*Ph.D. in Electronic and Computer Engineering*
*Dept. of Electrical and Electronic Engineering*
*University of Cagliari*

# Some Proposals for Combining Ensemble Classifiers

Nima Hatami

A dissertation submitted in partial fulfillment
of the requirements for the degree of Doctor of Philosophy
at the
University of Cagliari

*Advisor*: Prof. Giuliano Armano
*Curriculum*: ING-INF/05

XXIV Cycle
March 2012

## Thesis Committee Members

- Prof. Giuliano Armano

  Department of Electrical and Electronic Engineering, University of Cagliari, Italy

- Dr. Gavin Brown

  School of Computer Science, University of Manchester, UK

- Prof. Pekka Kalevi Abrahamsson

  Faculty of Computer Science, Free University of Bozen-Bolzano, Italy

## External Referees

- Prof. Michal Wolniak

  Department of Systems and Computer Networks, Faculty of Electronics, Wroclaw University of Technology, Poland

- Prof. Manuel Graña Romay

  Department of Computer Science and Artificial Intelligence, Universidad del País Vasco UPV/EHU, Spain

- Prof. José R. Villar Flecha

  Computer Sciences Department, University of Oviedo, Spain

## Research Stage Supervisor

- Prof. Ludmila I. Kuncheva

  School of Computer Science, Bangor University, UK

# اللَّطِيفُ

## Acknowledgements

It is a great pleasure to thank those people whose company I enjoyed in my lovely PhD journey.

It is difficult to overstate my gratitude to my friend and Ph.D. supervisor, Prof. Giuliano Armano. During these three years, he gave me an opportunity to learn not only how to be a thinker but most importantly be a life lover. Working with him was an honor which remains with me for ever.

Living away from family and hometown for the first time is a nice challenge. But you will learn to find new mates. Filippo Ledda was one who generously accepted me as a friend and helped me like *un amico*. My friends and colleagues in the IASC group, in particular Eloisa Vargiu, Alessandro Giuliani, Andrea Addis, and also Alessandro Giua, Carla Piras, Maria Paola Cabasino, Zisis Bimpisidis, Chiara Onnis, Giorgio Giacinto, Gian Luca Marcialis, Daniele Muntoni, Zahid Akhtar, Giorgio Fumera and Ajita Rattani who were there any time I needed. "Grazie a tutti voi!"

I would like to take this opportunity to thank Prof. Ludmila Kuncheva and Prof. Fabio Roli for their valuable time and trust on me; Prof. Michal Wozniak, Prof. Manuel Graña and Prof. José Ramón Villar for their nice reviews and comments on my thesis. This thesis is partially supported by Hoplo Srl.; in particular I would like to express my appreciation to Ferdinando Licheri and Roberto Murgia.

Lastly, and most importantly, I am indebted to my parents and grandma, Behnaz, Ali and Nazmaman. They supported me, taught me, trusted me, inspired me and loved me. Nina and Reza who are the hope and the reason. And to Camelia who will remain for ever. To them I dedicate this thesis. "Yashasin!!!"

February 2012, Cagliari

# Contents

# List of Figures

# List of Tables

# Overview

Most real-world pattern recognition problems are too complex to be efficiently handled using standard classification methods. Large number of classes or feature vectors, dataset sparsity, classes having high overlap, low number of samples available, and the need for online-training and classification are just some of the complexity issues that should be considered while designing a classification system. Some important examples that come from real-world applications are text categorization (characterized by large number of categories and words, sparsity of data and hierarchical relationships between class concepts) and cancer cell diognosis based on gene expression microarray information (HDLSS[1] problem).

*Combining classifiers* (in pattern recognition) or *ensemble learning* (in machine learning), based on the divide and conquer principle, has proved to be efficient in many of these complex situations. Ensemble methods such as bagging, boosting, Error-Correcting Output Codes (ECOC), Mixture of Experts (ME) and random forests use a combination of *simple* classifiers, working in a cooperative manner, instead of a single classifier responsible for the entire task. These approaches are able to obtain better recognition rates (bias reduction) and furthermore stabilize predictions (variance reduction).

This PhD thesis mainly focuses on theoretical and practical aspects of ensemble learning and multiple classifier systems. The novelty comes from developing new ideas by extending some classical approaches and standard algorithms, such as ME, random oracles, and ECOC. Two newer versions of ME, HME and random oracle have been proposed with the result of boosting their accuracy and efficiency. The standard ECOC method has also been extended, giving rise to the Multi-Label ECOC (ML-ECOC, hereinafter). The proposed ideas and methods have been assessed not only using publicly available benchmark datasets (from the UCI repository), as some real-world application areas have also been used for experiments.

The thesis is organized as follows: in chapter 1, a quick introduction to multiple classifier systems is given and some important algorithms described in the literature (such as bagging, boosting, random oracle, ME and ECOC) are briefly recalled. The main characteristics, pros, and cons of these algorithms different algorithms are also reported (this information may be helpful to identify which methods are expected to work better for a given problem).

Chapter 2 describes the proposed Random Prototype-based Oracle (RPO) method, which is an ensemble of miniensembles. Inspired by the random linear oracle model, the proposed method divides the problem space (i.e., the sample space) into smaller and hopefully simpler subspaces using randomly selected *prototype* points, rather than using hyperplanes as done with standard linear oracles. RPO has the advantage of decomposing the orginal problem

[1] high-dimensional low-sample size

into two or more subspaces, whereas linear oracles have the limitation of enforcing only binary decomposition.

Continuing with the idea of random decomposition of a complex problem, chapter 3 proposes *Mixture of Random Prototype-based Experts* (MRPE), together with its hierarchical version. Embedding a random prototype for each expert in the ME framework is the main idea of this method. In so doing, a simple distance-based rule can be used in both training and operation phases of an ME ensemble instead of a trainable gating network that needs to be trained together with the rest of experts. This simple modification boosts accuracy while reducing the overall time required for training the ensemble.

Finally, chapter 4 is about ECOC, applied to a text categorization problem. In the first subsection, we propose a metric extracted from ECOC decoding to better evaluate the label assigned by the classifier. This ECOC-based reliability measure can be used to increase the confidence of the classifier's output on the inputs with high risk of mislabeling. The second part of the chapter extends the ECOC algorithm to multi-label problems. To validate the proposed ML-ECOC and compare its performance with the state-of-the-art methods, we apply the ML-ECOC on the real-world problem of multi-label text categorization.

**Keywords:** Classifier ensemble, random linear oracle, mixture of experts, error-correcting output codes, text categorization.

# Chapter 1

# Brief Overview of Multiple Classifier Systems

The ultimate goal in pattern recognition is to achieve the best possible classification performance for the task at hand. A promising approach towards this goal consists of combining classifiers, as monolithic classifiers are typically not able to properly handle the complexity of difficult problems. It has been proven that combining a set of independent classifiers with acceptable accuracy leads to better performance [28], provided that the *diversity* among *accurate* base classifiers in an ensemble system is enforced in some way. This chapter presents a brief review of Multiple Classifier Systems (MCSs), also known in the literature as *ensemble learning*, *committee machine*, or *combining classifiers*.

The main idea of MCSs is to employ ensemble of classifiers and combine them in various fashions. Theoretically speaking, if we have an ensemble of $M$ *independent* classifiers with uncorrelated error areas, the error of an overall classifier obtained by simply averaging/voting their output can be reduced by a factor of $M$. Unfortunately, the key assumption of independence is unrealistic; in practice, as typically errors are highly correlated, so that the factor of error reduction is usually much smaller than $M$. Many techniques have been proposed in the machine learning and pattern recognition communities to generate diverse classifiers. Among others, let us recall (i) bagging [12], (ii) boosting [61], (iii) random oracles [29], (iv) mixture of experts (ME) [45], and (v) ECOC [17]. This chapter gives a brief description of the cited methods, although in the following, we will concentrate only on random oracles, ME, and ECOC.

## 1.1 Bagging

*Arcing* (adaptive reweighting and combining) is a generic term that refers to reusing or selecting data in order to improve classification performance. Bootstrap AGGregation or Bagging [12] was proposed by Breiman in 1996 to improve the classification by combining classifications of randomly generated training sets. Bagging was the first effective method of ensemble learning and is one of the simplest methods of arcing. The meta-algorithm, which is a special case of model averaging, was originally designed for classification and is usually applied to decision tree models, but it can be used with any type of classification model. The

method uses multiple versions of a training set by using the bootstrap, i.e. sampling with replacement. Each of these data sets is used to train a different model, increasing diversity among individuals. The outputs of the individual models are combined by voting the individual outputs to create a final ensemble output.

Given a standard training set $D$, the bagging technique generates $m$ training sets $D_i$, $i = 1,2,\ldots,m$, by sampling examples from $D$ uniformly and with replacement. In so doing, it is likely that some samples will be repeated for any training set $D_i$. If $\mid D \mid = \mid D_i \mid$ and if the training set is large enough, the $D_i$ is expected to have about 63.2% of the unique examples of $D$, the rest being repeated. Samples with this characteristic are known as "bootstrap samples". Each $D_i$ is then used to train a base classifier. The output of the final ensemble is calculated by combining the individual's outputs (normally by using a voting strategy).

Experimental and theoretical results concerning the behavior of bagging allow to conclude that it is only effective when using unstable non-linear models[1] Since the method averages several individual base classifiers, it is not useful for improving linear models. Similarly, bagging is not superior to very stable models like *k-nearest neighbors* [58].

## 1.2 Boosting

Boosting [51] is a meta-learning algorithm which is based on the question posed by Kearns [27]: *"can a set of weak learners create a single strong learner?"*. A weak learner is defined to be a classifier which is only slightly better than random labeling. In contrast, a strong learner is a classifier that is arbitrarily well-correlated with the true classification. Boosting is the most widely used ensemble method and one of the most powerful learning ideas introduced in the last twenty years.

One first creates a *weak* classifier. Several models are built iteratively, adding a new classifier at each step. Each new classifier is trained on a data set in which samples misclassified by the previous model are given more weight while samples that are classified correctly are given less weight (some boosting algorithms actually decrease the weight of repeatedly misclassified samples, e.g., boost by majority and BrownBoost). Classifiers are weighted according to their accuracy and outputs are combined using a voting schema.

The original boosting algorithm has been proposed by Robert Schapire (a recursive majority gate formulation [51]) and Yoav Freund (boost by majority [16]) in 1990. They were not adaptive and could not take full advantage of weak learners. Many variations of the boosting algorithm have been proposed, typically focusing on a different way of weighting training data points and hypotheses. First of all, let us recall *AdaBoost* (standing for "adaptive boosting") [61] deserves a special citation, as it was the first algorithm that could adapt to weak learners. AdaBoost is also the most popular boosting algorithm. It uses the same training set over and over again (thus it does not need to be large) and can also combine an arbitrary number of base-learners. Other relevant proposals include LPBoost, TotalBoost, BrownBoost, MadaBoost, and LogitBoost.

[1] Unstable models are characterized by the fact that small changes in the training set can cause a significant change in the model.

## 1.3 Random Oracles

To design a classifier ensemble, two main alternative approaches have been proposed in the literature: classifier fusion and classifier selection. Base classifiers in fusion methods have the expertise to handle the whole classification boundary whereas in selection methods, each classifier is responsible for a specific part of the classification problem. Therefore, decision making in fusion methods uses the result of all base classifiers whereas in selection methods only one classifier is responsible for each test sample. In this case, the classifier responsible for classification is selected by an expert called "oracle".

Recently, Kuncheva and Rodriguez proposed another framework for ensemble making, which combines fusion and selection [29]. The idea is to replace each classifier in the ensemble with a *miniensemble* of two classifiers and an oracle, which embeds a linear hyperplane to decide which classifier in the miniensemble should process the current input. Hence, for each miniensemble, classification data are randomly split into two parts by a linear hyperplane, and then a classifier is assigned to each part. In the classification phase, for each miniensemble, the competent classifier is chosen by the oracle depending on which region of the input domain the current input belongs to. As highlighted by the authors, this approach encourages extra diversity in the ensemble while promoting high accuracy of the individual ensemble members.

In the second chapter of this thesis 2, we first introduce Random Prototype-based Oracle (RPO) for miniensembles of classifier selection. For each classification problem, RPO randomly splits the feature space into $N$ subspaces, expecting that each resulting subspace introduces easier boundaries. Then we extend the work done by Kuncheva and Rodriguez and use the RPO method in the fusion-selection framework to design classifier ensembles using different types of base classifiers. Considering the random nature of the RPO method and its strategy aimed at decomposing the given problem into an arbitrary large number of subproblems, base classifiers are expected to be more diverse while preserving high accuracy. As a result, a significant improvement in the overall accuracy has been achieved.

## 1.4 Mixture of Experts

Most real-world pattern recognition problems are too complicated for a single classifier to solve. Divide-and-conquer principle has proved to be efficient in many of the complex situations, using a combination of classifiers which have complementary properties. The issues are (i) how to divide the problem into simpler subproblems, (ii) how to assign base classifiers to solve these subproblems, and (iii) how to obtain the final decision using their outputs.

In some cases the problem can be decomposed manually. However, in most real-world problems, we either know too little about the problem, or it is difficult to achieve a clear understanding of how to manually decompose it into subproblems. Thus, a method for automatically decomposing a complex problem into a set of overlapping or disjoint subproblems is desirable, assigning one or more classifiers (experts hereinafter) to each subproblem.

Jacobs et al. [45] have proposed an ensemble method based on the divide-and-conquer principle called Mixture of Experts (ME), in which a set of networks referred to as *expert networks* is trained together with a *gating network*. This tight coupling mechanism (i) encourages diversity among experts by automatically localizing them in different regions of the input space and (ii) achieves good dynamic combination weights of the ensemble mem-

bers by concurrently training the gating network together with the experts. The Hierarchical Mixture of Experts (HME) [23] is a well-known tree-structured architecture, which can be thought of as a natural extension of the Mixture of Experts (ME) model. The expert networks form the leaves of the tree, whereas gating networks are located at the branch-points. Tasks are approached using a "recursive" divide-and-conquer strategy: complex tasks are decomposed into subtasks which in turn are themselves decomposed into sub-subtasks. Like many known classical artificial neural network ensemble methods, diversity in the standard HME is promoted by randomly initializing their weight parameters. This choice drives experts to start learning their task from different points in the search space, with the goal of getting them specialized on different subspaces.

