LI-MLC: A Label Inference Methodology for Addressing High Dimensionality in the Label Space for Multilabel Classification

Francisco Charte, Antonio J. Rivera, María J. del Jesus, and Francisco Herrera, *Member, IEEE*

***Abstract*— Multilabel classification (MLC) has generated considerable research interest in recent years, as a technique that can be applied to many real-world scenarios. To process them with binary or multiclass classifiers, methods for transforming multilabel data sets (MLDs) have been proposed, as well as adapted algorithms able to work with this type of data sets. However, until now, few studies have addressed the problem of how to deal with MLDs having a large number of labels. This characteristic can be defined as high dimensionality in the label space (output attributes), in contrast to the traditional high dimensionality problem, which is usually focused on the feature space (by means of feature selection) or sample space (by means of instance selection). The purpose of this paper is to analyze dimensionality in the label space in MLDs, and to present a transformation methodology based on the use of association rules to discover label dependencies. These dependencies are used to reduce the label space, to ease the work of any MLC algorithm, and to infer the deleted labels in a final postprocessing stage. The proposed process is validated in an extensive experimentation with several MLDs and classification algorithms, resulting in a statistically significant improvement of performance in some cases, as will be shown.**

***Index Terms*— Association rules (ARs), data transformation, dimensionality reduction, multilabel classification (MLC).**

I. Introduction

In a traditional classification data set, each data instance is associated with one, and only one, class (label). By contrast, in a multilabel data set (MLD), every data sample has a set of labels associated with it, and therefore the classifier has to predict multiple outputs. There are MLDs [1] with several hundreds of labels, but many of the proposed approaches to multilabel classification (MLC) reduce this problem transforming the MLD to obtain one or more single label data sets.

High dimensionality is present in data used every day and establishes an important obstacle in many areas, among them information retrieval [2], natural language processing [3], and machine learning [4]. The problem of high dimensionality in classification tasks is well studied. However, published work has been mainly restricted to feature and sample spaces. In MLC, we find the same problem in a new space: the label space. How high dimensionality in the label space influences multilabel classifiers, and how to deal with this problem, is something poorly studied in the literature until now.

Feature selection algorithms [4], [5] have been used for several years to reduce dimensionality in the input attribute space, and there are some specific proposals [6] for MLC. These algorithms evaluate correlations between features and the class associated with each instance, as well as between one feature and another, deleting those which do not offer useful information to the task at hand: redundant features and features not correlated to the class. In the end, the classifier is trained with a reduced feature space that improves both learning time and classification results. The question arises of how this basic idea could be transferred to the label space in MLDs.

Data set characterization should be an important step to determine the suitability of an algorithm prior to its application. In MLC, the two main measures used with this goal (called Card and Dens, defined below) offer limited information, basically the average number of labels per instance in an MLD. It would be desirable to have additional information to know how the labels are distributed in the MLDs, as this fact will influence the MLC classifiers behavior.

The aim of this paper is to analyze the impact that high dimensionality in the label space has on the behavior of multilabel classifiers, to propose some measures for MLD characterization, and to present a methodology for label dimensionality reduction, called label inference for MLC (LI-MLC). It should be emphasized that LI-MLC is not a new MLC algorithm, but a method that acts as a wrapper around any existing multilabel classifier. As will be observed, some of the classification algorithms documented in the specialized literature use label dependency information internally to improve their functioning. LI-MLC generalizes this technique, so that it could be used regardless of the underlying classification algorithm.

Manuscript received February 20, 2013; revised October 13, 2013; accepted December 21, 2013. Date of publication January 9, 2014; date of current version September 15, 2014. The work of F. Charte was supported by the Spanish Ministry of Education through the F.P.U. National Program under Grant AP2010-0068. This work was supported in part by the Spanish Ministry of Science and Technology, FEDER founds, under Project TIN2012-33856 and Project TIN2011-28488, and in part by the Andalusian Research Plan, FEDER founds, under Project TIC-3928 and Project P10-TIC-6858.

F. Charte is with the Department of Computer Science and Artificial Intelligence, University of Granada, Granada 18071, Spain (e-mail: francisco@fcharte.com).

A. J. Rivera and M. J. del Jesus are with the Department of Computing Science, University of Jaén, Jaén 23071, Spain (e-mail: arivera@ujaen.es; mjjesus@ujaen.es).

F. Herrera is with the Department of Computer Science and Artificial Intelligence, University of Granada, Granada 18071, Spain, and also with the Faculty of Computing and Information Technology-North Jeddah, King Abdulaziz University, Jeddah 21589, Saudi Arabia (e-mail: herrera@decsai.ugr.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2013.2296501

LI-MLC is a pre-/post-processing methodology designed to reduce label dimensionality in MLDs. It can be used along with any MLC algorithm and, as will be shown, it can improve execution performance as well as classification results. In the preprocessing phase, LI-MLC works with the labels associated with each sample, applying an association rule (AR) mining algorithm to obtain a set of strong ARs. These rules are used to reduce the dimensionality of the label space, obtaining as a result a simplified training partition. This is later used to build the classifier. Finally, in the postprocessing stage, the same set of rules allows the inference of the labels to be added to the prediction made by the multilabel classifier.

The benefits of this approach will be demonstrated with an extensive experimentation, using 16 MLDs from several domains, and comparing the results obtained with seven MLC algorithms. The improvements are substantial in some cases, in both classification performance and speed, as will be shown.

This paper is organized as follows. Section II introduces the MLC problem and offers an analysis of the difficulties produced by high dimensionality in the label space in this kind of learning task. Section III details the proposed methodology and describes its relation with other published proposals. Section IV introduces some measures to characterize the label distribution in MLDs. In Section V, the experimental framework used is described, whereas Section VI analyzes the results obtained from this experimentation. Section VII provides the final conclusion.

II. Multilabel Classification

Classification is one of the most important tasks using supervised learning. The process starts with a set of labeled samples (x_i, y_i), and obtains a model f that is capable of labeling new samples not observed during the learning phase. Traditionally, classifiers are designed with data sets in which each sample x_i is associated with one class or label $y_i \in L$, which is the target value to obtain once the model is built. Therefore, the goal is to associate each sample to a class among $|L|$ possible classes, so that the range of possible output values is limited by the number of existing classes.

In MLC [7], the output returned by the classifier has to be a set of labels $Y_i \subseteq L$. Thus, there are $2^{|L|}$ different possible values as output: these can be any combination of labels in L. As stated in [8], this prediction can be generated in one of two ways: 1) with a binary partition of the label set or 2) with a label ranking.

The traditional classification algorithms cannot be used directly, as such, to tackle a problem of MLC. Reference [8] proposes two different ways to deal with this problem.

