1. INTRODUCTION

The task of supervised machine learning can be seen as the problem of finding an unknown function \( C(x) \) given the training set of example pairs \( \langle x_i, C(x) \rangle \). \( C(x) \) is usually a set of discrete labels. For example, in face detection, \( C(x) \) is a binary function \( C(x) \in \{\text{face, nonface}\} \), in optical digit recognition \( C(x) \in \{0, \ldots, 9\} \).

In order to address the binary classification task many techniques and algorithms have been proposed: decision trees, neural networks, large margin classification techniques, etc. Some of those methods can be easily extended to multiclass problems. However, some other powerful and popular classifiers, such as AdaBoost [1] and Support Vector machines [18], do not extend to multiclass easily. In those situations, the usual way to proceed is to reduce the complexity of the multiclass problem into multiple simpler binary classification problems.

There are many different approaches for reducing multiclass to binary classification problems. The simplest approach considers the comparison between each class against all the others. This produces \( N_c \) binary problems, where \( N_c \) is the number of classes. Other researchers suggested the comparison of all possible pairs of classes [3], resulting in an set of binary problems. In the literature, one can find several powerful binary classifiers. However, when one needs to deal with multiclass classification problems, many learning techniques fail to manage this information. Instead, it is common to construct the classifiers to distinguish between just two classes and to combine them in some way.

In this sense, Error-Correcting Output Codes (ECOC) were born as a general framework to combine binary problems to address the multiclass problem. The strategy was introduced by Dietterich and Bakiri in 1995. Based on the error correcting output codes design, ECOC represents a successful framework to deal with multiclass problems. Recent works in the ECOC framework showed significant performance improvements. The ECOC framework is a powerful tool to deal with multi-class categorization problems. As the error correcting output codes have error correcting ability and improve the generalization ability of the base classifiers. This paper describe both state-of-the-art coding (one-versus-one, one-versus-all, dense-random, sparse-random, DECOD forest-ECOC, and ECOC-ONE) and decoding designs (hamming, Euclidean, inverse hamming, laplacian, \( \beta \)-density, attenuated, loss-based, probabilistic kernel-based, and loss weighted). This paper also contains the empirical study on ECOC.

2. ERROR CORRECTING OUTPUT CODES

Steps for ECOC are as follows:

1) Given a set of \( N_c \) classes, the basis of the ECOC framework consists of designing a codeword for each of the classes.

2) These code words encode the membership information of each class for a given binary problem.

3) Arranging the code words as rows of a matrix, we obtain a coding matrix \( M_c \) where \( M_c \in \{-1, 0, +1\}^{N_c \times n} \), being \( n \) length of the code words codifying each class.

4) From the point of view of learning, \( M_c \) is constructed by considering in binary problems each one corresponding to a column of the matrix \( M_c \).

5) Each of these binary problems splits the set of classes in two partitions (coded by +1 or -1 in \( M_c \) according to their class set membership or 0 if the class is not considered by the current binary problem.

6) Then at the decoding step, we applying the \( n \) trained binary classifiers, a code is obtained for each data point in the test set.

7) This code is compared to the base code words of each class defined in the matrix \( M_c \), and the data point is assigned to the class with the closest codeword.

Below figure show ECOC coding design for a 4 class problem. White, black and grey positions corresponds to the symbols +1, -1 and 0. Once the four binary problems are learnt, at the decoding step a new test sample \( x \) is tested by the base classifiers. Then the new code word \( x = (\ldots, x_n) \) is compared with the class code words \( C_1, \ldots, C_4 \), classifying the new sample by the class \( C_i \) which codeword minimizes the decoding measure.

Fig. 1. ECOC Design
2.1. Coding Design

Here the ECOC coding design covers the state-of-the-art of coding strategies, mainly divided in two main groups: problem-independent approaches, which do not take in account the distribution of the data to define the coding matrix, and the problem-dependent designs, where in formation no the particular domain issue is to guide the coding design.