Since Jacob's proposal in 1991, the ME model has been widely investigated. Many of the earlier works on the ME and HME models use preprocessing to partition or transform the input space into simpler and more separable spaces. An expert is then specialized on each subspace without altering the learning rules established by the standard ME model. As a consequence, the major effort in earlier works has been spent in the task of increasing the individual accuracy of experts, so to facilitate their task on the corresponding areas of expertise. Waterhouse and Cook [49] and Avnimelech and Intrator [41] proposed to combine ME with the boosting algorithm. Since boosting encourages classifiers to become experts on patterns that previous experts disagree on, it can be successfully used to split the data set into regions for the experts in the ME model, thus ensuring their localization and diversity. Tang et al. [9] tried to explicitly "localize" experts by applying a cluster-based preprocessing step, aimed at partitioning their input space. In particular, they used self-organizing maps (SOM) to partition the input space according to the underlying probability distribution of the data. As a result, better generalization ability, together with more stability in parameter setting, is achieved. However, as they argue at the end of the paper, the proposed method has been designed for (and validated on) only binary and low dimensional problems. Wan and Bone [13] used a mixture of radial basis function networks to partition the input space into statistically correlated regions and learn the local covariance model of the data in each region. Ebrahimpour et al. [43] proposed a view-independent face recognition system using ME by manual decomposition of the face view space into specific angles (views), an expert being specialized on each view. Nevertheless, the proposed method is only efficient in 2D face recognition and, as argued by the authors, extending this approach to other classification problems and applications could be challenging and not always possible.

It is worth pointing out that, in the original formulation of the ME model, parameters are determined by maximum likelihood, which is prone to severe overfitting, including singularities in the likelihood function. This can be particularly problematic in a complex model such as the HME, due to the relatively large number of parameters involved in defining the distributions for experts and gating networks. Indeed, there are many singularities in the likelihood function which arise whenever one of the mixture components *collapses* onto a single data point. Simultaneously training gating networks and experts in an HME architecture (with the goal of obtaining sufficiently accurate classifiers with relatively optimum parameters) continues to pose a research challenge.

## 1.5 ECOC

ECOC is a technique which manipulates output labels of the classes. ECOC achieved promising results particularly on the problems with large number of classes. In ECOC method, a discrete decomposition matrix (code matrix) is first defined for the multi-class problem at hand. Then this problem is decomposed into a number of binary subproblems, (dichotomies), according to content of the code matrix. After training binary classifiers on these dichotomies and testing them on any incoming test sample, a binary output vector is created. The final label is assigned to the class with the shortest distance (maximum similarity) between this vector and the codewords. In particular, given a classification problem with $N_c$ classes, the main idea of ECOC is to create a codeword for each class. The code matrix, say $M$, is obtained by arranging the codewords row-by-row. $M \in \{-1,0,+1\}^{N_c \times L}$ and $L$ is the code length. From a learning perspective, $M$ specifies $N_c$ classes to train $L$ classifiers (dichotomizers), $f_1 ... f_L$. A classifier $f_l$ is trained according to the column $M(.,l)$. If $M(N,l) = +1$ then all examples of class $N$ are positive, if $M(N,l) = -1$ then its all examples are negative and, finally, if $M(N,l) = 0$ none of the examples of class $N$ participate in the training of $f_l$. Let $\overline{y} = [y_1 ... y_L]$, $y_l \in \{-1,+1\}$ be the output vector of the $L$ classifiers in the ensemble for a given input $x$. In the decoding step, the class output that maximizes the similarity measure $s$ between $\overline{y}$ and the codeword $M(N,.)$ is selected. In symbols:

$$Class\ Label = ArgMax\ S(\overline{y}, M(N,.)) \tag{1.1}$$

As for the similarity measures, common techniques are (i) the Hamming decoding distances (Equation (1.2)), when classifier outputs are hard decisions, and Margin decoding (Equation (1.3)), when outputs are soft decisions. In symbols:

$$S_H(\overline{y}, M(N,.)) = 0.5 \times \sum_{l=1}^{L} 1 + y_l M(N,l) \tag{1.2}$$

$$S_M(\overline{y}, M(N,.)) = \sum_{l=1}^{L} y_l M(N,l) \tag{1.3}$$

The ECOC matrix codifies the class labels in order to achieve different partitions of classes, considered by each dichotomizer. The main coding strategies can be divided into problem-indepentent (or fixed) and problem-dependent.

*Problem-independent strategies.*
Most of the popular ECOC coding strategies up to now are based on pre-designed problem-independent codeword construction, which satisfy the requirement of high separability between rows and columns. These strategies include: *1vsA*, where each classifier is trained to discriminate a given class from the rest of classes using $N_c$ dichotomizers; *random techniques*, which can be divided into the *dense random strategy*, consisting of a binary matrix with high distance between rows with estimated length of $10 \log_2 N_c$ bits per code, and the *sparse random strategy* based on the ternary symbol and with the estimated optimal length of about $15 \log_2 N_c$. *1vs1* is one of the most well known coding strategies, with $N_c(N_c-1)/2$ dichotomizers including all combinations of pairs of classes [55]. Finally, BCH codes [48] are based on algebraic techniques from Galois Field theory, and while its implementation is fairly complex, it has some advantages such as generating ECOC codewords separated by a

minimum, configurable Hamming distance and good scalability to hundreds or thousands of categories.

All these codification strategies are defined independently of the data set and satisfy two properties:

- *Row separation.*
  In order to decrease misclassifications, the codewords should be as far apart from one another as possible. We can still recover the correct label for $x$ even if several classifiers fail. A measure of the error-correcting ability of any code is the minimum Hamming distance, $H_c$ , between any pair of codewords. The number of errors that the code is guaranteed to be able to correct is $[\frac{H_c-1}{2}]$.

- *Column separation.* It is important that the dichotomies given as the assignments to the ensemble members are as different from each other as possible. This will drive the ensemble towards low correlation between the classification errors (high diversity) which will hopefully increase the ensemble accuracy [17].

*Problem-dependent code matrices.*
All the coding strategies described above are fixed in the ECOC matrix design step, defined without considering the problem characteristics or the classification performance. Recently some researchers [14, 38, 37, 62, 21] argue that the selection and the number of dichotomizers must depend on the performance of the ensemble on the problem at hand.

The first approach to design problem-dependent ECOC has been proposed in [6], where the back-propagation algorithm is used to drive the codewords for each class. However, this method is only applicable when the base learner is a multi-layer perceptron. Utschick et al. [60] also tried to optimize a maximum-likelihood objective function by means of the expectation maximization (EM) algorithm in order to achieve optimal decomposition of the multi-class problem into two-class problems.

Crammer et al. [26] proved that the problem of finding the optimal matrix is computationally intractable since it is (Non-deterministic Polynomial) NP-complete. Furthermore, they introduce the notion of continuous codes and cast the design problem of continuous codes as a constrained optimization problem.

Recently, Zhou et al. [62] proposed a method called Data-driven ECOC (DECOC) to explore the distribution of data classes and optimize both the decomposition process and the number of base learners. The key idea of DECOC is to selectively include some of the binary learners into the predefined initial codematrix based on a confidence score defined for each learner. The confidence score for each column is computed by measuring the separability of the corresponding binary problem. This measure is used to determine how likely a learner will be included in the ensemble. The method needs to search the output label space and ensure the validity of each candidate. Therefore, the efficiency of the method on problems with larger number of classes is limited.

The Discriminant ECOC [37] renders each column of the code matrix to the problem of finding the binary partition that divides the whole set of classes so that the discriminability between both sets is maximum. The criterion used for achieving this goal is based on the mutual information between the feature indexes and class labels. Since the problem is defined as a discrete optimization process, the Discriminant ECOC uses the floating search method as a suboptimal search procedure for finding the partition that maximizes the mutual information. The whole ECOC matrix is created with the aid of an intermediate step

formulated as a binary tree. Considering all the classes of a problem, a binary tree is built beginning from the root as follows: each node corresponds to the best bi-partition of the set of classes maximizing the quadratic mutual information between the class samples and their labels. The process is recursively applied until sets of single classes corresponding to the tree leaves are obtained. This procedure, ensure decomposition of the multi-class problem into $N_c - 1$ binary subproblems.

Forest ECOC [14] is an extension of Discriminant ECOC. It takes advantage of the tree structure representation of the ECOC method to introduce a multiple-tree structured called "Forest" ECOC. This method is based on embedding different optimal trees in the ECOC approach to obtain the necessary number of classifiers able to ensure the required classification performance.

ECOC Optimizing Node Embedding (ONE) [38] presents an approach that improves the performance of any initial code matrix by extending it in a sub-optimal way. ECOC-ONE creates the new dichotomizers by minimizing the confusion matrix among classes guided by a validation subset. As a result, overfitting is avoided and relatively small codes with good generalization performance are obtained.

Finally, Hatami [21] proposes a heuristic method for application-dependent design of optimal ECOC matrix based on a thinning algorithm. The main idea of the proposed Thinned-ECOC method is to successively remove some redundant and unnecessary columns of any initial codematrix based on a metric defined for each column. As a result, the computational cost for training the ensemble is reduced while preserving accuracy.

## 1.6 Summary of the Selected MCSs

Different classifier ensemble approaches can be compared using various characteristics. Some combination schemes are adaptive in the sense that the combiner evaluates (or weighs) the decisions of individual classifiers depending on the input pattern. In contrast, nonadaptive combiners treat all the input patterns the same. Adaptive combination schemes can further exploit the detailed error characteristics and expertise of individual classifiers. Furthermore, different combiners expect different types of output from individual classifiers. Xu et al. [30] grouped these expectations into three levels: 1) measurement (or confidence), 2) rank, and 3) abstract. At the confidence level, a classifier outputs a numerical value for each class indicating the belief or probability that the given input pattern belongs to that class. At the rank level, a classifier assigns a rank to each class with the highest rank being the first choice. Rank value cannot be used in isolation because the highest rank does not necessarily mean a high confidence in the classification. At the abstract level, a classifier only outputs a unique class label or several class labels (in which case, the classes are equally good). The confidence level conveys the richest information, while the abstract level contains the least amount of information about the decision being made.

Tables 1.1 and 1.2 compare above mentioned classifier ensemble techniques from different perspectives. Table 1.1 reports features such as ensemble architecture, dynamic or static behaviours and the level of fusion, whereas Table 1.2 discusses the main advantages and disadvantages of the ensembles.

Table 1.1: The main characteristics of the selected MCSs.

| Technique | Architecture | Dynamic | Info-level |
|---|---|---|---|
| Bagging | Parallel | No | Confidence |
| Boosting | Parallel (Serial training) | No | Abstract |
| Rand. Subspace | Parallel | No | Confidence |
| Rand. Oracle | Parallel | YES | Confidence |
| ME | Gated parallel | YES | Confidence |
| HME | Gated parallel hierarchical | YES | Confidence |
| ECOC | Parallel | No | Confidence |

Table 1.2: Pros and cons of the selected MCSs.

| Technique | Pros | Cons |
|---|---|---|
| Bagging | Stable against noise | Needs many comparable classifiers |
| Boosting | Improves margins; unlikely to overtrain | Unstable against noise |
| Rand. Subspace | Effective for high dimensional data | Needs many comparable classifiers |
| Rand. Oracle | Explores local expertise | Over division of data |
| ME | Explores local expertise | Difficulty of joint optimization |
| HME | Explores local expertise | Difficulty of joint optimization |
| ECOC | Handles large number of classes | Defining a good code matrix |

# Chapter 2

# Random Prototype-based Oracle

Classifier ensembles based on selection-fusion strategy have recently aroused enormous interest. The main idea underlying this strategy is to use miniensembles instead of monolithic base classifiers in an ensemble with the goal of improving the overall performance. This chapter proposes a classifier selection method to be used in selection-fusion strategies. The method requires to split the given classification problem according to some prototypes randomly selected from training data and to build a classifier on each subset. The trained classifiers, together with an oracle used to switch among them, form a miniensemble of classifier selection. With respect to other methods used in the selection-fusion framework, the proposed method has proven to be more efficient in the decomposition process with no limitation in the number of resulting partitions. Experimental results on selected datasets from the UCI repository show its validity.

The rest of this chapter is organized as follows: in Section 2, we introduce the Random Prototype-based Oracle (RPO) as a novel classifier selection miniensemble method. Then we use RPO for designing classifier ensembles. Experimental results are reported and discussed in Section 3.

## 2.1 RPO miniensemble

The term "miniensemble of classifiers" (or simply miniensemble), used by Kuncheva and Rodriguez for the first time, refers to the possibility of substituting a base classifier in an ensemble with another ensemble, giving rise to a two-tiered ensemble architecture. The miniensemble is expected to have approximately the same computational cost of a base classifier while significantly improving the ensemble accuracy. In particular, Kuncheva and Rodriguez proposed a miniensemble of classifier selection method called Random Linear Oracle (RLO) [29]. An RLO consists of a pair of classifiers and a fixed, randomly created, linear oracle. Each of these two classifiers learns on a different subspace (identified by the oracle) and the oracle is also entrusted with deciding which classifier must be activated depending on the current input to be classified. Figure 2.1 shows how this method works when applied to a 2-class toy problem. It is worth pointing out that RLO is more useful for classifiers not expressive enough to provide a good approximation of their class boundary. In fact, these classifiers can not learn class boundary perfectly and decomposition of boundaries

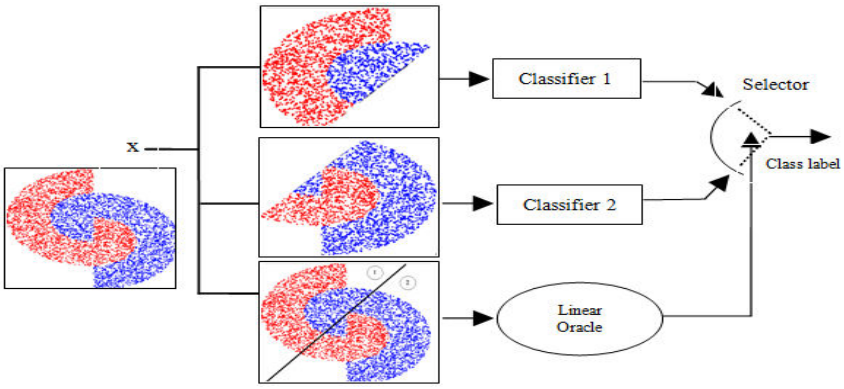


Figure 2.1: The RLO method applied to a 2-class toy problem.

introduces new, possibly easy to learn, problems [46, 5].

The Random Oracle method opened a very promising research field [29], [46], [5]. The RPO introduced in this chapter has been devised within this novel research field.

For some weak learning algorithms, or in the event that classification boundaries are too complex, decomposing the original problem into only two partitions (using a linear hyperplane) may not be enough to deal with the underlying complexity. A viable solution in these cases can be to further decompose the problem, thus increasing the chances of introducing simpler subproblems. This is the underlying idea in which the RPO algorithm has been framed. Algorithm 1 and figure 2.2 show the algorithmic view and a block diagram representation of the proposed method, respectively.