1) *The data transformation approach:* Its goal is to transform the data set, making it possible to process it using non-MLC algorithms.
2) *The method adaptation approach:* Its goal is to adapt a traditional classification algorithm, adding the ability to deal with samples, which are associated with multiple labels.

Sections II-A and II-B offer a brief introduction to each approach. It is also important to know some specific measures used in MLC, described in Section II-C. The problems that the presence of high dimensionality add to this task are discussed in Section II-D. Finally, some of the available options for obtaining label dependency information are enumerated in Section II-E.

A. Data Transformation Approach

While many data transformation-based methods have been proposed (a complete taxonomy can be found in [9]), binary relevance (BR) and label powerset (LP) are the most important ones. These methods are algorithm independent and also known as problem transformation methods.

The BR [10] transformation divides an MLD into multiple binary data sets. An independent binary classifier is trained for each label. Therefore, there will be as many binary classifiers as labels there are in the original data set. BR dismisses the relationship between labels. It also implies a linear increase in execution time by the total number of labels.

There are transformation methods based on pairwise comparison. Taking all the possible pairs of labels, each binary classifier works with one of them. The predictions made by all the classifiers are combined with a voting algorithm. Some proposals, such as CLR [11], use an improved voting mechanism to prevent evaluation of all label pairs.

The LP [12] method transforms an MLD into a multiclass data set, in which each data instance is associated with only one class, allowing the use of any multiclass classification algorithm. This is done interpreting as class each different combination of labels, which appears in the MLD. The main problem with this method is that the number of combinations of labels is $2^{|L|}$, so the amount of classes could become intractable.

An important side effect of these transformation methods is the extreme imbalance problem, which its use generates. For BR, each individual classifier is trained considering only the samples with a particular label associated as positive, and all others as negative. Usually, the number of negative samples is much larger than the number of positive, and therefore there is a great imbalance ratio. LP increases significantly the number of different classes in the data set given to the classifier. The more classes exist, the fewer samples per class, and the greater the imbalance.

B. Method Adaptation Approach

The transformation methods described above allow us to address the MLC problem using algorithms that are not designed for the specificities of the task. Faced with this choice, the focus of the algorithm adaptation approach [8] aims to modify existing algorithms so that they can deal with MLDs, without requiring any preprocessing. In recent years, the number of proposals published in this regard has increased strikingly, and they have been reviewed in [7], [8], and [13]. Only the more remarkable ones are listed here.

Clare and King [14] modified the C4.5 algorithm with two changes: each leaf of the tree stores not a class but a set of them, and the original entropy measure is adapted to consider the fact that the samples are multilabel. Another tree-based algorithm is proposed in [15].

There are several adaptations of instance-based algorithms. The most notable are ML-kNN [16] and IBLR-ML [17], the latter being a variation of the former.

The first adaptation of a neural network to MLC was BP-MLL [18], a perceptron with backpropagation learning, which introduces a modified error function that considers the multilabel nature of the samples. Another proposal in this field is the ML-RBF [19] algorithm for designing multilabel RBFNs.

There are also proposals based on support vector machines, such as rank SVM [20], as well as several methods based on the use of ensembles of classifiers with transformation of data, such as classifier chains (CCs) and ensembles of CCs (ECC) [21], random k-labelsets (RAkEL) [22], and hierarchy of multilabel classifiers (HOMER) [23], and even those based on ant colonies, like MuLAM [24]. The number of publications related to the adaptation of algorithms for MLC is constantly growing.

C. Evaluation Metrics

The peculiarities of the MLC problem require the use of new measures: 1) to characterize the MLDs and 2) to facilitate the evaluation of the MLC algorithms.

For the first objective, different metrics have been proposed, label cardinality (Card) and label density (Dens) being the two most widely used. Y_i being the subset of labels associated with the ith sample, Card is defined in (1) as the average number of labels per sample in the data set D. Dens, defined in (2), provides a measure independent of the absolute number of labels in the data set

$$\text{Card}(D) = \sum_{i=1}^{|D|} \frac{|Y_i|}{|D|}. \tag{1}$$

$$\text{Dens}(D) = \frac{\text{Card}(D)}{|L|}. \tag{2}$$

In terms of measures that assess the quality of predictions, they can be grouped: 1) by operating on a bipartition of the labels or on a ranking of these and 2) according to the calculation method: averaging by instance (example based) or label (label based). There are more than a dozen different measures that are explained in detail in [7] and [8]. The measures used in this paper will be introduced later, along with the experimental framework description.

D. Label Space Dimensionality Problem

The following is a discussion about how a high-dimensional label space influences different kinds of MLC algorithms.

1) *Adapted Algorithms:* With a few exceptions, like proposals based on the k-NN approach, adapted algorithms have to construct a model, which is a representation of correlations between input attributes and output labels. The higher the number of labels the greater the complexity of this model, with an increment in the time used to train it. Simpler models tend to be more efficient.
2) *Binary Classifiers:* BR and some adapted algorithms, such as CC and ECC among others, are based on the training of several binary classifiers, one per label. Thus, working with MLDs, which have hundreds of labels, it will be necessary to train the same number of classifiers, something very time—and memory—consuming. The higher the number of labels, the greater the likelihood that relationships between them exist, and generally this correlation is not considered by these kinds of MLC methods.
3) *Combinatorial Methods:* Combining active labels in each sample, and using the result as a class identifier, is an easy way to work with MLDs leaning in multiclass classifiers. It is an approach used by the LP transformation and some other MLC algorithms, but an unfeasible option when there are a large number of labels because of its exponential combinatorial generation of new classes. It also usually suffers from high dispersion, with a low number of instances associated with each class.
4) *Ensembles:* CC/ECC, RAkEL, and HOMER are examples of ensemble methods applied to MLC. Like them, many of the existing proposals work internally as a collection of binary classifiers or classifiers trained with subsets of label combinations. Therefore, they suffer from the same weakness cited above for BR and LP.

As a general rule, reducing the output space (number of labels) will also reduce the time and memory needed to train the classifier, and will generate simpler models, which usually work better. Based on this idea, in the literature, there are several proposals [25]–[27] (see discussion in Section III-C) whose goal is to compress the label space. The common approach in these proposals is to project the label space in a lower dimensionality space, transforming the initial MLC problem into another kind of task, such as regression or binary classification. Once this intermediate problem has been solved, it is necessary to invert the previous transformation to get the multilabel predictions. The process of obtaining this preimage of the projection creates a new problem that can be more difficult to confront than the original one.

An alternative to the transformation of the label space would be the selection of individual labels based on label dependency information. The correlation among labels has been used for different purposes, as detailed in the following, and it is the foundation of the proposed method for reducing label dimensionality described in Section III.

E. Label Dependency in MLC

Assuming an ideal real-world scenario, working with cleaned data, that two or more labels appear together very frequently suggests some levels of correlation among them. The certainty and strength of this dependency should be analyzed, as this information can be very useful for any MLC algorithm.