2.1.1. Problem-Independent ECOC Designs:

The methods of problem-independent are as follows:

- **One-versus-all** (Rifkin and Klautau, 2004): Nc dichotomizers are learnt for Nc classes, where each one splits one class from the rest of classes.
- **One-versus-one** (Nilsson, 1965): n = Nc(Nc−1)/2 dichotomizers are learnt for Nc classes, Splitting each possible pair of classes.
- **Dense Random** (Allwein et al., 2002): n = 10 \cdot \log Nc dichotomizers are suggested to be learnt for Nc classes, where P(−1) = P(+1), and P(±1) the probability of the symbols −1 and +1 to appear, respectively. Then, from a set of defined random matrices, the one which maximizes a decoding measure among all possible rows of Mc is selected.
- **Sparse Random** (Escalera et al., 2009): n = 15 \cdot \log Nc dichotomizers are suggested to be learnt for Nc classes, where P(0) = 1 − P(−1) − P(+1), defining a set of random matrices Mc and selecting the one which maximizes a decoding measure among all possible rows of Mc.

2.1.2. Problem-Dependent ECOC Designs:

- **DECOC** (Pujolet al., 2006): problem-dependent design that uses n = Nc−1 dichotomizers. The partitions of the problem are learnt by means of a binary tree structure using exhaustive search or a SFFS criterion. Finally, each internal node of the tree is embedded as a column in Mc.
- **Forest-ECOC** (Escalera et al., 2007): problem-dependent design that uses n = (Nc−1) \cdot T dichotomizers, where T stands for the number of Binary tree structures to be embedded. This approach extends the variability of the classifiers of the DECOC design by including extra dichotomizers.
- **ECOC-ONE** (Pujolet al., 2008): problem-dependent design that uses n = 2 \cdot Nc suggested dichotomizers. A validation sub-set issues due to extend any initial matrix Mc and to increase its generalization by including new dichotomizers that focus on difficult to split classes.

2.2. Decoding Design

The notation used refers to that used in (Escalera et al., 2008):

- **Hamming decoding**: HD(x, y) = \text{being } x a \text{ test codeword and } y a \text{ codeword from } Mc \text{ corresponding to class } Ci.
- **Inverse Hamming decoding**: HD(x, y) = \max (−1DT), where ([i, j]DT = HD(yi, yj), and D is the vector of Hamming decoding values of the test codeword x for each of the base code-words yj.
- **Euclidean decoding**: ED(x, y) = n \{x \cdot y \}
- **Attenuated Euclidean decoding**: PD(y(x)) = −\log \{\frac{\text{K}}{\text{K}+\text{Mc}(i,j)} \neq 0 \}
- **Laplacian decoding**: \text{LP}(x, y) = \text{where } o is the number of matched positions between x and yi, b is the number of miss-matches without considering the positions coded by 0, and K is an integer value that codifies the number of classes considered by the classifier.

3. OUTLINE OF ECOC ALGORITHM

**Training:**

1) Load training data and parameters, i.e., the length of code L and training class K.
2) Create a L-bit code for the K classes using a kind of coding algorithm.
3) For each bit, train the base classifier using the binary class (0 and 1) over the total training data.

**Testing:**

1) Apply each of the L classifiers to the test example.
2) Assign the test example the class with the largest votes.

3.1. What does a good ECOC?

The key problem for ECOC approach is how to design the coding matrix M. Many studies [10, 11, 12, 13, and 14] have shown that the final classifier will have good discriminate ability if the coding matrix M has the following characteristics:

1) **Rowseparation**: Each code word (a row in the coding matrix M) should be well-separated in Hamming distance from each of the other code words.
2) **Columnseparation**: Each column should be uncorrelated with one another. This means that the binary classifiers of different columns have low correlations among them.
3) **Binary classifiers have low Errors**: While for recognition of a large number of classes, Besides classification accuracy, the efficiency is also quite important. Take a quick decision, it is expected to evaluate as few binary classifiers as possible. This requires the code word to be efficient (i.e. contains a small number of bits). As explained in [15], for a code to be efficient, different bits should be independent of each other, and each bithasa 50% chance of being one or zero. In ECOC design, independent bits can be relaxed as uncorrelated columns (i.e., property 2 mentioned above). And 50% chance of firing for each bit requires:

4) **Balanced column**: For each column i, the numbers of 1 and −1 are equal, i.e., Finding an ECOC satisfying the above characteristics is a NP-hard problem.[16] So we can say that for efficient and accurate recognition of a large number of classes, a good ECOC is expected to have the following characteristics:

- Efficient—requires a small number of bits.
- Good diversity—the coding matrix has good row and column separation.
- The resulting binary classifiers are accurate.