**Algorithm 1** The generic algorithm for classifier selection based on RPO.

TRAINING:

1. choose randomly $N$ prototype samples from the training set
2. assign each remaining sample in the training set to the nearest prototypes and split the data into $N$ subsets
3. build a classifier on each subset

TESTING:

1. for any incoming sample, ask the oracle to select the nearest prototype
2. give authority to the classifier responsible for this prototype to assign the class label

The main advantage of the proposed algorithm with respect to RLO is its flexibility in decomposing the problem at hand. While RLO is able to partition the problem into only two subproblems by means of a linear function ($N = 2$), RPO is able to extend this ability to $N \geq 2$ parts. Hence, for more complex classification boundaries, RPO is expected to perform better than RLO. Figure 2.3 compares the performance of these two methods on a 2-class toy problem. As clearly shown, the classification boundaries introduced by RPO are easier to

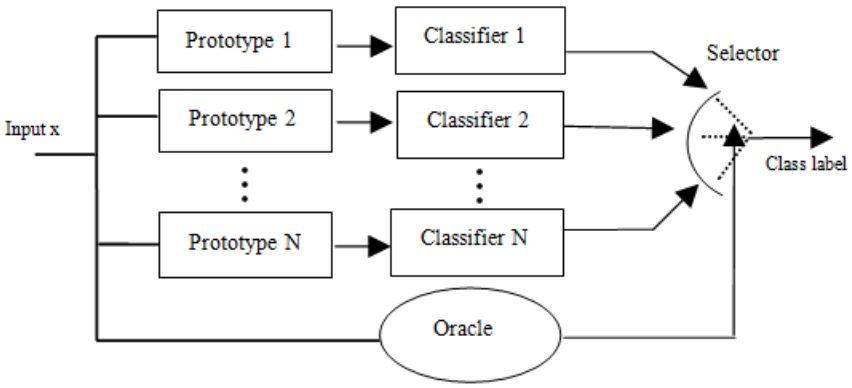


Figure 2.2: Block diagram of classifier selection based on RPO.

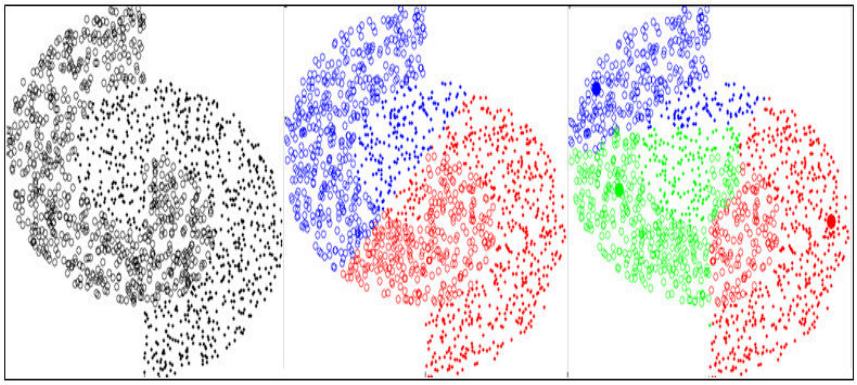

Figure 2.3: Decomposition of a 2-class toy problem, RLO vs. RPO.

handle by any learning algorithm. 2.3 Left: original toy problem, 2.3 Middle: splitting with a random linear function, 2.3 Right: splitting with random prototypes based on the nearest distance measure. Selected prototypes ($N = 3$) from training data are represented by bold points.

The parameter $N$ in the RPO algorithm is the number of random prototypes needed to decompose the original problem. This parameter depends on the characteristic of the dataset, on the complexity of decision boundaries, on the number of training samples, and on the type of base classifiers. In particular, in the event that a weak learning algorithm is used to implement the base classifiers in the RPO algorithm and that the problem has complex boundaries, $N$ should be large.

The idea of applying RPO in a fusion-selection scheme consists of replacing each base classifier in an ensemble with a miniensemble of $N$ classifiers and an oracle. At the testing phase, the labels issued by the miniensembles are combined throughout the ensemble combination rule.

RPO addresses both accuracy and diversity issues in ensemble making. In particular, accuracy is improved by dividing the original classification problem into $N$ subproblems and specializing each subclassifier on each part (divide and conquer principle), whereas diversity is improved by including randomness in the decomposition process. For each miniensemble, RPO randomly selects different prototypes to partition the problem. As a result, subclassifiers are created on different training samples –thus enforcing diversity.

Table 2.1: The main characteristics of the selected UCI datasets.

| Problem | # Train | # Test | # Attributes | # Classes |
|---|---|---|---|---|
| Glass | 214 | - | 9 | 7 |
| Iris | 150 | - | 4 | 3 |
| Letter | 20000 | - | 16 | 26 |
| Pendigits | 7494 | 3498 | 16 | 10 |
| Satimage | 4435 | 2000 | 36 | 6 |
| Segment | 210 | 2100 | 19 | 7 |
| Vowel | 990 | - | 11 | 11 |
| Yeast | 1484 | - | 8 | 10 |

Table 2.2: Accuracy of RPO vs. RLO, using different learning algorithms.

| | Glass | Iris | Letter | Pendigits | Satimage | Segment | Vowel | Yeast |
|---|---|---|---|---|---|---|---|---|
| RPO-MLP | **76.7** | 93.3 | 75.7 | **93.2** | 83.3 | 83.1 | 62.0 | 54.2 |
| RPO-SVM | 74.4 | **94.1** | **76.9** | 92.7 | **84.9** | **84.4** | **62.9** | **55.5** |
| RPO-FLDA | 74.5 | 91.9 | 73.4 | 90.9 | 80.7 | 80.9 | 60.5 | 53.2 |
| RLO-MLP | 72.2 | 90.3 | 74.4 | 90.9 | 80.0 | 81.1 | 60.9 | 52.0 |
| RLO-SVM | 71.0 | 89.9 | 71.9 | 91.0 | 80.3 | 82.6 | 60.7 | 51.0 |
| RLO-FLDA | 70.8 | 87.5 | 70.1 | 90.1 | 77.9 | 78.0 | 58.8 | 51.1 |
| Single-MLP | 69.1 | 85.7 | 70.0 | 88.1 | 76.0 | 77.8 | 58.1 | 47.9 |
| Single-SVM | 68.8 | 84.4 | 70.7 | 86.9 | 76.2 | 77.3 | 55.9 | 48.0 |
| Single-FLDA | 66.6 | 83.9 | 69.5 | 85.1 | 74.0 | 76.4 | 55.5 | 47.3 |

## 2.2 Experimental results

We used some of the UCI machine learning data sets [40] to check the validity of the proposed method. These data sets include real-world and synthetic problems, with variable characteristics, previously investigated by other researchers. Table 2.1 shows the selected datasets in more detail.

To determine the best value for N, we varied it from 2 to 10 for each dataset. We also used 10-fold cross validation to ensure statistical significance while evaluating the accuracy of classifiers. Three different types of algorithms, multi-layer perceptron (MLP) with back-propagation learning rule, Support Vector machine (SVM) with linear kernels, and Fisher Linear Discriminant Analysis (FLDA), have been tested to create base classifiers in ensembles, also to study whether the proposed method is independent from the learning algorithm.

Our first goal was to show the robustness of RPO with respect to RLO. Table 2.2 compares the performances of RPO against RLO. The Performance of single base classifiers is also presented here as a base line. As clearly shown, RPO outperforms RLO regardless of the type of algorithms used to train base classifiers, and both are more accurate than strategies that rely on single base classifier.

Table 2.3: Average improvement in the accuracy of standard ensembles using RPO.

| | Glass | Iris | Letter | Pendigits | Satimage | Segment | Vowel | Yeast |
|---|---|---|---|---|---|---|---|---|
| Adaboost | +0.3 | +2.2 | +0.9 | 0.0 | +0.8 | +0.2 | +2.0 | +1.3 |
| Bagging | +2.5 | +2.7 | +1.8 | +1.1 | +2.2 | +2.9 | +3.4 | +3.0 |
| Rand. subspace | +2.1 | +3.1 | +2.7 | +2.2 | +3.0 | +1.4 | +3.3 | +2.8 |
| ECOC-1vsA | +1.1 | +2.7 | +0.9 | +1.1 | +0.5 | +0.8 | +2.0 | +2.7 |
| ECOC-1vs1 | +0.7 | +1.1 | +0.6 | +0.2 | +1.1 | 0.0 | +1.8 | +3.4 |
| ECOC-dense | +1.2 | +2.6 | +1.1 | +0.5 | +1.4 | +2.0 | +2.1 | 0.0 |
| Majority Vote | +2.0 | +2.0 | +1.5 | +1.9 | +1.6 | +2.3 | +2.8 | +2.1 |

Table 2.3 shows that RPO improves the accuracy of standard ensemble methods. In this experiment, the RPO algorithm has been applied to create Random Subspaces [22], Adaboost [61], Bagging [12] and Error-Correcting Output Codes (ECOC) [17]. 1vs1, 1vsA and dense random are different types of code matrices used in the ECOC ensemble strategy. As shown, RPO improves the performance of standard ensemble strategies in almost all cases.

# Chapter 3

# Mixture of Random Prototype-based Experts

The Mixture of Experts (ME) is one of the most popular ensemble methods used in pattern recognition and machine learning. This algorithm stochastically partitions the input space of a problem into a number of subspaces, experts becoming specialized on each subspace. To manage this process, the ME uses an expert called gating network, which is trained together with the other experts.

In this chapter, we propose a modified version of the ME algorithm which first partitions the original problem into centralized regions and then uses a simple distance-based gating function to specialize the expert networks. Each expert contributes to classify an input sample according to the distance between the input and a prototype embedded by the expert. The Hierarchical Mixture of Experts (HME) is a tree-structured architecture which can be considered a natural extension of the ME model. The training and testing strategies of the standard HME model are also modified, based on the same insight applied to standard ME. In both cases, the proposed approach does not require to train the gating networks, as they are implemented with simple distance-based rules. In so doing the overall time required for training a modified ME/HME system is considerably lower. Moreover, centralizing input subspaces and adopting a random strategy for selecting prototypes permits to increase at the same time individual accuracy and diversity of ME/HME modules, which in turn increases the accuracy of the overall ensemble.

Despite many studies on the theory and application of the ME model, to our knowledge, its training, testing, and evaluation costs have not been investigated yet. After analyzing the ME model in terms of number of required floating point operations, this chapter makes an experimental comparison between the ME model and the proposed Mixture of Random Prototype Experts.

Experimental results on a binary toy problem and on selected datasets from the UCI machine learning repository show the robustness of the proposed methods compared to the standard ME/HME models.

The rest of the chapter is organized as follows: in Section 2, we briefly recall the standard ME model and its hierarchical counterpart. Section 3 first introduces the concept of random prototype-based splitting and then describes the proposed mixture of random prototype-based experts. An extension of the model for hierarchical settings is also proposed. Section

4 reports and discusses experimental results.

## 3.1 Standard Mixture of Experts Models

### 3.1.1 Standard ME Model

The adaptive mixture of local experts [45] is a learning procedure which achieves improved generalization performance by assigning different subtasks to different experts. Its basic idea consists of concurrently training several experts and a gating network. The gating function assigns a "probability" to each expert based on the current input. In the training phase, this value denotes the probability for a pattern to appear in the training set of an expert. In the test step, it defines the relative contribution of each expert to the ensemble. The training step attempts to achieve two goals: (i) for a given expert, find the optimal gating function; (ii) for a given gating function (network), train each expert to achieve maximal performance on the distribution assigned to it by the gating function. Accordingly, the accuracy of an ME classifier is affected by the performance of both expert networks and gating network. Resulting misclassifications in this model derive from two sources: (a) the gating network is unable to correctly estimate the probability for a given input sample and (b) local experts do not learn their subtask perfectly. Let us consider the network shown in Figure 3.1, which represents an ME model with $N = 3$ experts. The $i$-th expert produces its output $o_i(x)$ as a generalized linear function of the input $x$:

$$o_i(W_i, x) = f(W_i \cdot x) \tag{3.1}$$

where $W_i$ is the weight matrix of the $i$-th expert and $f(.)$ is a predefined continuous nonlinearity. The gating network is also a generalized linear function, and its $i$-th output, $g_i(V_i, x)$, is the multinomial logit (aka softmax) function of the gating network's output, $o_{g_i}$.

$$g_i(V_i, x) = \frac{exp(o_{g_i})}{\sum_{j=1}^{N} o_{g_i}} \quad i = 1, ..., N \tag{3.2}$$

where $V_i$ is the weight vector of the gating network. Hence, the overall output of the ME architecture, $o(x)$, is

$$o(x) = \sum_i g_i(V_i, x) \cdot o_i(W_i, x) \tag{3.3}$$

Two training procedures have been suggested in the literature [45, 23] for finding optimal weight parameters $W_i$ and $V_i$: (i) the standard error back-propagation algorithm with gradient descent and (ii) the Expectation-Maximization (EM) method.

### 3.1.2 Standard HME Model

The HME architecture 3.2 is a tree in which the gating networks lie at the nonterminal nodes and the expert networks lie at the leaves of the tree. Hence, it can be considered an ensemble of ME modules (as shown by dashed boxes). The task of each expert is to approximate a function over a region of the input space. Given a sample, the task of the gating network is to assign the weights to each expert. Figure 3.2 illustrates a mixture of four experts. In accordance with the typical terminology used for describing HME architectures: $\bar{x}$ is the

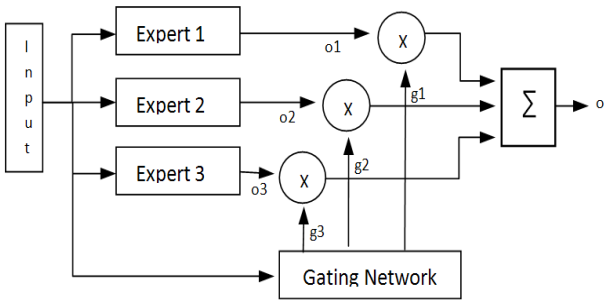


Figure 3.1: Block diagram representing the mixture of experts (ME) model (N=3).

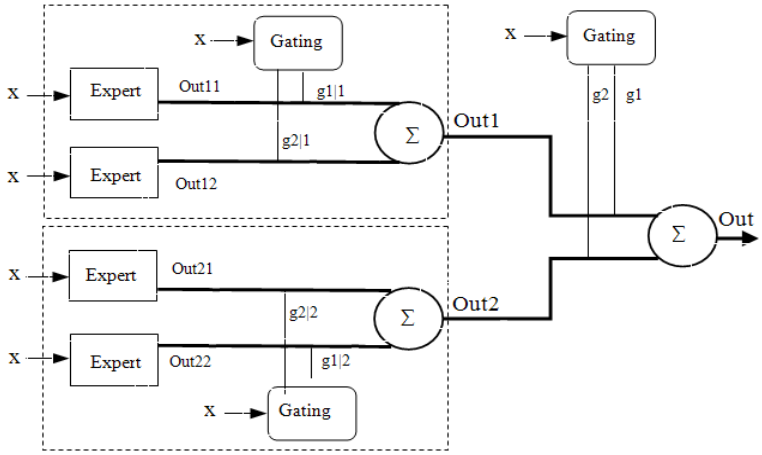


Figure 3.2: Block diagram representing a two-layer HME, with two ME modules.

input vector, $o_{ij}(\bar{x})$ is the output (expected value) of the $ij$-th expert, $g_i(\bar{x})$ is the output of the top gating network, denoting the prior probability for the pattern to be generated by the left or right branch of the root, and $g_{j|i}(\bar{x})$ is the output of the $i$-th bottom gating network, denoting the prior probability that the pattern is generated by the $ij$-th expert. In addition, $t$ is the target (desired output) and $P_{ij}(t|\bar{x})$ is the probability associated with the $ij$-th expert.