As stated in [7], one of the key challenges in multilabel learning is to reduce the output space exploiting correlations among labels. In this paper, the authors classify the strategies used by MLC algorithms to obtain these correlations into three categories: 1) first order; 2) second order; and 3) high order.

The order depends on the number of labels whose dependency is analyzed, only one, two, or more than two, respectively. In the following, we propose another way of grouping the classifiers, by means of the method they use to obtain the dependency information.

Some MLC methods, such as BR, completely overlook the presence of label dependency information, as they train independent classifiers for each existent label. Others, such as LP, explicitly incorporate this information using each different label combination as a class identifier. However, there are more sophisticated proposals in this field using various approaches to capture label dependencies, among them.

1) *Implicitly:* There are several methods, such as CC, ECC [21], 2BR [28], and BR+ [29], which extend the feature space of each binary classifier using as new input attributes the outputs of the other classifiers, through the composition of CCs or classifier stacking, thus implicitly using information about the relations between labels.
2) *Statistical Models:* Proposals made in [30]–[32] resort to statistical models, such as bayesian networks to explicitly represent dependencies among labels, or chi-square tests of independence to assign a dependence score to each pair of labels.
3) *Clustering:* Another way to collect label dependency consists in clustering the instances and obtaining information on the labels, which appear in each group. This is an approach used in HOMER [23].
4) *Others:* In addition to the previous groups, there are other approaches for extracting label dependency information. RAkEL [22] uses an ensemble of classifiers trained with a small random subset of labels, using LP to capture label dependencies. PLST [33] uses a geometric solution, using projections of a hypercube to smaller dimensions in order to obtain label correlations. Park and Fürnkranz [34] use ARs with the goal of defining classification restrictions based on label dependency.

Some of these approaches use the label dependency knowledge internally to enhance the learning process, others to divide the training data and construct ensembles, and another to impose restrictions over the classification results. None of them aim to reduce the label space from the beginning, prior to the training phase.

The effectiveness of label space reduction methods [25]–[27] and the usefulness of label-dependence information [21], [30]–[32] have both been demonstrated, therefore they are two proven techniques. Our hypothesis is that label dependency information can be used to remove some labels, reducing the label space without relying on compression/projection schemes. Thus, what we propose in LI-MLC is to combine these two techniques to face the label space high-dimensionality problem, relying on an AR mining algorithm as a tool to obtain strong correlations.

III. Label Inference for MLC

LI-MLC aims to reduce the number of labels of an MLD, but considering that they are a part of the output that the multilabel classifier has to predict. A method for analyzing strong correlations between labels, able to infer one label from the presence of others, is needed. Using this method, the labels that can be inferred will not participate in the training stage. Using an inference mechanism, removed labels will be added to predictions made by the multilabel classifier. If redundant labels are stripped from the data set, the training will need less time, and the classifier obtained will be simpler and, probably, more effective and efficient.

LI-MLC works over the label space of an MLD, using an AR mining algorithm to obtain a set of ARs. These are first used to reduce label dimensionality, and then, once the classifier has been obtained, to infer the labels to be added to the final results.

A. Retrieving Label Dependency Using ARs

Our interest is in obtaining subsets of labels with strong correlations, keeping in mind that even nonfrequent labels with a high correlation could be useful. AR mining algorithms [35] use various techniques, such as frequent itemset generation, which prunes the search space thus allowing the retrieval of correlations in limited time. We have to tune the ARs mining algorithm in two ways, choosing a rule selection measure and establishing its threshold value, as well as the minimum support (Supp).

The most common measure in ARs mining is confidence (Conf) and its joint use with Supp is known as the confidence–support framework [36]. Supp(Z) is defined (3) as the proportion of transactions in a database that contains a specific itemset Z. The confidence of a rule with antecedent X and consequent Y, $\mathrm{Conf}(X \rightarrow Y)$, is defined (4) as the probability of the consequent presence under the condition that the antecedent is also present in the transaction

$$\mathrm{Supp}(Z) = P(Z). \tag{3}$$

$$\mathrm{Conf}(X \rightarrow Y) = \frac{\mathrm{Supp}(X \rightarrow Y)}{\mathrm{Supp}(X)}. \tag{4}$$

As stated in the discussion of implication rules in [37], a high value of Conf in some cases could appear without the existence of an implication from the antecedent to the consequent, and this is a weakness with respect to the goal of this paper. There are several other measures for evaluating ARs, among them conviction (Conv) [37] (5) that better fits the aims in this proposal, as it is a directed measure, which considers the information on the absence of the consequent. A high value in this measure means, no matter what the Supp of the AR, that an implication between antecedent and consequent exists. Unlike other measures, Conv is sensitive to rule direction, and antecedent and consequent cannot be interchanged without changing the result obtained from the measure. For this reason, this measure was chosen to filter the ARs generated by the algorithm

$$\mathrm{Conv}(X \rightarrow Y) = \frac{1 - \mathrm{Supp}(Y)}{1 - \mathrm{Conf}(X \rightarrow Y)}. \tag{5}$$

With regard to which ARs mining algorithm to use, in [35], there is a review of the most common ones, many of them based on the best known: the *a priori* algorithm. As stated

in [38], FP-Growth is a better AR mining algorithm for working with large databases than *a priori*. The number of labels and samples in an MLD could be very large, and therefore it is important to choose an efficient mining algorithm.

The details of the methodology and its implementation are covered in Section III-B. To know when this methodology will be useful, it is important to understand how the labels are distributed in an MLD, performing a characterization, as discussed in Section IV.

B. Description of the Proposed Methodology

LI-MLC works in three stages: 1) preprocessing; 2) classifier training; and 3) postprocessing. They are clearly shown in Fig. 1. The goal of each phase is as follows.

1) *Preprocessing Phase:* Takes as input the training data of an MLD, and produces a reduced version of the training partition, with fewer labels, and a set of ARs in the form expressed in (6).
2) *Classifier Training:* The reduced training partition is then used to train any multilabel classifier.
3) *Postprocessing Phase:* Once the prediction made by the multilabel classifier has been obtained, the evaluation of the ARs adds the inferred labels to produce the final prediction

$$L_i, \ldots, L_j \rightarrow L_k, \ldots, L_m \quad L_x \in L. \tag{6}$$

In the preprocessing phase, LI-MLC takes each label as an item in a transaction, and the set of labels associated with each sample is interpreted as a transaction. The instances X_i of the training partition are processed (lines 11–14) to extract their label sets L_i, using each one of them as a transaction. The database T composed by those transactions is given to the FP-Growth algorithm, obtaining a set R of ARs, which is sorted from higher to lower Conv to apply the stronger rules first.

