for parent I_M. (d) Feature representation and boundaries for parent I_F. (e) ECOC coding matrix of parent I_M. (f) Coding matrix of parent I_F. (g) ECOC coding matrix composition steps for the offspring I_S. (h) Feature space and inherited classifiers for I_S.

Data: I_T, mt_c
// Individual and mutation control value
Result: I_X
$C^{I_T}_{N\times N}$ // Confusion matrix of I_T
$k := 0$// Number of recoded bits of M^{I_T}
while $k < mt_c$ **do**
  $(c^i, c^j) := \underset{i,j}{\text{argmax}}(C_{i,j} + C_{j,i})\ \forall i, j : i \neq j;$
  for $b \in \{1, \ldots, n\}$ **do**
    if $|y^i_b| + |y^j_b| \leq 1$ **and** $k < mt_c$ **then**
      if $y^i_b = 0$ **and** $y^j_b = 0$ **then**
        $y^i_b := +1$ // Invert both bits valued 0
        $y^j_b := -1;$
      else
        if $y^i_b = 0$ **then**
          $y^i_b := -y^j_b$ // Invert bit valued 0
        else
          $y^j_b := -y^i_b;$
        end
      end
      $k := k + 1;$
    end
  end
  $C^{I_T}_{i,j} := 0,\ C^{I_T}_{j,i} := 0;$
end

Algorithm 2: Specific ECOC-Compliant Mutation.

matrix $M_{N\times n}$ will be extended with a dichotomy d^{n+1} which will be valued 0 except for those two positions d^i and d^j corresponding to the maximum confused classes $(c^i, c^j) = \underset{i,j}{\text{argmax}}(C_{i,j} + C_{j,i})$, which will be inverse valued. The latter,

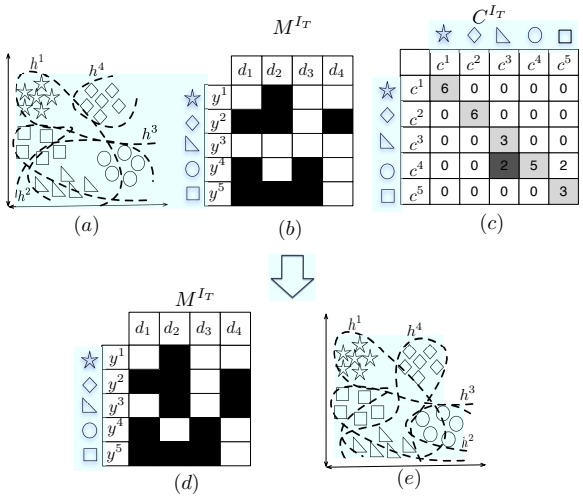

Fig. 2. Mutation example for a 5-class toy problem. (a) Feature space and trained dichotomizers for and individual I_T. (b) ECOC coding matrix of I_T. (c) Confusion matrix of I_T. (d) Mutated coding matrix. (e) Mutated feature space with trained dichotomizers.

follows the scheme in which two categories $\{c^i, c^j\}$ that maximize the confusion are discriminated.

3 Experimental Results

In order to present the results, we first discuss the data, methods, and evaluation measurements.

•**Data**: We consider five muti-class problems from the UCI Machine Learning Repository: Ecoli (8 classes), Vowel (11 classes), Yeast (10 classes), Shuttle (7 classes), and Glass (7 classes). In addition, we test our methodology over 4 challenging Computer Vision multi-class problems: 70 visual object categories from the MPEG dataset, 20 classes of the ARFace database, a real traffic sign categorization problem of 36 classes, and 7 handwritten music cleafs classes [2]. Computer Vision datasets are described using PCA keeping 99,9% of information.

• **Methods**: We compare the Genetic ECOC design with the One vs. All, One vs. One, Dense Random [1], Forest [8] and DECOC [7] designs. The ECOC base classifier is the libsvm implementation of a SVM with RBF kernel. The SVM ζ and γ parameters are tuned via Genetic Algorithms for all the methods, minimizing the classification error of a two-fold evaluation over a training sub-set [2].

• **GA settings and parameters**: The number of generations of each GA optimization process was set to $3N$ where N is the number of classes of each particular classification problem. The number of individuals of the GA was set

to $5N$. Furthermore, elitism was applied at each generation, and thus, the 10% fitter individuals are automatically selected to form part of the next generation. On the other hand, the specific and generic variants of the Crossover, Mutation and Extension operators where equiproportional.

• **Evaluation Measurements**: The classification performance is obtained by means of a stratified ten-fold cross-validation. Finally, we test for statistical significance using Friedman and Nemenyi statistics at 95% of the confidence interval [3]. The classification results are shown in Table 1. The table shows the classification performance of each ECOC design on each dataset, the average performance ranking, and the mean number of classifiers of the ensemble. In order to compare the performances provided for each strategy, Table 2 shows the mean rank of each ECOC design considering the 18 different experiments (9 dataset performances and 9 PC values).

Table 1. Classification results and number of classifiers per coding design.