Assuming that experts are mutually exclusive, the overall probability, $P(t|\bar{x})$ and the expected value at the network output, $o(\bar{x})$, are given by:

$$P(t|\bar{x}) = \sum_i g_i(\bar{x}) \cdot \sum_j g_{j|i}(\bar{x}) \cdot P_{ij}(t|\bar{x}) \tag{3.4}$$

$$o(\bar{x}) = \sum_i g_i(\bar{x}) \cdot \sum_j g_{j|i}(\bar{x}) \cdot o_{ij}(\bar{x}) \tag{3.5}$$

Note that the notations defined for the two-level depth tree shown in Figure 3.2 can be easily extended to larger HME networks with a binary tree architecture.

Two training procedures are suggested in the literature [9, 13, 43] for finding optimal weight parameters of the HME architecture. Again, the first is the standard error back-propagation algorithm with gradient descent and the second procedure is based on the Expectation-Maximization (EM) method.

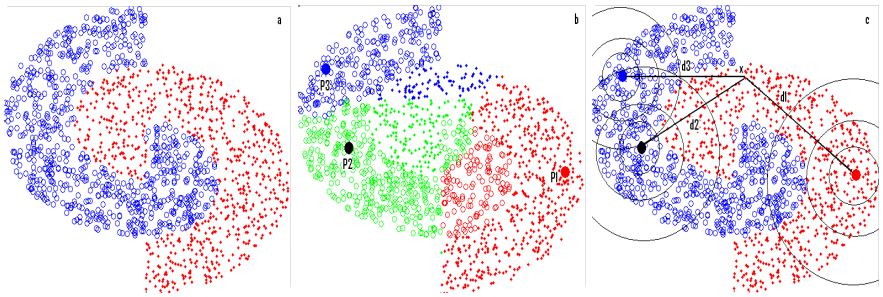


Figure 3.3: a) original problem, b) disjoint partitioning c) overlapping partitioning.

## 3.2 Mixture of Random Prototype-based Experts (MRPE) and Hierarchical MRPE

### 3.2.1 Mixture of Random Prototype-based Local Experts

In this section, we illustrate the proposed mixture of random prototype-based experts with more detail. The key underlying idea is to randomly partition the input space of the problem into subspaces and then specialize each expert on each subspace by means of "soft" competitive learning. First of all, the input space is partitioned according to some prototypes randomly chosen from the training set, so that the input samples are weighted during the training and testing phases based on their distances from the selected prototypes. The main advantage of this method is that, instead of a complex gating network which must be trained concurrently with other experts, the generated gating function has no parameters (weights) to adjust –as it simply enforces a distance-based weighting policy. This modification improves three important aspects of the standard ME model. First, it reduces the training time by decreasing the number of parameters to be estimated. Secondly, as simple distance measures used by the gating function are more robust with respect to errors in determining the area of expertise of an expert, errors in the proposed ME model are mainly limited to the error made by the expert networks, thus improving the overall accuracy of the classifier. Lastly, the region of expertise for each expert in the standard ME model is nested, which makes the problem difficult to learn. In the proposed method, each expert's area of expertise is more centralized, which makes the subproblem easier to learn. The latter property also makes the rules embedded by an expert easy to analyze, which is vital in some applications that need to make explicit the information about the area of expertise of each expert.

For the sake of simplicity and ease of comprehension, we describe this approach for the synthetic two-class problem shown in Figure 3.3.a. We used two different partitioning methods, i.e. disjoint and overlapping, shown in Figure 3.3.b and Figure 3.3.c, respectively. In case of disjoint partitioning, we first measure the distance between each training sample and the prototypes, and then assign a fixed value, $\eta_j$ to the $h_i$ of the expert proportional to these distances. $h_i$ is an estimate of the "a posteriori" probability for the $i$-th expert to generate the desired output $o$ and used as the coefficient of the learning rate for updating the weight parameters of the expert (static strategy). This implies that the weight update on the expert network whose prototype is nearest to the current input sample will be stronger than those performed on the others (the closer the expert the stronger the update is). Similarly, in the testing phase, the expert whose prototype is nearest to the input sample will contribute to a greater extent to the final output.

**Algorithm 2** Mixture of Random Prototype-based Experts

**PARAMETERS**:

- $strategy = \{static, dynamic\}$
- $N$ number of experts in an ME classifier
- $E = \{\eta_j \in (0,1) \mid j = 1..N\}$ such that: $\eta_k \leq \eta_{k+1}; k = 1..N-1$ and $|E| = \sum_j \eta_j = 1$

**WITH**:

- $\Psi = \{\epsilon_i \mid i = 1..N\}$ set of experts
- $P = \{p_i \in LS \mid i = 1..N\}$ set of randomly chosen prototypes, each assigned to an expert
- $\mathscr{X}_t, \mathscr{T}_t$ training set and $\mathscr{X}_e, \mathscr{T}_e$ testing set

**TRAINING**:

For $x \in \mathscr{X}_t$ Do:

- $D(x) = \{d_i(x) \mid i = 1..N\}$ where
  $d_i(x) = \|x - p_i\|$
- $H(x) = \{h_i(x) \mid i = 1..N\}$ where
  $h_i(x)$ represents the expected capability of $\epsilon_i$ to deal with the given input $x$
  $[strategy = static] : h_i(x) = \eta_r, \quad r = Rank(\epsilon_i, D(x))^*$
  $[strategy = dynamic] : h_i(x) = 1 - \frac{d_i}{\|D(x)\|}, \quad \|D(x)\| = \sum_j d_j(x)$
- update each expert $\epsilon_i$ $(i = 1..N)$ according to the standard learning rule for ME

**TESTING**:

Given an $x \in \mathscr{X}_e$ Do:

- $D(x) = \{d_i(x) \mid i = 1..N\}$
- $G(x) = \{g_i(x) \mid i = 1..N\}$ where
  $[strategy = static] : g_i(x) = \eta_r, \quad r = Rank(\epsilon_i, D(x))^*$
  $[strategy = dynamic] : g_i(x) = 1 - \frac{d_i}{\|D(x)\|}, \quad \|D(x)\| = \sum_j d_j(x)$
- calculate the overall output:
  $o_j(x) = \sum_i^N g_i(x) \cdot o(x, W_i)$
- select the class label $c_k$ such that
  $k = argmax_j\ (o_j(x))$

* $r = Rank(\epsilon_i, D(x))$ returns the rank of expert $\epsilon_i$ (i.e. a number in [1,N]) according to the distance $D(x)$ evaluated on the input $x$ (the lower the distance, the highest the ranking).

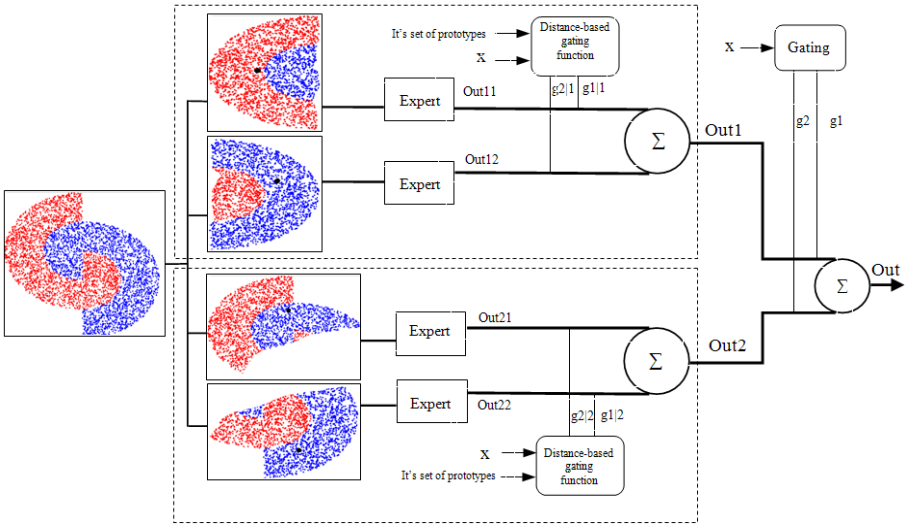


Figure 3.4: Block diagram representation of the proposed HMRPE model.

Unlike disjoint partitioning, where the learning rate coefficients are fixed for each partition and change sharply from one to another, in the overlapping method they change smoothly, proportional to the distances (dynamic strategy). Similarly, the amount of $d_i$ for the $i$-th expert depends on how close the prototype is to the current input sample $x$. In other words, for disjoint learning, the amount of expertise and contribution of experts is fixed for each partition, whereas, for overlapping learning, their expertise smoothly vary with the distance $d_i$ from the prototypes embedded in the experts. It is worth pointing out that the proposed method is general enough to be applied for building ME classifiers using both standard error back-propagation and EM learning rules. Algorithm 2 reports the procedure to be used for training and testing a mixture of random prototype-based experts, using both disjoint and overlapping partitioning rules for any chosen learning method.

### 3.2.2 Hierarchical MRPE Model

This section presents the proposed hierarchical mixture of random prototype-based experts (HMRPE) with more detail. The key underlying idea is to randomly partition the input space of the problem into subspaces and then pushing each expert to specialize on its subspace by means of "soft" competitive learning.

**RP-based splitting for HME** – For each ME module of an HME architecture, the input space is partitioned according to some prototypes randomly chosen from the training set. Let us note that the learning rules of the first-layer gating networks (gating of the ME modules) change with respect to the standard HME model, whereas the gating networks of the other layers (second, third, and so on) do not.

**Why does HMRPE work?** – Notwithstanding the amount of empirical studies proving that diversity and individual accuracy of ensemble members are two primary factors that affect the overall classification accuracy, theoretical studies clearly show that they are not independent [24]. Hence, the success of the proposed HMRPE approach can be attributed to three factors as follows:

1. Splitting the input space into $N$ centralized parts makes the subproblems easier to

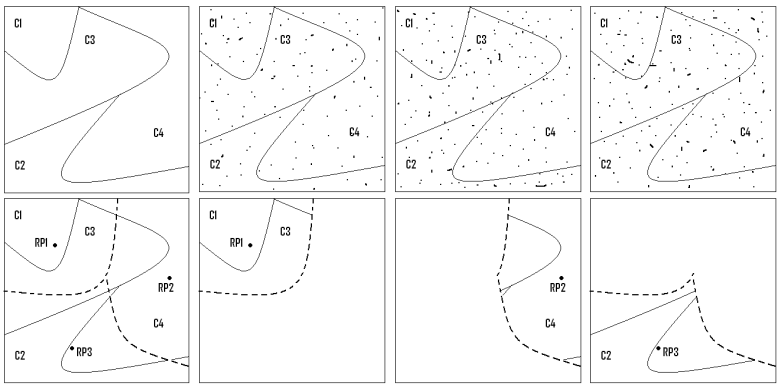


Figure 3.5: Input space partitioning performed by the HME (top) and the HMRPE (bottom)

learn for the expert network. As a consequence, the individual accuracy of the ensemble members is expected to be better than, or at least not worse than, the one exhibited by sets of experts specialized over the nested and stochastic subspaces. It is worth noting that, although expected, higher individual accuracy is not guaranteed by any means, since it depends on the complexity of classification boundaries, on the adopted learning algorithm, as well as on the position of the selected prototypes. 3.5 compares the regions of expertise of an ME module, embedded in both the standard HME and the HMRPE models, on a four-class toy problem. The top left figure shows the original problem, and the next three figures report the nested areas of expertise for the three experts in the standard ME module. The bottom left figure shows how the problem is partitioned using three random prototypes, and the next three figures highlight the centralized areas of expertise of three experts in the proposed HMRPE module.

2. Since each ME module embedded in the HMRPE architecture has its own set of prototypes (which are different from those embedded by the other ME modules), experts are specialized on very different data subsets, thus enforcing diversity.

3. The accuracy of an HME classifier is affected by the performance of both experts and gating networks. Accordingly, resulting misclassifications in this model derive from two sources: (a) the gating networks are unable to correctly estimate the probability for a given input sample and (b) local experts do not learn their subtask perfectly. Since simple distance rules used by the gating function are more robust with respect to errors in determining the area of expertise of an expert, errors in the proposed HMRPE model are mainly limited to the error made by the expert networks, thus improving the overall accuracy of the classifier.

## 3.3 Run-time Performance of ME and MRPE

In pattern recognition, the accuracy of classifier systems is usually the main concern. However, in real applications, their run-time performance may play an important role as well. In practice, many well-performing classifiers cannot be used in real applications due to the

amount of computational resources required for training, testing, or online evaluation.[1]

In computer science, the worst-case time complexity of algorithms is typically evaluated in terms of asymptotic behavior, denoted by the Landau symbol O (also called "big-O" notation) [42]. For practical applications, the asymptotic behavior is not informative enough, as the order of magnitude expressed in terms of well-known functions (e.g., logarithmic, polinomial, and exponential) hides details that can make the difference in terms of elapsed time for a run. For this reason, we prefer to carry out our analysis in terms of number of floating point operations (FLOPs). This is not an ideal measure [25]; however, it is useful for comparing the performance of a classifier system, as it focuses on the number of additions and multiplications required to perform a given task. In modern computer systems with floating point units (FPUs), addition and multiplication are comparable in complexity. Division has a similar complexity and function evaluations (e.g. $exp(x), x^p$) have roughly 2.5 times the complexity of an addition.

The expected run-time performance of any learned classifier system, including those compliant with the ME model, requires different formulations, depending on which specific phase (i.e., training or testing/online evaluation) is considered. As training is usually performed once (or from time to time in the event that the underlying process is time-variant), in the following we will concentrate on testing/online evaluation.

### 3.3.1 Run-time Performance of the ME Model

In general, this run-time performance depends on the type of classifier used, on its parameters, and on the characteristics of the problem to be solved (e.g., the number of samples used for training $n$, the number of features (dimensions) in each sample $d$, and the number of classes to be distinguished $c$).

As reported in Figure 3.6, the run-time performance of an ME classifier system can be decomposed as three main components: 1) expert networks, 2) gating network, and 3) aggregation (also called combination). Hence, the overall testing/online evaluation performance of an ME classifier equipped with $N_e$ experts is:

$$T_{ME} = \sum_{e=1}^{N_e} T_e + T_g + T_a \tag{3.6}$$

1. *Run-time performance of an expert ($T_e$):* Let us consider an expert with $d$ nodes in input, $H_e$ nodes in the hidden layer and $c$ nodes in the output layer. According to these hypotheses, between the input and the hidden layer, and between the hidden and the output layer, there are $W_{i1} = d \times H_e$ and $W_{i2} = H_e \times c$ connections, which yield to $d \times H_e$ and $H_e \times c$ multiplications, $d \times H_e$ and $H_e \times c$ additions, respectively. Considering $H_e + c$ additions of bias terms and $H_e + c$ evaluations of the unit transfer functions in the output nodes (e.g., sigmoids), the total complexity of an individual expert implemented by a Multi-Layer Perceptron (MLP) and equipped with a single hidden layer is:

$$T_e = 2H_e(d + c) + 3.5(H_e + c) \tag{3.7}$$

[1] We assume that a classifier system behaves in the same manner when testing and online evaluation is performed. Indeed, the difference holds only at an abstract level (i.e., it is related with the interpretation of such phenomena) rather than at a concrete level. For this reason, in the following, testing and online evaluation will be considered in the same way.