With the set R of ARs at its disposal, LI-MLC collects the labels that appear in the consequent of each rule obtaining the set of labels to delete. As the same label could appear in the consequent of more than one rule, the cardinality of the set of labels could be lower than that of the set of rules. In this situation, only the rule with the highest Conv will be used, the rest are discarded (line 20) from R. In addition, it is not possible to delete a label if it is the only one to appear in some instances of the data set, something that will reduce the number of labels to eliminate (lines 21–23) in some cases.

Taking the final set C of labels to delete as reference, the method generates a version of the training partition without them (line 27). This is the reduced training set, with fewer output attributes than the original one, given as input to the selected MLC algorithms. The complexity of this process depends on the ARs mining algorithm used, but it is important to notice that preprocessing is needed only once per partition since the ARs mining algorithm used is deterministic. Thus, having generated the reduced training partition, it could be used to train any classifier.

The postprocessing obtains the predictions P_i made by the multilabel classifier for each sample (line 33). This prediction

```
1: procedure LI-MLC(X)                          ▷ Dataset to process
2:     DTra ← trainingPartition(X)
3:     DTst ← testPartition(X)
4:     DTra', Rules ← preprocessing(DTra)
5:     Classifier ← obtainClassifier(DTra')
6:     P ← postprocessing(Classifier, DTst, Rules)
7:     Evaluate(P)
8: end procedure

9: procedure PREPROCESSING(DTra)
10:    T ←Ø                                     ▷ Set of transactions
11:    for each instance X_i in DTra do
12:        L_i ← labelsOf(X_i)
13:        T ← T ∪ L_i
14:    end for
15:    R ← FPGrowth(T)
16:    R ← orderByConviction(R)
17:    ▷ Clean rules that are not applicable
18:    for each R_i in R do          ▷ Higher to lower conviction
19:        LC_i ← labelInConsequent(R_i)
20:        deleteRemainRulesWhoseConseqHas(LC_i)
21:        if isOnlyLabelInSomeSample(LC_i) then
22:            deleteRule(R_i)
23:        end if
24:    end for
25:    ▷ R has the final set of rules to apply
26:    C ← labelsInConsequent(R)
27:    DTra ← DTra − C          ▷ Reduced training partition
28:    return DTra, R
29: end procedure

30: procedure POSTPROCESSING(Classifier, DTst, R)
31:    P ←Ø                                     ▷ Set of predictions
32:    for each instance X_i in DTst do
33:        P_i ← Classifier(X_i)
34:        for each R_i in R do
35:            P_i ← P_i ∪ applyRule(R_i)
36:        end for
37:    end for
38:    return P
39: end procedure
```

Fig. 1. LI-MLC pseudocode.

is incomplete: the final prediction must reflect the strong label dependencies extracted previously. To do so, LI-MLC evaluates the set R of rules obtained in the preprocessing stage. The labels present in the partial prediction P_i will determine the antecedent of the rules in R to be applied, adding the labels represented by each consequent. This is an iterative process (lines 34–36) in which the rules are taken in the same order used in the preprocessing phase, activating the label of the consequent in each step if the current sample has activated the labels of the antecedent of the rule.

In this way, the final prediction is generated and given as the result. The complexity of this process is linear with respect to the number of rules, always far smaller than the number of samples or input attributes.

C. Relationship With Other Label Space Dimensionality Reduction Proposals

The need to reduce the label space dimensionality has been approached in several different ways in the literature in the past. In the following, there is a brief analysis of the relationship between LI-MLC with some of these proposals.

1) Label Subset Selection: The decomposition of the set of labels in several smaller ones is an approach used by algorithms, such as RAkEL [22] and HOMER [23]. Both use subsets of labels to train several multiclass classifiers, by means of the LP transformation. RAkEL picks the subsets randomly, whereas HOMER uses a hierarchical clustering to generate subsets of correlated labels. In the end, all labels are used in training, as the label space is not reduced but divided in groups. The inner classifier used by these algorithms is a multiclass classifier, not a multilabel classifier. LI-MLC reduces the label space before the classifier starts to work, thus there will be labels that do not participate in the process. Furthermore, LI-MLC does not transform the original problem in a different one, and does not imposes the use of a specific type of classifier. Any multilabel classifier can be used over the reduced data.

2) Pruning of Infrequent Label Sets: Read *et al.* [39] propose a method based on the LP transformation. The label space is reduced by pruning those label sets, which do not appear above a particular threshold. These label sets are later decomposed in simpler ones, with fewer labels, reintroducing those subsets, which exceed this threshold. This way only the most important relationships among labels are implicitly captured, improving the generalization ability in classification. An ensemble of pruned sets (EPS) is also proposed in the same paper. As with RAkEL and HOMER, the underlying classifier is a multiclass classifier, not a multilabel classifier. Therefore, it is not a method generally applicable independent of the problem faced or the classification algorithm chosen, as is LI-MLC. In addition, LI-MLC obtains the label dependencies by means of ARs mining, measuring the correlations among them, whereas pruned sets and EPS obtain this information implicitly.

3) Kernel Dependency Estimation: The technique proposed in [25] is not specifically designed to reduce the label space of an MLD, but as a general way of finding dependencies between a set of inputs and a set of outputs, MLC being one of its possible applications. By means of a kernel principal component analysis of the label space, a set of uncorrelated projections is obtained. In addition, the number of outputs can be reduced in this phase selecting only the most significant representations (those with larger eigenvalues). The mappings between the inputs and the representations of the outputs can be learned independently, for example, using regression. In the final step, with the independent predictions obtained from the regressors, a function is applied to find the preimage of the projection and get the final set of outputs. Several functions can be used for this task, depending on the specific application at hand. For classification, the authors opted for finding the solution among a set of candidates acquired from the training set. Our approach with LI-MLC is totally different. The original label space is used to learn the correlations between sets of labels, but it is not transformed to get independent representations. Only the labels that can be inferred with a specific level of confidence are eliminated, giving as a result a new MLD. Therefore, the classification problem is still a multilabel one, and thus can be faced with any multilabel classifier. Most importantly, in the final phase, LI-MLC obtains a multilabel prediction, which only needs to be complemented with labels inferred from the rules. There is no need to find the preimage of a projection.

4) Multilabel Prediction Via Compressed Sensing: The proposal made in [26] is based on a proven compression technique called compressed sensing (CS), which states that the complexity of a model with k labels can be reduced to the training of $\mathcal{O}[\log(k)]$ simpler models. There is a premise to accomplish: a significant level of sparsity in the label space must exist. Thus, it is a useful approach for MLDs that have a very large number of different labels, but with only a small subset of them appearing in each instance. This is the nature of the two data sets used in their experimentation. The procedure followed is similar to that described above for kernel dependency estimation. The compression phase is made by random projections of the original binary label space, obtaining a representation in a real (nonbinary) lower dimensionality space. Afterward, these projections are used to train a set of regression models. Finally, the classification is made using this set of regressors, and their outputs are decompressed to obtain the labels predicted for each sample. The only similarity between this approach and LI-MLC is founded on the existence of a preprocessing phase, which reduces the output space and a postprocessing phase in charge of its reconstruction. LI-MLC can be applied without assuming a sparse label space. If the label space is very sparse the extraction of ARs can be more difficult (see Section VI). Furthermore, the proposal in [26] is a complete MLC algorithm based on compression and regression, whereas LI-MLC, as has been said above, works as a wrapper around any multilabel classifier, which reduces the label space at the input and complements the prediction made by this classifier at the output. Thus, LI-MLC has a broader field of applications.