Dataset	Compact ECOC		GA Ins. ECOC		D. Random ECOC	
	Perf.	Classif.	Perf.	Classif.	Perf.	Classif.
Ecoli	80.5±1.9	3	**81.4±1.3**	3.8	68.1±2.7	8
Vowel	48.6±3.5	3	54.4±4.3	3.2	42.8±1.1	7
Yeast	57.7±2.4	3	68.1±1.5	5.6	66.8±3.3	11
Shuttle	80.9±2.1	3	81.1±1.3	3.2	90.6±2.3	7
Glass	50.2±1.2	4	55.1±6.1	5	54.9±6.4	10
MPEG	90.8±4.1	6	95.3±3.2	6	83.3±1.0	36
ARFACE	61.5±3.2	5	86.3±1.2	6	73.0±1.3	20
TRAFFIC	81.2±1.2	3	96.3±2.4	4.2	82.3±1.1	7
CLEAFS	84.6±1.1	7	84.1±2.8	7	90.0±1.4	70
Mean Rank & #Class.	**5.6**	**4.2**	2.5	4.9	4.9	19.5

1vsAll		1vs1		DECOC		FECOC	
Perf.	Classif	Perf.	Classif.	Perf.	Classif.	Perf.	Classif.
75.5±1.8	8	79.2±1.8	28	69.4±1.3	7	75.2±3.5	21
53.8±6.2	7	**60.5±2.9**	15	55.1±2.5	6	43.9±2.1	15
80.7±2.2	11	78.9±1.2	28	66.7±1.3	10	68.1±1.3	30
90.6±1.1	7	86.3±1.1	21	77.1±1.4	6	80.3±1.5	18
47.1±1.3	10	52.4±2.8	45	55.8±2.2	9	**56.0±3.2**	27
91.8±2.6	36	90.6±2.1	630	86.2±4.2	35	**96.7±1.3**	105
84.0±3.3	20	**96.0±2.5**	190	82.7±2.1	19	81.6±0.4	57
80.8±1.2	7	84.2±2.8	21	96.9±2.4	6	**97.1±1.1**	18
87.8±2.4	70	**92.8±1.3**	2415	83.4±1.5	69	81.9±2.3	207
3.9	19.5	1.5	377	5.2	18.5	4.8	55.3

Table 2. Mean rank per coding design.

Rank	Compact ECOC	GA ECOC	Dense ECOC
Perf. rank	5.6	2.5	4.9
Perf. per Class rank	**1**	2	5
Mean rank	3.3	**2.2**	4.9

Rank	1vsAll	1vs1	DECOC	FECOC
Perf. rank	3.9	**1.5**	5.2	4.8
Perf. per Class rank	4	7	3	6
Mean rank	3.9	4.2	4.1	5.4

We use the Nemenyi test to check if one of the techniques can be singled out. In our case with $k = 7$ ECOC approaches to compare and $N = 9 \cdot 2 = 18$ experiments, the critical value for a 90% of confidence is $CD = 1.415 \cdot \sqrt{\frac{56}{108}} = 1.0189$. Since none of the methods ranks intersect with the GA Inspired ECOC rank for $CD = 1.0189$, we can state that the proposed ECOC design significantly improves the rest of methods performances at 90% of confidence.

4 Discussion and Conclusions

We presented the novel Genetic ECOC optimization procedure, which has been carefully defined in order to take into account the ECOC properties. New ECOC Crossover and Mutation operators have been defined to avoid non-valid coding matrix generation, reducing the search space and the number of individuals needed for convergence. Moreover, a new Extension ECOC operator has been proposed, which allows the ECOC design to take benefit from error correction in a problem dependent way. The methodology was tested on several public Machine Learning and Computer Vision datasets, obtaining significant performance improvements compared to state-of-the-art ECOC approaches using far less number of dichotomizers, which results in a much more efficient coding.

Acknowledgments This work has been supported by projects TIN2009-14404-C01/C02 ,CONSOLIDER-INGENIO CSD 2007-00018, IMSERSO-Ministerio de Sanidad 2011 Ref. MEDIMINDER and RECERCAIXA 2011 Ref. REMEDI.

References

1. E. Allwein, R. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *JMLR*, 1:113–141, 2002.
2. M. Ángel Bautista, S. Escalera, X. Baró, P. Radeva, J. Vitriá, and O. Pujol. Minimal design of error-correcting output codes. *PRL*, 33(6):693 – 702, 2012.
3. J. Demsar. Statistical comparisons of classifiers over multiple data sets. *JMLR*, 7:1–30, 2006.
4. S. Escalera, O. Pujol, and P.Radeva. On the decoding process in ternary error-correcting output codes. *TPAMI*, 99(1), 2009.
5. N. Garcia-Pedrajas and C. Fyfe. Evolving output codes for multiclass problems. *TEC*, 12(1):93 –106, 2008.
6. A. C. Lorena and A. C. P. L. F. Carvalho. Evolutionary design of multiclass support vector machines. *JIFS*, 18:445–454, October 2007.
7. O. Pujol, P. Radeva, and J. Vitrià. Discriminant ECOC: A heuristic method for application dependent design of ecoc. *TPAMI*, 28:1001–1007, 2006.
8. S.Escalera, O.Pujol, and P.Radeva. Boosted landmarks and forest ECOC: Framework to detect and classify objects in clutter scenes. *PRL*, 28(13):1759–1768, 2007.