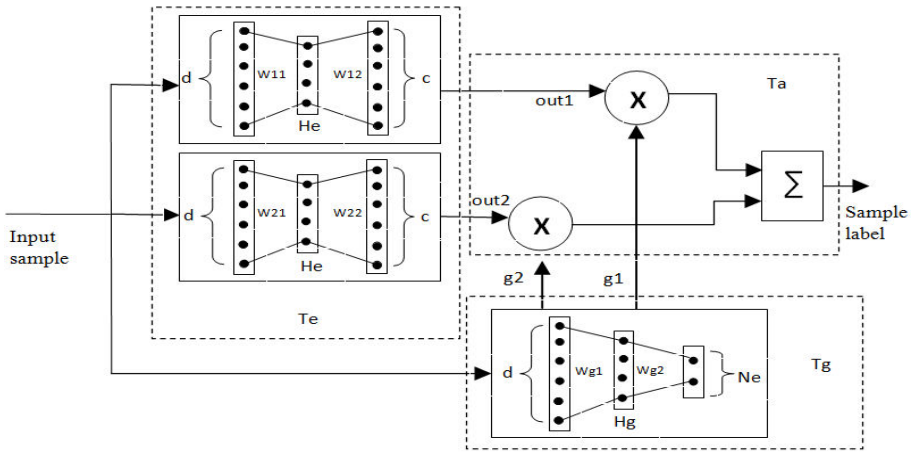


Figure 3.6: Total run-time of an ME classifier has 3 components: $T_e$),$T_g$ and$T_a$.

2. *Run-time performance of the gating network ($T_g$):* In the standard ME model, the technology used for implementing the gating network is typically the same of that used for implementing base classifiers. Hence, let us assume that gating networks are also MLPs. However, parameters of the gating function, such as the number of nodes in the hidden and in the output layer, are usually different, as the task of the gating function is different form the one performed by experts. For instance, an implementation of the ME model equipped with $N_e = 3$ experts and applied to a 10-class problem, the output layer of each expert has 10 neurons while the output layer of the gate has 3 nodes. $H_g$ being the number of hidden nodes, the complexity of the gating network formulates as follows:

$$T_g = 2H_g(d + N_e) + 3.5(H_g + N_e) \tag{3.8}$$

where $N_e$ is the number of individual experts contributing in the ME classifier.

3. *Run-time performance of aggregation ($T_a$):* In the aggregation step, the final label for each incoming sample is obtained by summing up, suitably weighted, the outputs of the experts (see Figure 3.6). As the weighted output of an expert has $c$ multiplications, $T_a$ furmulates as follows:

$$T_a = c \cdot (N_e + 1) \tag{3.9}$$

### 3.3.2 Run-time Performance of the MRPE Model

The experts and the aggregation rule in the MRPE model do not change with respect to the standard ME model. Hence, we only need to reformulate the expected run-time performance for the gating network. Indeed, the main contribution of the MRPE is its simple gate structure which makes it less complex *and* more accurate. The distance-based gating function does not need any training process, as it assigns learning rates and weights to an expert only based on the distance between the incoming samples and the random prototypes. The complexity of this distance measure $D(x, p_i)$, where $x$ and $p_i$ are d-dimensional vectors, is

Table 3.1: The accuracy for the ME vs. the proposed MRPE on the UCI datasets (in %).

| | Standard ME | Disjoint partition | Overlapping partition |
|---|---|---|---|
| Glass | 87.7 ± 0.61 | 89.3±0.43 | 89.6±0.40 |
| Iris | 88.7 ± 1.05 | 90.9±0.80 | 91.1±0.78 |
| Letter | 88.0 ± 0.43 | 89.5±0.44 | 90.2±0.31 |
| Pendigits | 71.5±0.94 | 73±0.73 | 73.8±0.82 |
| Satimage | 60.9±1.55 | 63.8±1.0 | 64.4±1.21 |
| Segment | 79.0±0.85 | 82.2±0.68 | 82.9±0.79 |
| Vowel | 72.1±1.75 | 75.8±1.77 | 76.9±1.44 |
| Yeast | 50.6±2.22 | 52.7±1.56 | 54.0±1.45 |

$3d$. At each evaluation, the gating network has to calculate $N_e$ times the distance. Assuming that the function evaluation is $h_i(x) = 1 - \frac{d_i}{\|D(x)\|}$, the expected run-time performance is:

$$T_g = N_e \cdot (3d + 2.5) \tag{3.10}$$

As an example, let us suppose that we have an implementation of the ME model with three experts and a gate network with 20 nodes in its hidden layer. Moreover, let us assume that we must deal with a 20-class problem characterized by a 50-dimentional sample space. The ratio of $T_{g_{ME}} / T_{g_{MRPE}}$ is 4.65 which clearly shows a significant saving in terms of evaluation complexity.

## 3.4 Experimental Results and Discussion

Some UCI machine learning data sets [40] have been used to check the validity of the proposed method. These data sets include real-world and synthetic problems, with variable characteristics. For the datasets with no train/test partitioning, the classification performance assessed by the 10-fold cross-validation provides realistic generalization accuracy for unseen data. To build the standard HME and the proposed HMRPE models, we used a Multi-Layer Perceptron (MLP) architecture with one hidden layer, trained with the back-propagation learning rule [47]. To determine the best value for the $N$ partitions, which is equal to the number of experts, we varied it from 2 to 10 for each dataset. We also varied the number of hidden neurons in expert networks to experimentally find the optimal architecture of the MLP experts for each problem. The results of these experiments (shown in Tables 3.1 and 3.2) highlight that the proposed method outperforms the standard ME and its hierarchical counterpart for all selected datasets, no matter whether disjoint or overlapping partitions are adopted.

For further comparison, the time required for training the different datasets is shown in Table 3.3, which highlights that the training time of the proposed method is considerably shorter than the standard version. Simulations are performed using an Intel CPU with 2.83GHz and 4GB RAM memory. Note that the results presented here which compare standard HME and the proposed method, use the same parameters and architecture.

Table 3.2: The accuracy for the HME vs. the proposed HMRPE on the UCI datasets (in %).

| | Standard HME | Disjoint partition | Overlapping partition |
|---|---|---|---|
| Glass | 88.7±0.59 | 89.3±0.55 | 90.5±0.49 |
| Iris | 87.6±1.1 | 90.2±0.7 | 91.3±0.7 |
| Letter | 89.0±0.81 | 89.0±0.45 | 90.2±0.4 |
| Pendigits | 70.9±0.44 | 72.9±1.1 | 73.1±1.05 |
| Satimage | 61.5±2.05 | 62.3±2.0 | 64.1±2.3 |
| Segment | 78.5±0.95 | 82.9±0.78 | 83.8±0.8 |
| Vowel | 73.3±1.8 | 76.8±1.87 | 77.0±1.65 |
| Yeast | 50.0±2.35 | 53.7±2.6 | 54.1±2.6 |

Table 3.3: Training time of the ME and HME vs. the proposed MRPE and HMRPE (seconds).

| | Glass | Iris | Letter | Pendigits | Satimage | Segment | Vowel | Yeast |
|---|---|---|---|---|---|---|---|---|
| Standard ME | 50 | 232 | 351 | 324 | 59 | 49 | 30 | 41 |
| MRPE | 28 | 158 | 221 | 258 | 39 | 32 | 21 | 29 |
| Standard HME | 84 | 324 | 451 | 604 | 99 | 71 | 44 | 67 |
| Hierarchical MRPE | 198 | 311 | 451 | 451 | 63 | 53 | 30 | 42 |

To ensure statistical significance while evaluating the accuracy of classifiers, we used 10-fold cross-validation to compare run-time performances. To build the standard ME and the MRPE models, we used a MLP architecture with one hidden layer, trained with the back-propagation learning rule [47]. To determine the best value for the $N_e$ number of partitions, which is equal to the number of experts, we varied it from 2 to 10 for each dataset. We also varied the number of hidden neurons in expert networks to experimentally find the optimal architecture of the MLP experts for each problem. The results of these experiments are reported for the ME and for the MRPE models, in terms of training time, error rate and testing/online evaluation performance. Figure 4.2 highlights that the latter model outperforms the former for all selected datasets.

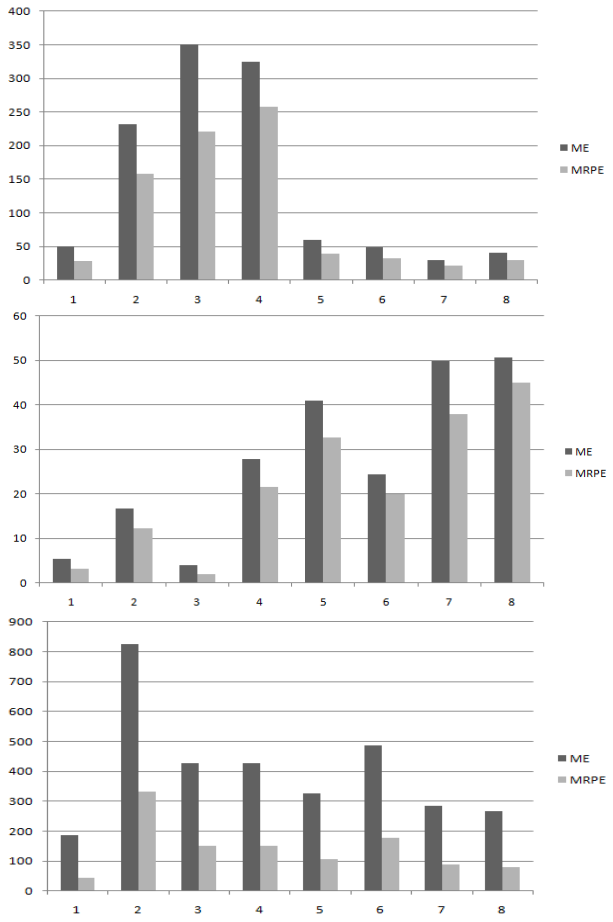


Figure 3.7: ME vs. MRPE in terms of training time, error rate and evaluation time, respectively.

# Chapter 4

# ECOC for TC task

Text categorization is a key task in information retrieval and natural language processing. Providing a reliability measure of the classification result for a text document into a particular category can benefit the recognition rate as well as better inform the user with regard to the confidence that should be attributed to the output. In the first section of this chapter, a novel reliability measure is proposed starting from running different binary classifiers in the Error-Correcting Output Codes (ECOC) framework. Documents classified in a particular category which have a higher ECOC-computed distance from their classification in the next ranked category also have a higher associated reliability. This is the main idea explored in the proposed ECOC-based text classifier with a reject option.

Furtheremore, when a sample belongs to more than one label from a set of available classes, the classification problem (known as multi-label classification) turns to be more complicated. Text data, widely available nowadays in the world wide web, is an obvious instance example of such a task. The second section of this chapter, presents a new method for multi-label text categorization created by modifying the Error-Correcting Output Coding (ECOC) technique. Using a set of binary complimentary classifiers, ECOC has proven to be efficient for multi-class problems. The proposed method, called ML-ECOC, is a first attempt to extend the ECOC algorithm to handle multi-label tasks. Experiments performed for some commonly used text categorization benchmark datasets demonstrate the potential of the proposed method.

## 4.1 ECOC classifier with a reject option

Information growth in the world wide web and digital form makes vital the development of automatic intelligent tools to manage massive amount of data. A large percentage of the information available on the web is stored as text data which needs to be analysed and processed in an efficient way.

Text classification (TC) is a typical information retrieval task with important real-world applications such as document indexing, routing and filtering, web page hierarchical categorization [44]. The problem refers to the classification of text documents into one or more predefined categories. A machine learning approach to this problem builds a classifier based on a training set of labeled documents. Several classifiers have been investigated in the lit-

erature for the text categorization task including neural networks, k-nearest neighbors, support vector machines, naive Bayes and multiple classifier systems [15] [2].

In some applications such as search engines, the prediction of web pages that match user-given keywords does not require to be fully precise since the user is involved in deciding which of the top ranked documents is the most relevant to his/her search. However, the reliability of a classification result is crucial in other applications such as recommender systems. This latter issue is the focus of the current research.

Reliability measures in the context of text categorization can be used to improve the classification accuracy and inform the user (or human operator) on the level of decision confidence given by the automatic system. Research in this area is still in the early stages [18] [19] [56] [39]. The methods proposed so far in the literature need further improving and more efficient reliability models are necessary.

In this section, we propose an ensemble-based reliability measure for the text categorization task. To be more specific, the Error-Correcting Output Codes (ECOC) algorithm [17] is engaged to define the distance of a given document to a category. Comparing the distances of a given document from each category, we can asses the reliability of the candidate class as an output result. The classes with a reliability lower than a threshold suggest the user to follow another classification strategy. Computational experiments are performed for the Reuters benchmark text classification dataset and results are compared with that of related work.

### 4.1.1 Related Work

This subsection focuses on related work devoted to reliability measures for text categorization. Given a classifier result stating that a document $d$ belongs to a class $c_i$, how reliable is this decision and how can it be explored in obtaining better classification accuracies?

In [56], the concept of reliability is used in connection with the problem of literature-based discovery. The authors describe an association rule mining approach to determining relationships among scientific publications. Documents are represented as vectors of weights (the importance of a word in a document). A number of preprocessing techniques including n-gram representation of text (more specifically unigram and bigram), stemming and stopwords are considered. Additionally, a term weighting scheme for indicating the importance of a term in a document is employed. In this context, a reliability measure is defined to assess quality of the discovered patterns (docsets). The reliability measure proposed is based on a citation matrix built from citing and cited information available in a scientific publication database. A $n$-th order association citation matrix of size $m \times m$ (where $m$ is the number of considered documents) is created to indicate the citation paths (of size $n$) from a document to another. Experiments are performed for a dataset collected by the authors from the ACM Digital Library (more than 10000 articles considered). As the authors indicate, the document representation is shown to highly influence the reliability score while bigram scheme significantly outperforms the unigram one.