5) Compressed Labeling on Distilled Label Sets: In addition, based on the idea of CS, Zhou *et al.* [27] propose a method that combines a variation for compressing the label space while still using classification algorithms to make the predictions, instead of relying on regression. There are two key elements in this proposal: 1) the method to extract the most frequent subsets of labels in the MLD, the so called distilled label sets (DLs), and the transformation of the real lower dimensionality space obtained by random projections in a binary one. The latter is accomplished using the signs of the random projections. The former applying a recursive clustering approach over the label space. Once the compressed label space has been obtained, a binary classifier is used to predict each label independently. Those predictions are then complemented by means of the information of correlation stored in the DLs. The steps followed by compressed labeling (CL) are the same as LI-MLC. First, information about label dependence is obtained. CL relies on the extraction of

DLs, whereas LI-MLC obtains a set of ARs. Second, the label space is compressed. In CL, this step is done by random projections, while in LI-MLC, it is accomplished deleting the labels that can be inferred from the rules previously obtained. Third, the instances with the reduced label space are handed to a classification algorithm order to obtain the initial predictions. CL can use any binary classifier for this task, retrieving independent predictions for each label. LI-MLC can use any multilabel classifier, retrieving joint predictions. Finally, these initial predictions are completed and the final label set for each instance is returned. In CL, this process is supported by the DLs and the application of a statistical test, whereas in LI-MLC, it is by the inference of the ARs.

One of the main differences between LI-MLC and the proposals enumerated above lies in that they transform the MLC problem into another type of problem: multiclass classification, regression, or binary classification, whereas LI-MLC preserves its original multilabel nature. In addition, LI-MLC does not rely on a decomposition of the global goal in several simpler ones that have to be trained individually, and their independent outputs later recombined. Once the ARs have been extracted, a process that takes a very few seconds, only one classifier has to be trained. The outputs of this classifier are complemented with the inference of the ARs, without the need of merging outputs from several predictors nor finding a preimage by means of some decompression algorithm.

IV. Characterization of Label Distribution in MLDs

The proposed methodology will be useful if the labels in the MLDs are not distributed in a very sparse way. Obviously, if the Card measure is very low, it will be almost impossible to obtain correlation between labels, since the number of samples with two or more labels will not be enough. The Card measure alone, however, will not give the information needed to know if there are a few or many samples with two or more labels. This measure is a mean of the number of labels in the whole data set, without any information on how these labels are distributed among the samples. Neither will the Dens measure be useful, because this is simply Card divided by the total number of labels. It is necessary to characterize the data sets using additional measures.

The dispersion of labels in an MLD can be known calculating the coefficient of variation (CV), as shown in (7), σ being the standard deviation and μ the mean. This is a measure commonly used in imbalanced problems [40]. If Card is not very low (there is more than one label per sample) and CV is quite low (the labels are distributed uniformly), the likelihood of finding correlations will be higher. With large CV values (high dispersion), it could be difficult to obtain rules, even when Card is also high

$$\mathrm{CV} = \frac{\sigma}{\mu}. \tag{7}$$

Another possibility for finding out the distribution of labels is relying on two classical statistical measures [41]: 1) kurtosis (KR) (8) and 2) skewness (SK) (9). In this context, x_i must be interpreted as a data sample. The former is an indication of the level of concentration of the distribution around a specific value, while the latter points out where this value is. The higher KR is the more peaked is the distribution of the variable and, in this case, the more condensed will be the number of labels per sample around the same value. On the other hand, SK is a measure of asymmetry and can be positive or negative. When the distribution is peaked (high KR) a positive SK denotes that the bulk of the values are below the mean (lie to the left). By contrast, a negative value implies the opposite, and most of the values are above the mean (lie to the right). An SK near zero suggests a symmetrical distribution

$$\mathrm{KR} = E\left(\frac{x_i - \mu}{\sigma}\right)^4 - 3. \tag{8}$$

$$\mathrm{SK} = E\left(\frac{x_i - \mu}{\sigma}\right)^3. \tag{9}$$

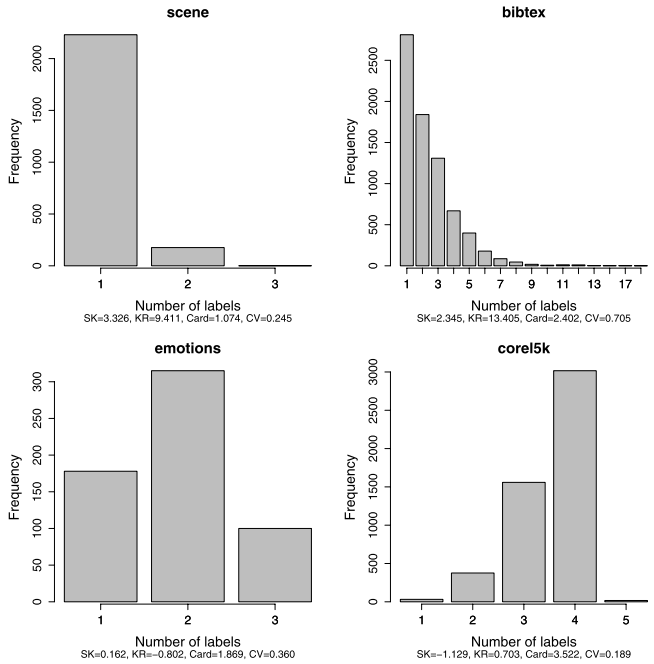

Fig. 2. Label distribution per instance (histogram).

In Fig. 2, the label distribution in four MLDs can be observed. Two of them, scene and bibtex, have a high value of KR and their SK is well above zero, so most of the samples in these data sets have only one label associated. For emotions and corel5k, the value of KR is near zero, meaning a more uniform distribution. In the former, SK is also almost zero: the bulk of the samples have the number of labels shown by the Card measure. The latter shows a negative SK, and therefore the number of labels per instance is above the mean for many samples. It is easy to see that obtaining correlations between labels will be more difficult for scene and bibtex than for emotions and corel5k.