3.2. What’s so good about ECOC?

1) Improves classification accuracy.
2) Can be used with many different classifiers.
3) 3. Commonly used in many areas.
4) Not prone to over fitting.
5) Possibly try a variant.

3.3. Practical Advantages of ECOC

1) It is fast, simple and easy to program
2) It is flexible can combine with any learning algorithm
3) Able to reduce the bias and variance produced by the learning algorithm. So it widely used to deal with multi-class categorization problems.
4) Low computational cost.
5) Out performs the direct multiclass method.
6) Can use with data that is textual, numeric, discrete etc.
7) Generally a running scheme-can be used Various learning tasks
8) Good generalization.

3.4. Disadvantages

1) ECOC is not effective if each individual codeword is not separated from each of the other code words with a large Hamming distance.
2) ECOC only succeeds if the errors made in the individual bit positions are relatively uncorrelated, so that the numbers of simultaneous errors in many bit positions is small. If there are many simultaneous errors, the ECOC will not able to correct them (Petersen & Weldon, 1972).
3) ECOC support vector machines are not always superior to one-against-all fuzzy support vector machines.
4) One-versus-all schemes are merest able than other ECOC schemes.
5) Some times decomposition of multi-class
6) Problem in to multiple binary problems we are doing in ECOC incurs considerable bias for centroid classifier which results in noticeable degradation of performance for centroid classifier.
7) Finding the optimal ECOC is NP hard.

3.5. Comparison of Some ECOC methods
3.5.1 One-Versus-All strategy
The most well-known binary coding strategies are the one-versus-all strategy [17], where each class is discriminated against the rest of classes. In Fig. 1a, the one-versus-all ECOC design for a four-class problem is shown. The white regions of the coding matrix M correspond to the positions coded by 1and the black region sto-1. Thus, the codeword for class C1 is (1,-1,-1,-1). Each column I of the coding matrix codifies a binary problem learned by its corresponding dichotomizer hi. For instance, dichotomizer h1 learns C1 against classes C2, C3, and C4, dichotomizer h2 learns C2 against classes C1, C3, and C4, etc.

3.5.2. The Dense Random Strategy
The dense random strategy [10], where a random matrix M is generated, maximizing the rows and columns separability in terms of the Hamming distance [7]. An example of a dense random matrix for a four-class problem is shown in Fig.1c.

3.5.3. One-Versus-One and Random Sparse Strategy
It was when Allwein et al. [10] introduced a third Symbol (the zero symbol) in the coding process when the coding step received special attention. This symbol increases the number of partitions of classes to be considered in a ternary ECOC framework by allowing one classes to be ignored. Then the ternary coding matrix becomes. In this case, the symbol zero means that a particular class is not considered by a certain binary classifier. Thanks to this, strategies such as one-versus-one [20] and random sparse coding [10] can be formulated in the framework. Fig. 1b shows the one-versus-one ECOC configuration for a four-class problem. In this case, the gray positions correspond to the zero symbols. A possible sparse random matrix for a four-class problem is shown in Fig. 1d.

4. CONCLUSIONS
In this paper the different coding and decoding methods for Error Correcting Output Code have been studied. Advantages and disadvantages of some ECOC coding methods are discussed. From this study on ECOC one can conclude that compared to other methods, better performance can be achieved by using Error Correcting Output Code.

Fig. 2. (a) One-versus-all (b) one-versus-one (c) Dense random and (d) sparse random ECOC Designs
3.5.3. One-Versus-One and Random Sparse Strategy

REFERENCE