Fumera et al [18] investigate the potential of introducing a reject option for text categorization. To do this, the authors propose a strategy for deciding if the classifier result is reliable. Normally, the classifier computes a score called *posterior probability* $s_i \in [0,1]$ for a document $d$ to belong to category $c_i$. A threshold $\tau_i$ is then used for each category to decide if document $d$ should be assigned to category $c_i$ i.e. if $s_i(d) \geq \tau_i$ then $d \in c_i$. Fumera et al observe that the classifier decision is highly reliable when the value of $s_i$ is much higher than

the threshold $\tau_i$. Based on this observation, the reject option is simply implemented by using two thresholds (denoted by $\tau_{Hi}$ and $\tau_{Li}$) instead of only one as follows: if $\tau_{Li} < s_i(d) < \tau_{Hi}$ then $d$ is rejected (and later manually classified). Of course, if $s_i(d) \geq \tau_{Hi}$ then $d \in c_i$ and if $s_i(d) \leq \tau_{Li}$ then $d \notin c_i$. This work is extended in [**?**] to include a second stage in which the documents rejected in the first stage are automatically categorized using another classifier. Experiments on the Reuters 21578 benchmark dataset indicate a performance improvement with a reasonable number of rejected documents. It should be noted that in [18] [19] the constraint is put on the maximum number of withheld decisions for each individual category.

Recently, the same authors –Pillai et al [39]– report the results for multi-label classification when the objective is to maximize the classification accuracy on the non-rejected documents with a constraint on the maximum number of rejected ones. A reliability measure is used to decide whether a document should be rejected or not. For each category, the Van Rijsbergen's $F$ measure is used as a scalar performance measure based on the values of precision (the probability that a retrieved document is relevant to a given topic) and recall (the probability that a relevant document is retrieved). Considering multi-label classifiers, the global precision and recall over all categories are obtained by averaging (at micro or macro level) the class values. Correspondingly, the value of the $F$ measure is obtained and denoted by $\hat{F}$. To decide if a document $d$ has to be rejected, the authors define a reliability measure $R(d)$ based on maximizing $\hat{F}$ for the non-rejected documents. The idea is that the higher the value of $\hat{F}$ for all documents except $d$ the less reliable is the classification of $d$. All documents $d$ for which $R(d)$ falls below a prespecified rejection threshold are removed. Experiments are reported for two text categorization benchmark datasets (Reuters 21578 - ModApte and Heart Disease subset of the Ohsumed dataset) and one image annotation task (Scene dataset). The authors report the improvement of the classification accuracy (always increasing with the rejection rate) - particularly when about 30% of samples are rejected.

In [3], a reliability measure for Naive Bayes (NB) is proposed to address text classification problems which involve extremely asymmetric misclassification costs. The NB classifier is extended by modulating the score returned by NB based on the information-theoretic Kullback-Leibler (KL) divergence. The idea is to assess the confidence of NB decisions by measuring the difficulty of reversing the NB result for a given input. Given a classifier result which says that a document $d$ belongs to a certain class, the authors ask the question how much extra training with $d$ is necessary to reverse the classifier outcome. The KL divergence is used to measure the effected change to the training distribution. The paper reports the improvement of results using NB-KL compared to the baseline NB for three benchmark datasets i.e. Reuters 21578, 20 Newsgroups and TREC-AP.

A meta-classifier approach to reliable classification is investigated in [36] [33]. A conformity-based classifier is trained as a meta-classifier to predict the correctness of each document classification of a base classifier. For this purpose, the p-values of the meta-predictions are computed based on non-conformity functions and used in deciding which class output of the base classifier is the most reliable. In this way, the meta-classifier decides if the class predicted by the base classifier for an individual instance is a reliable result (otherwise, the classification is rejected). The estimation of p-values for each class can be however a difficult task in this approach.

Looking over the main existing related work highlights the fact that they mostly use a simple thresholding strategy on the posterior probability or evaluation measures such as $F_1$. Moreover, they normally rely on the opinion of a single expert in a rejection/acceptance

decision, which obviously is not as reliable as an ensemble decision. However, the most important motivation of continuing research in this area is a low performance of existing systems in obtaining a good reliability. The method proposed in this chapter tries to address these issues by consensus of a set of binary experts, forming an ECOC classifier. This strategy takes advantage of significant ensemble features e.g. redundancy and diversity to increase the reliability of decisions.

### 4.1.2 The ECOC Classifier with a Reject Option

The proposed reliability measure for text categorization takes advantage of the computed distances of a given document from any category in an ECOC decoding step to evaluate the confidence of label assignment. ECOC is a strategy to indirectly deal with a multi-class problem by hiring complementary binary classifiers, each focusing on different partitions (dichotomies) of the problem [17]. The main advantages of ECOC, which particularly proved to be efficient for large number of classes, are as follows: (i) possibility of using strong binary classifiers such as boosting and support vector machines (SVM) algorithms which can not directly address multi-class problems, (ii) generally, it is expected to be easier to address binary problems than multi-class problems with two many classes (divide and conquer principle), (iii) introduces redundancy for the same solutions so that if a classifier makes a mistake the final true label can still be recovered using information given by the other classifiers which have contributed to the same task (error-correcting property), and (iv) in datasets with small number of samples per class or imbalanced classes such as text data, the ECOC approach can lead to denser problems by merging different classes into a superclass which can *potentially* be better addressed by a *dichotomizer*.

The ECOC matrix codifies the class labels in order to achieve different partitions of classes considered by each dichotomizer. The main coding strategies can be divided into problem-independent (or fixed) [48] [48] and problem-dependent [21] [37] [38] [62].

The main idea of the proposed method is that a category assigned to a document is more reliable if the distance of predicted codeword to the candidate class is shorter than the distance to the second ranked candidate. For the ECOC-based text categorization system, the proposed reliability measure for the classification of a document $d$ is defined as follows:

$$Reliability(d) = \frac{H_d(cw_2, \overline{y}_d) - H_d(cw_1, \overline{y}_d)}{H_d(cw_2, cw_1)} \times 100 \tag{4.1}$$

where $cw_1$ and $cw_2$ are the closest and second closest rows of the code matrix to the output vector $\overline{y}_d$ given by ECOC classifier for each test document and $H_d$ is the Hamming distance between two codewords. A reliability threshold can be set on $Reliability$ so that testing documents with reliability smaller than the threshold can be rejected.

Finally, the recognition rate and the rejection rate computed by the proposed method for text categorization with a reject option are defined as follows:

$$Recognition\ Rate = \frac{\psi_c}{\psi_a} \tag{4.2}$$

$$Rejection\ Rate = \frac{\psi_r}{\psi_n} \tag{4.3}$$

where $\psi_c$ is the number of correctly labeled documents, $\psi_a$ is the number of accepted documents which obtained the reliability score above the calculated threshold, $\psi_r$ is the number

of rejected documents and $\psi_n$ represents the total number of documents. The threshold can be adjusted based on a trade-off between recognition rate and rejection rate. For example, for applications with low tolerance on errors such as those in information security, the threshold should be set higher and the error rate can be reduced at the cost of more rejections.

## 4.2 Multi-Label ECOC

Text Categorization (TC), also known as document classification, plays a key role in many information retrieval (IR) -based systems and natural language processing (NLP) applications. First research on TC goes back to Maron's [34] seminal work on probabilistic text classification. Since then, TC has been used for a number of different applications using techniques from machine learning, pattern recognition and statistics. In [15], TC applications are grouped into hierarchical categorization of web pages, word sense disambiguation, automatic indexing for boolean IR systems, document filtering and organization. Speech categorization as combination of a speech recognition and TC methods, multimedia document categorization through the analysis of textual captions, author identification for literary texts of unknown or disputed authorship, language identification for texts of unknown language, automated identification of text genre, and automated essay grading are some examples for such applications in real-world problems [50] [52] .

A traditional classification problem in pattern recognition refers to assigning any incoming sample to one of two (binary problem) or more (multi-class problem) distinct predefined classes. An even more complex scenario, called multi-label classification, is one in which the classes have overlap between each other. TC or automatically labeling natural language texts with thematic categories from a predefined set is one such task. An instance document or web page about "Persian carpet exhibition" can belong to both "economy" and "art" categories. Despite its multi-label nature, the majority of research studies on TC have considered it as single-label task by assigning the samples into only one of the existing classes. However, this approach simplifies the task and handles it using a huge bibliography of learning algorithms, yet failing to provide a complete solution to multi-label TC.

There are two main approaches in the literature to deal with multi-label classification: *(i) Problem transformation* approaches which transform the multi-label problem into one or more single-label problems, and *(ii) Algorithm adaptation* approaches which extend specific learning algorithms in order to handle the multi-label task directly. Although many approaches have been proposed based on different kinds of classifiers and architectures over a variety of application domains, there is no clear winner method over the rest (see [59] [11] for some recent surveys) and each of them has its own advantages and disadvantages.

This section proposes a method for multi-label TC called ML-ECOC created by extending the ECOC strategy. ML-ECOC modifies the coding/decoding phases of the standard ECOC algorithm making it suitable to the multi-label problems. This modification includes setting up new rules in both coding and decoding phases to avoid the occurrence of any inconsistency while handling multi-label data. Experiments on the text mining problem of Multi-Label Text Categorization (ML-TC) show a good performance of the proposed ML-ECOC. Comparissons to the state-of-the-art methods from different perspectives are carried out and the obtained results are analysed in detail.

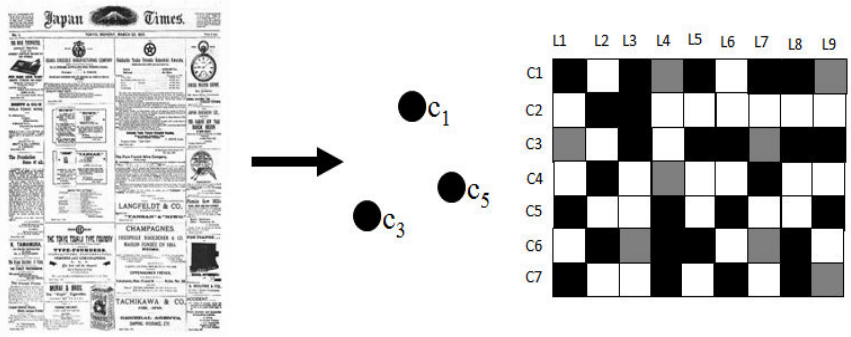


Figure 4.1: An instant document $d$ and a codematrix of $7 \times 9$.

### 4.2.1 Multi-Label ECOC for TC

The first application of ECOC algorithm on TC dates back to 1999 [10] [20], However, in these studies, the authors simply use standard single-label classifiers and view the problem as a traditional multi-class classification. Since then, many researchers also used ECOC with different types of classifiers on various applications but with more or less the same assumptions. From the ECOC literature, one can conclude that there are three main possible ways to improve ECOC classifiers: (i) code matrix design, (ii) building binary classifiers, and (iii) decoding step. In TC area, the improvements are mainly limited to the second option i.e. building binary classifiers as accurate as possible. This goal is achieved in [57] by Model-Refinement strategy which is used to adjust the so-called bias in centroid classifiers. The basic idea is to take advantage of misclassified examples in the training data to iteratively refine and adjust the centroids of text data. In [32], Li et al. proposed a simple strategy to improve binary text classification via multi-class categorization (dubbed 2vM) for applications where sub-class partitions of positive and/or negative classes are available. As multi-class categorization may implicitly capture the interactions between sub-classes, detailed subclasses are expected to help differentiating the positive and negative classes with high accuracy.

The reason that all these works are limited to single-label assumption is that an *inconsistency* would occur otherwise in ECOC classification while applying to multi-label data. For instance, imagine a document $d$ belongs to a label set $[1,3,5]$, each label representing a content based topic. Also imagine 5-th column of an instance (predefined or given) matrix $M^{7\times 9}$ shown in Figure 4.1 which is used to create dichotomizer $f_5$. Considering $d \rightarrow \omega = [c_1, c_3, c_5]$, now the question is which super-class sample $d$ belongs to (+1 or -1)? According to traditional decoding of ECOC, the sample belongs to both super-classes of the dichotomy at the same time. This inconsistency in assignment of $d$ is not only limited to $f_5$ but also occures for dichotomies 3,4, 6, 7 and 8. In fact, standard ECOC algorithm is only capable of single-label prediction for a traditional multi-class problem while it suffers from lack of capability to handle multi-label data in general. Therefore, a modification in the ECOC algorithm is required such that it can directly address multi-label data in both training the dichotomizers and label set prediction without any assumption and limitation. As mentioned before, the only way to address this issue so far was simplifying the problem to single-label classification [20] [10].

Although the single-label assumption may be true in some TC applications, it certainly limits the application of ECOC to real-world multi-label cases. This is the point where ECOC algorithm requires a major modification to be applicable to multi-label problems. In the

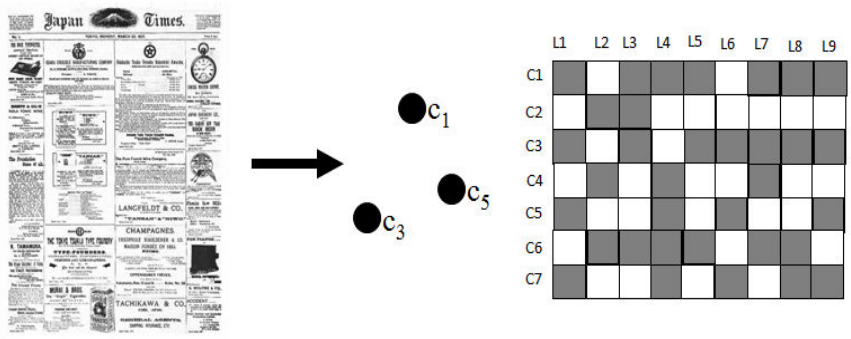


Figure 4.2: ML-ECOC: Gray and white boxes represent 0 and +1, respectively.

following, we introduce the ML-ECOC method to address any multi-label problem without any constraint and restricting assumption.

The main idea of ML-ECOC is to generate a codeword for each category of a TC task with only +1 (positive class) and 0 (don't care) bits. Unlike standard ECOC algorithm, where at least one +1 and one −1 bits are required at each column to define a dichotomy, to be non-zero is all ML-ECOC needs for a column. A classifier defined according to each column of the ML matrix and used to calculate degree of membership of $d$ into a super-class which includes one or more categories. The inconsistency in the dichotomizing process is avoided by defining only positive class and *neutral* set which can not have any overlapping area. It is worth noting that a document belongs to $i$th positive class *if and only if* at least one of its labels from the label set is in the $i$th super-class. A document $d$ (Figure 4.2) either should belong to positive class of $i$th column or its neutral set. For instance, $d$ is a member of 2,3,4,5,6,7 and 8 positive class sets while should be considered as neutral for 1st and 9th.

Subsequently, it is obvious that this modification requires also different decoding strategy, since standard Euclidean or Hamming distances with $ArgMax$ labeling are not applicable anymore. Let us suppose a predicted codeword $\overline{y}_d = [\overline{y_1}...\overline{y_L}]$, $0 \leq \overline{y_l} \leq +1\}$ is a string assigned to document $d$ (each bit representing the output of a classifier i.e. $\mathscr{P}_l(+1 \mid d)$). The posterior probability of each class using ML-ECOC is calculated as follows:

$$\mathscr{P}(c_N \mid d) = \frac{1}{\mid M(N,.) \mid} \sum_{l=1}^{L} \mathscr{P}_l(+1 \mid d) M(N,l) \tag{4.4}$$

For each document, ML-ECOC sorts categories by score and assigns YES to each of the $t$ top-ranking categories. Parameter $t$ is an integer ranging from 1 to the number of categories $N_c$ whose value can be either specified by the user or automatically tuned using a validation set. It should be noted that when $t = 1$, this multi-label assignment turns into the standard single-label TC with $ArgMax$ rule. Obviously, it is just typical thresholding strategy adopted to ML-ECOC and the other existing thresholding methods can be applied. The generic ML-ECOC is summarized in Algorithm 1.