Another way to see the distribution of labels per instance in a data set would be that shown in Fig. 3. Each row (y-axis) is a sample and the columns (x-axis) represent the different labels. The presence of one label in a certain sample is denoted as a line in the crossing of both axes. Fig. 3(a) and (d) come from

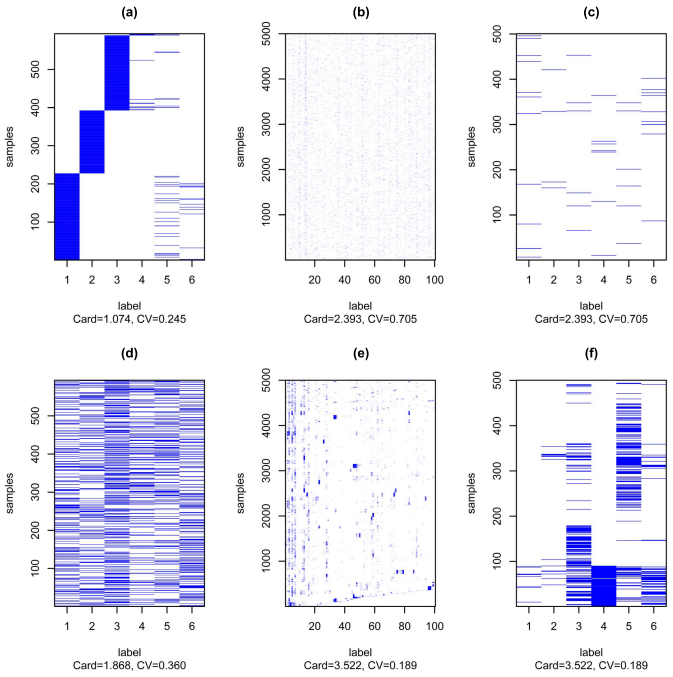

Fig. 3. Label distribution per instance (map). (a) Scene. (b) Bibtex. (c) Bibtex. (d) Emotions. (e) Corel5k. (f) Corel5k.

very similar data sets, with the same total number of labels and almost the same number of samples, but with a slightly higher Card value for Fig. 3(d). This is enough to obtain many samples with two or more labels, while in Fig. 3(a), it is clear that each sample has only one label with a few exceptions. Fig. 3(b) and (e) show two data sets with higher numbers of samples, labels and Card. The main difference between them is the dispersion: $CV = 0.189$ for Fig. 3(d) against $CV = 0.705$ for Fig. 3(b). The effect of this factor can be observed better in Fig. 3(c) and (f), magnifications of the previous ones.

The conclusion that could be obtained from this analysis is that the combined use of measures, such as Card, CV, SK, and KR, offer useful information to characterize an MLD, knowing how difficult it would be to work with it. In general, an extremely low Card (close to 1.0) will show that most of samples have only one associated label, and therefore it would be almost impossible to extract label dependency information. With Card values well above 1.0, the presence of a high CV, SK, or KR may denote a difficult MLD.

V. Experimental Framework

LI-MLC has been tested with 16 MLDs and seven MLC algorithms. Data sets and algorithms are detailed in the following parts of this section.

The mining of ARs was performed with the FP-Growth algorithm, which needs two parameters: the threshold values for Conv and Supp. Conv threshold was set to a minimum value of 1.25 as recommended in [37]. To obtain enough ARs to work with, we opted to apply a relatively low minimum threshold value for Supp, 0.025, with the hypothesis that even nonfrequent but strong associations between labels would be meaningful [42].

TABLE I
Data Set Characteristics

Dataset	Instances	Attributes	Labels	Card
bibtex	7395	1836	159	2.402
bookmarks	87856	2150	208	2.028
cal500	502	68	174	26.044
corel5k	5000	499	374	3.522
corel16k[a]	13766	500	161	2.867
delicious	16105	500	983	19.02
emotions	593	72	6	1.868
enron	1702	753	53	3.378
genbase	662	1186	27	1.252
imdb[b]	12000	1001	28	1.905
llog	1460	1004	75	1.180
mediamill	43907	120	101	4.376
medical	978	1449	45	1.245
scene	2407	294	6	1.074
slashdot	3782	1079	22	1.181
yeast	2417	198	14	4.237

[a]Corel16k is a dataset with 10 subsets. Average values are shown here

[b]Taking a random 10% of the original 120000 instances.

Cross validation was used with the usual configuration at 10 partitions. The same configuration was used to run each MLC algorithm without applying LI-MLC.

In Section V-A, the MLDs used in the experimentation are enumerated. The MLC algorithms and the parameters used are described in Section V-B. Sections V-C and V-D detail the performance measures and statistical tests applied.

A. Data Sets

Table I shows the data sets and their main characteristics: number of instances, attributes and labels, and Card. Seven of them (bibtex, bookmarks [43], delicious [23], enron [44], llog, medical [45], and slashdot) are from the text domain, three (corel5k [46], corel16k [47], and scene [12]) are from the image domain, two (cal500 [48] and emotions [49]) are from the music domain, two more (genbase [50] and yeast [20]) from the biology domain, and the last two (imdb and mediamill [51]) are from the video/movie domain. The number of labels in those MLDs is in the range [6, 983] and Card is in the range [1.074, 26.044]. All data sets can be obtained from MULAN [1] and MEKA [52] repositories.

B. Algorithms

A representation of MLC algorithms based on data transformation, method adaptation, and ensemble approaches was selected. It should be noted that the transformation methods do their work after the preprocessing phase of LI-MLC described previously, operating on a reduced data set having already eliminated the labels inferred from the rules.

The methods chosen, all used with default parameters, were the following: BR [10], LP [12], CLR [11], CC [21], IBLR-ML [17], RAkEL [22], and HOMER [23]. C4.5 was used as underlying binary/multiclass classifier. For IBLR-ML, the number of neighbors was set to 10. For HOMER, the number of clusters was set to the minimum between four and the number of labels in the MLD.

C. Performance Measures

More than a dozen MLC performance measures have been defined in [7] and [8]. One of the most widely used is Hamming loss (HL). L being the full set of labels, Z_i the set of predicted labels, and Y_i is the set of real labels, HL (10) is defined as the fraction of committed errors in sample–label pairs prediction. The operator Δ represents the symmetric difference

$$\text{HL} = \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i \Delta Z_i|}{|L|}. \tag{10}$$

HL tends quickly to zero when it is used with some MLDs, those in which only a small subset of labels appear in each instance but the total set of labels is large. In this context, using the same classifier and making the same number of classification errors, the performance would seem worse as the total number of labels is smaller. However, the real performance of the classifier would be the same. Although HL is a popular MLC performance measure, the previous reason demands the use of another one, stronger than the former. In [27], an evaluation on the strength of some MLC evaluation measures is offered, showing that HL is the weakest and less representative, whereas F-measure is one of the strongest. F-measure has been chosen as second measure for this paper.