### 4.2.2 Why does ML-ECOC work?

– The success of the ML-ECOC idea can be attributed to following three factors:

**Algorithm 3** ML-ECOC.

**I**nput: $\mathscr{X}_t$, $\mathscr{T}_t$ training set, $\mathscr{X}_e$, $\mathscr{T}_e$ testing set and $f$ learning algorithm.

**T**raining:

1. generate a binary codemtrix $M^{N_c \times L}$ which $N_c$ is the number of categories and $L$ varies with coding strategy.
2. for $i$-th column in $M$:
3. build (create) one-class set made of $\mathscr{T}_i^+$ and $\mathscr{T}_i^*$ supper-classes (positive and neutral sets respectively)
4. train $i$-th classifier $f_i$ with $i$-th training set

**T**esting:

1. apply $\mathscr{X}_e$ on entire set of $f_i$s
2. create a codeword which $i$-th bit is $f_i(\mathscr{X}_e) = \mathscr{P}_i(+1 \mid \mathscr{X}_e)$
3. calculate the posterior probability for each class using Eq. 4.4
4. use multi-label decoding to predict label set

**O**utput: $\bar{\omega} = [\bar{c_p}, \bar{c}_q, \bar{c}_r]$

1. Unlike the standard TC approaches trying directly to discriminate different classes, ML-ECOC transfers the entire class space to many super-classes, which are not necessarily carrying meaningful concepts, by mixing them . This is helpful particularly to deal with what is called in the literature *Data sparsity*. This is a measure for how much data we have for a particular dimension/entity of the model. A dataset is sparse if the number of samples for each class is not enough for a classifier to discriminate it from the rest which is normally the case in the TC problem. Therefore, mixing categories by ML-ECOC decomposing, not only used to define new class-boundaries which might provide additional information in final decision making, but also provides new one-class problems with more samples per positive class (in the case each super-class has more than one category). For instance, each super-class in first dichotomy of Figure 4.2 is made of 3 categories.
2. No matter which TC approach is chosen, a class-label is assigned to a document if its corresponding classifier *fires*. In fact, when a category is wrongly detected, there is no any *efficient* way to go back and fix it without the increase of the algorithm complexity and computational cost. However, in ML-ECOC there is no *dedicated* classifier for each category and decisions are made by *consensus* of all classifiers. Therefore, because of its *error-correcting* capability, even if some errors occur in the bit level, the final

Table 4.1: The main characteristics of the selected subset of the datasets.

| problem | total samples | nominal | numeric | label | cardinality | density | distinct |
|---|---|---|---|---|---|---|---|
| rcv1v2 | 600 | 0 | 47235 | 103 | 2.642 | 0.026 | 946 |
| tmc2007 | 28596 | 49060 | 0 | 22 | 2.158 | 0.098 | 1341 |

decision can still be reliable.

3. Another important issue arising while dealing with TC refers to *class-imbalanced* datasets where there is no balance between the positive and negative set of a category. This problem can badly affect the learning process particularly in the *Local Classifier per Category* approach when a category stands against the rest. ML-ECOC keeps more balance between two resulted positive classes and neutrals by having chance of including more than one class in the positive class set. For instance *Sparse-random* method can possibly include more than one category in a positive class resulting into more balanced data. Consequently, efficient learning of the class boundaries by classifiers results in more accurate prediction.

## 4.3 Numerical Experiments and Results

For the text categorization experiments, we have chosen two commonly used multi-label datasets i.e. the Reuters (RCV1-V2) and TMC2007. A brief description of each is given below.

*RCV1-V2*: Reuters Corpus Volume1-Version2 is a large-scale dataset for text classification task. It is based on the well known benchmark dataset for text classification, the Reuters (RCV1) dataset. We use the topics full set 3 that contains (804,414) news articles. Each article is assigned to a subset of the 103 topics. A detailed description of the RCV1 dataset can be found in [31]. We pre-processed RCv1v2 documents as proposed by Lewis et al. [31] and, in addition, we separated the training set and the testing set using the same split adopted in [31]. In particular, documents published from August 20, 1996 to August 31, 1996 (document IDs 2286 to 26150) are included in the training set, while documents published from September 1, 1996 to August 19, 1997 (document IDs 26151 to 810596) are considered for testing. The result is a split of the 804,414 documents into 23,149 training documents and 781,265 test documents. In order to save computational resources, we have randomly chosen 600 documents (300 training documents and 300 testing documents) as indicated in Table 4.1.

*TMC2007*: This is the dataset used for the SIAM 2007 competition organized by the text mining workshop held in conjunction with the 7th SIAM International Conference on Data Mining [1]. This competition sponsored by NASA Ames Research Center, focused on developing text mining algorithms for document classification. It contains 28596 aviation safety reports in free text form, annotated with one or more out of 22 problem types that appear during certain flights [54]. The dataset comes from human generated reports on incidents that occurred during the flights which means there is one document per incident. Text representation follows the boolean bag-of-words model. The goal was to label the documents with respect to the types of problems that were described. This is a subset of the Aviation Safety Reporting System (ASRS) dataset, which is publicly available. Some other statistics of the dataset are given in Table 4.1.

In the applications using text categorization as the core task, the computational efficiency is crucial because of very large number of features, classes and samples. Therefore, the need for designing a simple and fast classification system is important. There are many research studies using different kinds of classifiers such as k-nearest neighbors (kNN), support vector machines (SVM), artificial neural networks (ANN), bayesian methods and rocchio classifiers [15]. However, in practice most of them are not applicable as in real-world applications, e.g. search engines and recommender systems, a just-in-time response has great importance. Among them, the naive bayes and centroid classification algorithms are extremely simple and straightforward illustrating competitive performance on text categorization problems. Moreover, they do not need to memorize a huge amount of training data as some other classifiers do (e.g. kNN) and adjust so many parameters (e.g. ANN).

For the experiments presented in the current chapter, we used *centroid-based classifiers* as the ECOC dichotomizers. This means that the prototype vector or centroid vector ($\mu_i^+$) is computed for super-class $\mathcal{T}_i^+$ as:

$$\mu_i^+ = \frac{1}{|\mathcal{T}_i^+|} \sum_{d \in \mathcal{T}_i^+} d \tag{4.5}$$

where $|\mathcal{T}_i^+|$ denotes the cardinality of set $\mathcal{T}_i^+$, i.e. the number of documents that belong to positive set in the $i$-th individual and $d$ is a training document.

In the testing step, we calculate the similarity of a document $d$ to each centroid by the *cosine* measure,

$$S(d, \mu_i^+) = \frac{d \cdot \mu_i^+}{\| d \| \cdot \| \mu_i^+ \|} \tag{4.6}$$

This similarity can be regarded as the *posterior probability* of the dichotomizer and used for $i$-th bit of the predicted codeword $\bar{y}_d$.

Numerical results emphasize the performance of the proposed method in discriminating between the reliable and unreliable decisions made for documents classified by ECOC algorithm. Documents with low *reliability score* (i.e. lower than the threshold $\tau$) are rejected whereas those with higher *reliability score* are accepted. However, finding an optimal value for $\tau$ proved to be important and should be done by considering all factors of the classifiers. Among them, the recognition and rejection rates are the most effective ones and should be very well adjusted while designing a classifier with a reject option. False Rejection (FR) occurs either when a category is *correctly* assigned to a document with low reliability or when high reliability is given to a document with *incorrectly* predicted label. The results obtained for the Reuters dataset are presented in Table 4.2. The proposed method correctly rejects documents falsely predicted by classifiers, i.e. True Rejection (TR) ratio, whereas the FR ratio is very low considering the large number of test documents.

It is important to note that the FR and TR ratios are directly related to the threshold $\tau$. Higher thresholds avoid the label assignment for false predicted documents while the FR increases. Therefore, in applications with high cost for mislabeling, the proposed strategy reduces the cost at the risk of rejecting some correctly labeled documents.

We have compared the results of the proposed method with those of different commonly used TC algorithms [4] [35] as well as the related reliability methods [18] [19]. The standard TC methods used for comparison are the big-bang (global method), the flat method, Local

Table 4.2: Results obtained by the proposed method.

| Problem | Rejected documents | TR | FR | Accuracy boost (%) |
|---|---|---|---|---|
| Topics | 97 | 87 | 10 | 3.8 |
| Industry | 88 | 79 | 9 | 3.7 |
| Regions | 90 | 83 | 7 | 1.3 |

Table 4.3: Recognition rate of the proposed method compared to the standard TC methods.

| Problem | big-bang | LCN | ECOC D-Rand | ECOC S-Rand | ECOC BCH | Proposed Method (D-Rand) | Proposed Method (S-Rand) | Proposed Method (BCH) |
|---|---|---|---|---|---|---|---|---|
| Topics | 34.5 | 34.1 | 35.5 | 34.4 | 35.9 | **37.9** | 37.0 | 36.9 |
| Industry | 38.3 | 36.7 | 38.1 | 39.1 | 38.9 | 40.0 | **40.5** | 39.9 |
| Regions | 38.0 | 37.0 | 37.5 | 36.8 | 37.3 | **38.3** | 38.2 | 38.0 |

Classifier per Node (LCN) and ECOC classifier with various matrices (without a reject option). For all these methods, centroid-based classifiers with the same parameters have been implemented. As shown in Table 4.4, the proposed method boosts the accuracy of the standard TC approaches using the ECOC algorithm and outperforms the related TC reliability methods.

Consequently, the evaluation of methods to handle multi-label data requires different measures than those used for traditional single-label classification. Various measures are traditionally being used for evaluation of multi-label classification (particularly for document and text applications) such as *classification accuracy*, *precision*, *recall* and *F1*. These are defined below.

$$classification\ accuracy = \frac{1}{n}\sum_{d=1}^{n} \mathscr{I}(\omega_d = \bar{\omega_d}) \tag{4.7}$$

where $I(true) = 1$ and $I(false) = 0$ and $n$ is the number of documents in a dataset. This is a very strict evaluation measure as it requires the predicted set to be an exact match for the true set in the label set no matter if a classifier makes a mis-classification at only one category or the entire set.

$$precision = \frac{1}{N_c}\sum_{c_i=1}^{N_c} \frac{TP_{c_i}}{TP_{c_i} + FP_{c_i}} \quad and \quad recall = \frac{1}{N_c}\sum_{c_i=1}^{N_c} \frac{TP_{c_i}}{TP_{c_i} + FN_{c_i}} \tag{4.8}$$

where $TP$, $FP$ and $FN$ stand for the true positive, false positive and false negative for each category, respectively. The F1-score which considers both the $precision$ and $recall$ of the test set is formulated as:

$$F1 = \frac{2precision.recall}{precision + recall} \tag{4.9}$$

where an $F1$ score reaches its best value at 1 and worst score at 0.

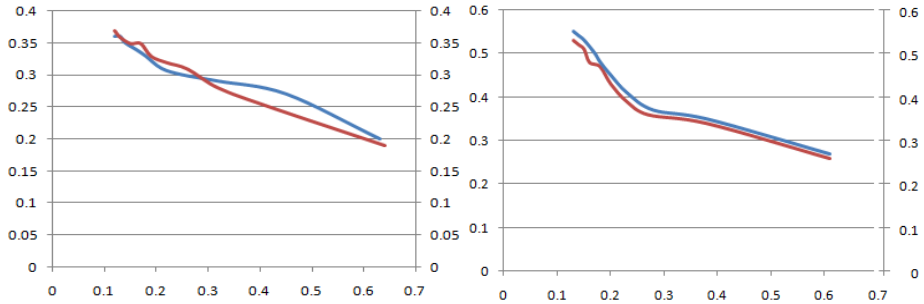


Figure 4.3: Precision-Recall curves for the RCV data (up) and TMC2007 (bottom).

Table 4.4: F1 score of the proposed method (PM)compared to thetandard TC methods.

| Problem | big-bang | LCC | ML-ECOC (drand) | ML-ECOC (2vsA) |
|---|---|---|---|---|
| rcv1v2 | 34.5 | 34.1 | **37.9** | 37.0 |
| tmc2007 | 38.3 | 36.7 | 40.0 | **40.5** |

We have compared the results of the proposed method with some of commonly used TC algorithms. The standard multi-label TC methods used as baseline methods are the big-bang (global method) and Local Classifier per Category (LCC). For all these methods, centroid-based classifiers with the same parameters have been implemented. As shown in Table 4.4, the proposed ML-ECOC using Dense random and 2vsA codes outperforms the standard TC approaches on the selected datasets by obtaining the maximum F1 scores. One can note that the results for 2vsA code for rcv1v2 data is missing. This is because of large number of classes of RCv1v2 data which make building ECOC classifier unfeasible.

To give more detailed information, Figure 4.3 shows precision-recall curves, red and blue curves being ML-ECOC and LCC approaches, respectively. Because of the superior performance on ML-TC datasets, the LCC approach is used to compare with the ML-ECOC. As clearly shown, the proposed ML-ECOC is able to obtain slightly better results on RCv1-v2 while always winning on TMC2007 data.

# Chapter 5

# Concluding remarks

As concluding remarks, let us briefly recall the ideas and the results concerning the methods described in this thesis.

Chapter 2 presents a novel miniensemble method, to be used in the selection-fusion framework. The proposed method, called random prototype oracle, splits the input domain based on some prototypes selected randomly from training data and then builds a classifier on each subset. These classifiers are used in combination with an oracle that knows the area of expertise of each classifier, thus generating a miniensemble. Thanks to the random nature of the decomposition procedure, miniensembles created on a specific problem differ from one run to another. These miniensembles can be used as base classifiers in any ensemble strategy to improve its accuracy without increasing the computational cost. We carried out experiments on eight data sets taken from the UCI machine learning repository. First we show that the method introduced here results in more accurate miniensembles with respect to the traditional random linear oracle. Then we have shown that RPO improves the performance of standard ensemble strategies. A critical point of the proposed method is to determine the best number for splits, which in principle depends on the nature and on the complexity of datasets and learners. Although currently found experimentally, we are studying a way for estimating it with an automatic procedure.

Chapter 3 presents a modified version of the popular ME algorithm. Unlike the standard ME, which specializes expert networks on nested and stochastic regions of the input space, the proposed method partitions the sample space into subspaces based on similarities with randomly-selected prototypes. This strategy enables to define a simple rule for the gating network for both training and testing. As shown by experimental results, despite its simplicity, the proposed method improves the accuracy of the both standard ME and HME models while reducing the training time. Furthermore, the expected run-time performance for the standard ME model and for the MRPE model have been formulated for the first time. Theoretical results are confirmed by experimental results, which also highlight that MRPE performs better also in terms of accuracy. Experimental results have been reported for error rate, training time and testing/online evaluation time. Future work for this research topic will be focused on defining a light procedure for automatically determining the number of experts for a given problem, without resorting to complex preprocessing and time consuming methods. Adapting this method to simple distance-based classifiers (instead of artificial neural networks) is another interesting future research direction, concerned with reducing

the training time of the overall network while maintaining high accuracy. We are also experimenting heuristics able to help in the process of partitioning the input space (instead of using random prototypes).