F-measure (11) is a balanced combination of precision (12) and recall (13). In this expressions, TP stands for true positives, FP for false positives, and FN for false negatives

$$\text{F-Measure} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{11}$$

$$\text{Precision} = \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i \cap Z_i|}{|Z_i|} = \frac{\text{TP}}{\text{TP}+\text{FP}}. \tag{12}$$

$$\text{Recall} = \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i \cap Z_i|}{|Y_i|} = \frac{\text{TP}}{\text{TP}+\text{FN}}. \tag{13}$$

D. Statistical Tests

To analyze whether the differences between obtained results are significant, it is usual to perform statistical tests. In this paper, the interest is in comparing values from the classification of several data sets in pairs: one set of results from the base MLC algorithm and a second set from the same algorithm using LI-MLC methodology. For this task, the paired T-test parametric test or the Wilcoxon nonparametric test could be applied.

References [53] and [54] have shown that classification experiments, which follow the 10-fold cross-validation scheme do not fulfill the necessary conditions of normality, heteroscedasticity, and independence to use parametric statistical tests. For this reason, to perform this kind of comparison, the Wilcoxon [55] nonparametric statistical test has been used in this paper.

For each algorithm, taking the performance measures of the version with LI-MLC as reference, this test was applied and the exact p-values obtained. The same method was also used to compare the training times of each classifier. The tests were executed using the statistical package of the data-mining software KEEL [56].

TABLE II
CHARACTERIZATION MEASURES

Dataset	Card	Dens	CV	SK	KR	#Rules
bibtex	2.402	0.015	0.704	2.353	13.384	0.0
bookmarks	2.028	0.010	0.904	4.197	36.084	0.0
cal500	26.044	0.150	0.221	0.518	0.398	6.4
corel5k	3.522	0.009	0.189	-1.129	0.703	5.0
corel16k	2.867	0.018	0.316	-0.344	-0.677	4.0
delicious	19.020	0.019	0.267	-1.206	0.988	8.0
emotions	1.868	0.311	0.360	0.162	-0.802	3.8
enron	3.378	0.064	0.454	0.652	1.062	13.8
genbase	1.252	0.046	0.555	3.501	14.597	5.7
imdb	1.905	0.068	0.630	1.669	3.492	3.0
llog	1.180	0.016	0.679	1.159	2.503	0.0
mediamill	4.376	0.043	0.533	0.440	0.291	8.0
medical	1.245	0.028	0.371	1.612	1.581	0.0
scene	1.074	0.179	0.245	3.326	9.411	0.0
slashdot	1.181	0.054	0.351	2.147	3.836	0.0
yeast	4.237	0.303	0.371	0.380	0.150	10.8

VI. EXPERIMENTAL STUDY AND DISCUSSION

In this section, the data sets used in the experimentation are characterized, the classification results are exposed, the statistical study is shown, and a final discussion of these results is offered.

A. Data Set Characterization

Using the measures proposed in Section IV, the analysis of the high dimensionality in the label space in the data sets used are offered in the following.

Table II shows the characterization measures Card, Dens, CV, SK, and KR, as well as the average number of rules obtained for each data set.

There are six cases: 1) bibtex; 2) bookmarks; 3) llog; 4) medical; 5) scene; and 6) slashdot in which no rules were obtained. Four of them (llog, medical, scene, and slashdot) have an extremely low Card, in the range [1.074, 1.245], which explains by itself the impossibility of obtaining any rules. bibtex and bookmarks have slightly larger Card values, 2.402 and 2.028, but also share quite high CV, SK, and KR values. This denotes that only a few instances contain two or more labels, while most of them have only one. By contrast, data sets, such as emotions, with a Card value of 1.868 but also low values in the other measures, allow the extraction of some rules.

The previous are some general guidelines, but there are also exceptions. For instance, the Card for medical is similar to genbase, and CV is higher for the latter denoting a larger dispersion, but in the former case, we had no rules, whereas in the latter, some rules have been obtained. genbase has almost half the total number of labels of medical. Although they have similar Card values, the number of label combinations is much bigger for medical (2^{45}) than for genbase (2^{27}).

B. Classification Results

Tables III and IV show the resulting HL and F-measure values for each algorithm–data set combination in two

TABLE III
CLASSIFIER PERFORMANCE IN TERMS OF HL (THE LOWER THE BETTER)

Dataset	CC	+LI-MLC	BR	+LI-MLC	CLR	+LI-MLC	HOMER	+LI-MLC	IBLR	+LI-MLC	LP	+LI-MLC	RAkEL	+LI-MLC
cal500	**0.1760**	**0.1760**	**0.1615**	0.1620	**0.1385**	0.1401	**0.1846**	0.1893	**0.2309**	0.2334	**0.1996**	0.2030	0.1615	**0.1611**
corel16k	0.0206	**0.0199**	0.0197	**0.0194**	0.0189	**0.0188**	0.0253	**0.0251**	**0.0191**	**0.0191**	0.0321	**0.0306**	0.0197	**0.0194**
corel5k	0.0099	**0.0098**	0.0098	**0.0097**	0.0095	**0.0094**	0.0132	**0.0128**	**0.0224**	0.0226	0.0168	**0.0162**	0.0098	**0.0097**
delicious	**0.0188**	0.0189	**0.0186**	0.0188	**0.0184**	0.0185	**0.0238**	0.0241	**0.0507**	0.0511	**0.0300**	**0.0300**	**0.0187**	**0.0187**
emotions	**0.2550**	0.2766	**0.2474**	0.2857	**0.2423**	0.3098	**0.2609**	0.2940	**0.1883**	0.2417	**0.2777**	0.2982	**0.2474**	0.2885
enron	**0.0524**	0.0546	**0.0508**	0.0553	**0.0471**	0.0525	**0.0584**	0.0623	**0.0564**	0.0593	0.0717	**0.0702**	**0.0508**	0.0551
genbase	**0.0011**	0.0055	**0.0011**	0.0058	**0.0013**	0.0059	0.0013	**0.0006**	**0.0029**	0.0078	**0.0019**	0.0061	**0.0011**	0.0058
imdb	0.0878	**0.0745**	0.0753	**0.0724**	0.0719	**0.0701**	0.0950	**0.0927**	0.0681	**0.0680**	0.1024	**0.0987**	0.0753	**0.0728**
mediamill	**0.0357**	0.0389	**0.0335**	0.0397	**0.0283**	0.0358	**0.0371**	0.0437	**0.0283**	0.0356	**0.0423**	0.0474	**0.0335**	0.0398
yeast	**0.2682**	0.2795	**0.2454**	0.2720	**0.2202**	0.2590	**0.2555**	0.2783	**0.1934**	0.2397	**0.2779**	0.2939	**0.2449**	0.2731

TABLE IV
CLASSIFIER PERFORMANCE IN TERMS OF F-MEASURE (THE HIGHER THE BETTER)