In the first section of Chapter 4, an efficient distance-based rule is introduced to evaluate the reliability of decisions made by ECOC text classifier for a given document. The proposed approach relies on the computed distances of an incoming document from each category given by the ECOC decoding step to make a final decision about the candidate category. A document will be assigned to a class with maximum posterior probability if its reliability score is higher than a predefined threshold. This way, by double checking the candidate category, the reliability of the decision made by ECOC classifier is increased. Experiments show capability of the method to boost the recognition rate with rejecting decisions about document class which have been assigned a low reliability. Future work for this research topic focuses on adapting the proposed ECOC-based reliability to more complicated paradigms such as multi-label [8] or hierarchical TC problems since the application of ECOC classifier on these areas has not been explored so far. Investigating different strategies to formulate reliability measures based on binary classifiers opinion given by ECOC ensemble is another interesting research line.

In the second section of Chapter 4, an extension of the ECOC algorithm called ML-ECOC is proposed to tackle multi-label TC problems. To avoid the inconsistency in coding step, the proposed ML-ECOC method decomposes a multi-label problem into some complementary one-class sub-problems unlike the standard ECOC which builds dichotomies. Multi-label relationship is taken into account in the testing phase by using a novel decoding strategy adopted for ECOC algorithm. Experimental results on Reuters datasets confirm the potential of the proposed ML-ECOC on multi-label classification with large number of categories. Recently, some studies [7] [53] try to increase ECOC reliability by proposing a reject mechanism. One interesting future research line refers to multi-label text categorization with a reject option.

Summarizing, this thesis proposes some new ideas in the field of classifier ensembles, making a step forward in this research topic. All methods described in the thesis have been successfully applied to various problems, including publicly available benchmark datasets from UCI repository and Reuters version 2 text categorization task.

# Bibliography


[1] http://www.cs.utk.edu/tmw07/. [cited at p. 37]

[2] Hotho A., Nurnberger A., and G. Paass. A brief survey of text mining. *LDV Forum*, 20(1):19–62, 2005. [cited at p. 30]

[3] Kolcz A. and Chowdhury A. Improved naive bayes for extremely skewed misclassification costs. *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2005. [cited at p. 31]

[4] Kosmopoulos A., Gaussier E., Paliouras G., and Aseervatham S. The ecir 2010 large scale hierarchical classification, workshop report. 2010. [cited at p. 38]

[5] A. Ahmad and G. Brown. A study of random linear oracle ensemblese. *MCS 2009. LNCS, Springer, Heidelberg*, 5519:488–497, 2009. [cited at p. 12]

[6] E. Alpaydin and E. Mayoraz. Learning error-correcting output codes from data. *Intl. Conf. on Artificial Neural Networks (ICANN'99)*, pages 743–748, 1999. [cited at p. 8]

[7] G. Armano, C. Chira, and N. Hatami. Ensemble of binary learners for reliable text categorization with a reject option. *HAIS 2012*, page in press, 2012. [cited at p. 42]

[8] G. Armano, C. Chira, and N. Hatami. Error-correcting output codes for multi-label text categorization. *Proceedings of 3rd Italian Information Retrieval Workshop (IIR 2012), Bari*, page in press, 2012. [cited at p. 42]

[9] Tang B., Heywood M., and Shepherd M. Input partitioning to mixture of experts. *in: International Joint Conference on Neural Networks*, pages 277–232, 2002. [cited at p. 6, 19]

[10] A. Berger. Error-correcting output coding for text classification. *In Proceedings of IJCAI-99 Workshop on Machine Learning for Information Filtering*, 1999. [cited at p. 34]

[11] Wei Bi and James T. Kwok. Multilabel classification on tree-and dag-structured hierarchies. *ICML*, pages 17–24, 2011. [cited at p. 33]

[12] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. [cited at p. 3, 15]

[13] Wan E. and Bone D. Interpolating earth-science data using rbf networks and mixtures of experts. *in: NIPS*, pages 988–994, 1996. [cited at p. 6, 19]

[14] S. Escalera, O. Pujol, and P. Radeva. Boosted landmarks of contextual descriptors and forest-ecoc: A novel framework to detect and classify objects in cluttered scenes. *Pattern Recognition Letters*, 28:1759–1768, 2007. [cited at p. 8, 9]

[15] Sebastiani F. Machine learning in automated text categorization. *ACM Computing Surveys*, 34, issue 1, 2002. [cited at p. 30, 33, 38]

[16] Y. Freund. Boosting a weak learning algorithm by majority. *Proceedings of the Third Annual Workshop on Computational Learning Theory*, 1990. [cited at p. 4]

[17] Dietterich T. G. and Bakiri G. Solving multiclass learning problems via error correcting output codes. *J. of Artificial Intelligence Research*, 2:263–286, 1995. [cited at p. 3, 8, 15, 30, 32]

[18] Fumera G., Pillai I., and Roli F. Classification with reject option in text categorisation systems. *Proc. 12th International Conference on Image Analysis and Processing. IEEE Computer Society*, pages 582–587, 2003. [cited at p. 30, 31, 38]

[19] Fumera G., Pillai I., and F. Roli. A two-stage classifier with reject option for text categorisation. *Proc. Structural, Syntactic, and Statistical Patt. Rec.*, pages 771–779, 2004. [cited at p. 30, 31, 38]

[20] R. Ghani. Using error-correcting codes for text classification. *17th International Conference on Machine Learning*, 2000. [cited at p. 34]

[21] N. Hatami. Thinned-ecoc ensemble based on sequential code shrinking. *Expert Systems with Applications*, 39(1):936–947, 2012. [cited at p. 8, 9, 32]

[22] T. K. Ho. The random space method for constructing decision forests. *IEEE Trans. on PAMI*, 20(8):832–844, 1998. [cited at p. 15]

[23] Jordan M. I. and Jacobs R. A. Hierarchical mixtures of experts and the em algorithm. *Neural Comp*, 6:181–214, 1994. [cited at p. 6, 18]

[24] Kuncheva L. I. Combining pattern classifiers: Methods and algorithms. *Wiley Interscience*, 2004. [cited at p. 22]

[25] Hennessy J. and Patterson D. Computer architecture: a quantitative approach. *Morgan Kaufmann, San Mateo, CA*, 1990. [cited at p. 24]

[26] Crammer K. and Singer Y. On the learnability and design of output codes for multiclass problems. *Machine Learning*, 47(2):201–233, 2002. [cited at p. 8]

[27] M. Kearns. Thoughts on hypothesis boosting. *Unpublished manuscript*, 1988. [cited at p. 4]

[28] Ali K.M and Pazzani M.J. On the link between error correlation and error reduction in decision tree ensembles. *Technical Report ICS-UCI*, pages 95–138, 1995. [cited at p. 3]

[29] L. I. Kuncheva and J. J. Rodriguez. Classifier ensembles with a random linear oracle. *IEEE Trans on Knowledge and Data Eng.*, 19(4):500–508, 2007. [cited at p. 3, 5, 11, 12]

[30] A. Krzyzak L. Xu and C.Y. Suen. Methods for combining multiple classifiers and their applications in handwritten character recognition. *IEEE Trans. Systems, Man, and Cybernetics*, 22:418–435, 1992. [cited at p. 9]

[31] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004. [cited at p. 37]

[32] Baoli Li and Carl Vogel. Improving multiclass text classification with error-correcting output coding and sub-class partitions. *Canadian Conference on AI*, pages 4–15, 2010. [cited at p. 34]

[33] Kaptein A. M. Meta-classifier approaches to reliable text classification. *Master Thesis, Universiteit Maastricht, The Netherlands*, 2005. [cited at p. 31]

[34] Maron M. Automatic indexing: an experimental inquiry. *J. Assoc. Comput. Mach.*, 8(3):404–417, 1961. [cited at p. 33]

[35] Silla Jr. C. N. and Freitas A. A. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 20(1), 2010. [cited at p. 38]

[36] Smirnov E. N., Nalbantov G. I., and Kaptein A. M. Meta-conformity approach to reliable classification. *Intell. Data Anal.*, 13(6):901–915, 2009. [cited at p. 31]

[37] Pujol O., Radeva P., and Vitria J. Discriminant ecoc: A heuristic method for application dependent design of error correcting output codes. *IEEE Transactions on PAMI*, 28(6):1001–1007, 2006. [cited at p. 8, 32]

[38] Pujol O., Escalera S., and Radeva P. An incremental node embedding technique for error correcting output codes. *Pattern Recognition*, 41:713–725, 2008. [cited at p. 8, 9, 32]

[39] I. Pillai, Fumera, G., and Roli F. A classification approach with a reject option for multi-label problems. *Proceedings of the ICIAP (1)*, pages 98–107, 2011. [cited at p. 30, 31]

[40] Murphy P.M. and D.W. Aha. Uci machine learning repository, univ. of california, irvine. 1994. [cited at p. 14, 26]

[41] Avnimelech R. and Intrator N. Boosted mixture of experts: an ensemble learning scheme. *Neural Comput.*, 11:483–497, 1999. [cited at p. 6]

[42] Duda R., Hart P., and Stork D. Pattern classification. *John Wiley & Sons, New York, NY, 2nd edition*, 2001. [cited at p. 24]

[43] Ebrahimpour R., Kabir E., and Yousefi M.R. Teacher-directed learning in view-independent face recognition with mixture of experts using overlapping eigenspaces. *Computer Vision and Image Understanding*, 111:195–206, 2008. [cited at p. 6, 19]

[44] Feldman R. and Sanger J. The text mining handbook - advanced approaches in analyzing unstructured data. *Cambridge University Press I-XII*, 2007. [cited at p. 29]

[45] Jacobs R.A., Jordan M.I., Nowlan S.E., and G.E. Hinton. Adaptive mixture of experts. *Neural Comput.*, 3:79–87, 1991. [cited at p. 3, 5, 18]

[46] J. J. Rodriguez and L. I. Kuncheva. Naive bayes ensembles with a random oracle. *MCS 2007. LNCS, Springer, Heidelberg*, 4472:450–458, 2004. [cited at p. 12]

[47] Haykin S. Neural networks: A comprehensive foundation. *Prentice Hall, 2nd Edition*, 1999. [cited at p. 26, 27]

[48] Lin S. and Costello. D. J. Error control coding, second edition. *Prentice-Hall, Inc.*, 2004. [cited at p. 7, 32]

[49] Waterhouse S. and Cook G. Ensemble methods for phoneme classification. *in: M. Mozer, J. Jordan, T. Petsche (Eds.), Advances in Neural Information Processing Systems, The MIT Press, Cambridge, MA*, 9:800–806, 1997. [cited at p. 6]

[50] C. L. SABLE and V. Hatzivassiloglou. Textbased approaches for non-topical image categorization. *Internat. J. Dig. Libr.*, 3(3):261–275, 2000. [cited at p. 33]

[51] R. Schapire. Strength of weak learnability. *Machine Learning*, 5:197–227, 1990. [cited at p. 4]

[52] R. E. Schapire and Y. Singer. Boostexter: a boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000. [cited at p. 33]

[53] P. Simeone, C. Marrocco, and F. Tortorella. Design of reject rules for ecoc classification systems. *Pattern Recognition*, 45(2):863–875, 2012. [cited at p. 42]

[54] A. Srivastava and B. Zane-Ulman. Discovering recurring anomalies in text reports regarding complex space systems. *In IEEE Aerospace Conference*, 2005. [cited at p. 37]

[55] Hastie T. and Tibshirani R. Classification by pairwise grouping. *The Annals of Stat.*, 26(5):451–471, 1998. [cited at p. 7]

[56] Theeramunkong T. and Sriphaew K. Discovery of relations among scientific articles using association rule mining. *Proceedings of the 2007 NSTDA Annual Conference Science*, 2007. [cited at p. 30]

[57] Songbo Tan, Gaowei Wu, and Xueqi Cheng. Enhancing the performance of centroid classifier by ecoc and model refinement. *ECML/PKDD (2)*, pages 458–472, 2009. [cited at p. 34]

[58] Cover T.M. and Hart P.E. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13 (1):21–27, 1967. [cited at p. 4]

[59] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13, 2007. [cited at p. 33]

[60] Utschick W. and Weichselberger W. Stochastic organization of output codes in multiclass learning problems. *Neural Computation*, 13(5):1065–1102, 2001. [cited at p. 8]

[61] Freund Y. and Schapire R.E. A decision-theoretic generalisation of on-line learning and an application to boosting. *J. of Computer and System Science*, 55(1):119–139, 1997. [cited at p. 3, 4, 15]

[62] J. Zhou, H. Peng, and C. Y. Suen. Data-driven decomposition for multi-class classification. *Pattern Recognition*, 41(1):67–76, 2008. [cited at p. 8, 32]

# List of Publications Related to the Thesis

- G. Armano, N. Hatami, "Random Prototype-based Oracle for selection-fusion ensembles". 20th International Conference on Pattern Recognition, ICPR 2010, August 23-26, 2010, Istanbul. (Relation to Chapter 2)
- G. Armano, N. Hatami. "An improved mixture of experts model: Divide and conquer using random prototypes". In: O. Okun, G. Valentini, and M. Re (eds.) Ensembles in Machine Learning Applications, Studies in Computational Intelligence, Vol 373, pp. 217-232. Springer-Verlag, Berlin/Heidelberg, 2011. (Relation to Chapter 3)
- G. Armano, N. Hatami, "Run-time Performance Analysis of the Mixture of Experts Model", Computer Recognition Systems 4, Advances in Intelligent and Soft Computing, 2011, Volume 95/2011, 167-175. (Relation to Chapter 3)
- G. Armano, N. Hatami, "Hierarchical Mixture of Random Prototype-based Experts", ECML PKDD 2010, SUEMA workshop, Barcelona. (Relation to Chapter 3)
- G. Armano and N. Hatami, "Mixture of Random Prototype-based local Experts". LNCS series, International Conference on Hybrid Artificial Intelligence Systems, HAIS 2010, San Sebastian. (Relation to Chapter 3)
- N. Hatami, "Thinned-ECOC ensemble based on sequential code shrinking". Expert Systems with Applications, 39 (2012) 936-947. (Relation to Chapter 1 and Chapter 4)
- G. Armano, N. Hatami, C. Chira, "Error-Correcting Output Codes for Multi-Label Text Categorization", 3rd Italian Information Retrieval (IIR) workshop, in press. (Relation to Chapter 4)
- G. Armano, N. Hatami, C. Chira, "Ensemble of Binary Learners for Reliable Text Categorization", LNCS series, International Conference on Hybrid Artificial Intelligence Systems, HAIS 2012, in press. (Relation to Chapter 4)