Dataset	CC	+LI-MLC	BR	+LI-MLC	CLR	+LI-MLC	HOMER	+LI-MLC	IBLR	+LI-MLC	LP	+LI-MLC	RAkEL	+LI-MLC
cal500	0.3626	**0.3627**	**0.3375**	0.3364	**0.2888**	0.2644	**0.3972**	0.3942	**0.3194**	0.3065	**0.3287**	0.3111	0.3375	**0.3393**
corel16k	0.5388	**0.5578**	0.5259	**0.5702**	0.5313	**0.5606**	0.4517	**0.4606**	0.5539	**0.6032**	0.4548	**0.4932**	0.5243	**0.5624**
corel5k	0.4859	**0.5125**	0.4697	**0.4884**	0.4511	**0.4812**	0.3894	**0.4013**	0.2876	**0.2889**	0.4031	**0.4305**	0.4697	**0.4921**
delicious	0.3290	**0.3421**	0.3097	**0.3198**	0.2549	**0.2763**	**0.3289**	0.3249	**0.1858**	0.1843	0.2599	**0.2602**	0.3056	**0.3176**
emotions	**0.7438**	0.7309	0.7086	**0.7234**	**0.7186**	0.6903	0.6949	**0.7123**	**0.7839**	0.7690	0.7244	**0.7291**	0.7086	**0.7283**
enron	**0.6268**	0.5977	0.6137	**0.6260**	0.6330	**0.6469**	0.5993	**0.6104**	0.5810	**0.6038**	0.5561	**0.5664**	0.6137	**0.6323**
genbase	**0.9919**	**0.9919**	**0.9919**	0.9915	**0.9914**	0.9905	**0.9917**	**0.9917**	0.9856	**0.9870**	**0.9919**	0.9915	**0.9919**	0.9918
imdb	0.7209	**0.7413**	0.6602	**0.7077**	0.6750	**0.7340**	0.5634	**0.5821**	0.7110	**0.7358**	0.6375	**0.6433**	0.6602	**0.6892**
mediamill	**0.6157**	0.5711	**0.6066**	0.5339	**0.6387**	0.5592	**0.5961**	0.5150	**0.6544**	0.5794	**0.1314**	0.0837	**0.6066**	0.5331
yeast	**0.6389**	0.6339	**0.6166**	0.5890	**0.6587**	0.5901	**0.6156**	0.5901	**0.6987**	0.6279	**0.6256**	0.6089	**0.6171**	0.5890

TABLE V
CLASSIFIER PERFORMANCE IN TERMS OF TRAINING TIME IN SECONDS (THE LOWER THE BETTER)

Dataset	CC	+LI-MLC	BR	+LI-MLC	CLR	+LI-MLC	HOMER	+LI-MLC	IBLR	+LI-MLC	LP	+LI-MLC	RAkEL	+LI-MLC
cal500	**148.68**	152.16	63.29	**54.79**	1257.52	**1154.58**	99.66	**93.16**	261.68	**233.29**	14.92	**15.09**	346.04	**328.47**
corel16k	19642.56	**19022.21**	39980.52	**26154.08**	73505.36	**62752.85**	12139.59	**11088.42**	20226.75	**20108.44**	1351.50	**1240.07**	201454.89	**152089.85**
corel5k	**3915.71**	3954.94	5053.38	**5013.73**	18411.18	**14369.86**	3934.75	**3575.49**	20742.53	**19817.02**	**336.88**	340.73	43754.60	**39703.89**
delicious	375022.66	**358317.20**	256064.79	**236233.64**	248904.86	**212829.72**	408823.09	**408405.82**	176880.45	**126634.57**	3538.60	**3531.68**	1457888.84	**1438587.70**
emotions	5.61	**2.93**	12.01	**3.16**	10.83	**4.06**	6.14	**3.07**	4.31	**3.65**	3.92	**2.64**	24.98	**2.52**
enron	**1512.23**	1623.53	2051.74	**1692.63**	4780.69	**3290.02**	2688.95	**1879.37**	143.97	**116.68**	**115.50**	125.39	13070.84	**10607.92**
genbase	7.68	**5.96**	17.07	**10.69**	26.31	**19.24**	12.00	**8.62**	60.24	**60.20**	4.67	**4.51**	63.93	**51.67**
imdb	**85817.48**	89018.88	**97036.83**	97962.34	158132.40	**141208.82**	37421.90	**34063.70**	**549.09**	558.92	**7967.42**	8416.61	**583141.02**	607884.85
mediamill	51554.52	**39843.20**	28622.91	**26448.25**	91883.01	**68983.46**	9129.91	**6910.66**	**21242.97**	21343.17	22968.71	**19839.37**	116597.17	**112656.82**
yeast	90.35	**78.20**	121.14	**86.01**	336.83	**202.17**	141.80	**100.01**	94.84	**94.25**	105.98	**81.13**	791.22	**588.73**

versions: the original multilabel classifier, and the same configuration applying the proposed methodology, noted as +LI-MLC. The six data sets for which it was not possible to obtain rules do not appear in this table, as LI-MLC is not applicable in these cases.

Assessing results in terms of HL, we found improvements in one-third of cases. There are several ties, as well as many cases in which the difference in either direction is minimal, in the range of a few ten thousandths. With delicious, for instance, LI-MLC loses in all cases but all differences are in the range [0.0001, 0.0004]. The significance of these differences is doubtful, but in any case, the improvement in execution time is very significant. The weaknesses of HL may be highlighted, exposed previously in Section V-C.

The use of F-measure to assess the results offer a quite different picture. In general, the differences are much bigger, in the scale of hundredths. The application of LI-MLC generates better results in 41 out of the 70 configurations. Classification performance is almost always improved for corel16k, corel5k, enron, and imdb. By contrast, the results are never improved for two data sets: 1) mediamill and 2) yeast. Mixed results are obtained for the rest of MLDs. Except for mediamill and yeast, LI-MLC has mainly a positive influence over the performance of the underlying classifiers. The statistical significance of these results is evaluated in the following section.

Table V uses the same structure as the previous ones, but exhibiting training times for each configuration, expressed in seconds. This is the time elapsed when training the classifier with the original base algorithm and with LI-MLC. In the latter case, the time spent obtaining the rules and preprocessing the data set has been added to the training time. As can be observed, for some algorithm–data set combinations, the training time has been reduced to a fraction of the original. The savings in some cases are in the order of several hours. Taking as reference the time of the base algorithm, Fig. 4 shows the relative gain in time achieved by the classifier training process once LI-MLC has been applied. It must be highlighted that this improvement in execution time, in general, is achieved without major damage to classification performance, but even with an amelioration in many cases. For instance, the use of LI-MLC with BR and RAkEL resulted in more than 22% and 15% of reduction of the execution time, while in classification

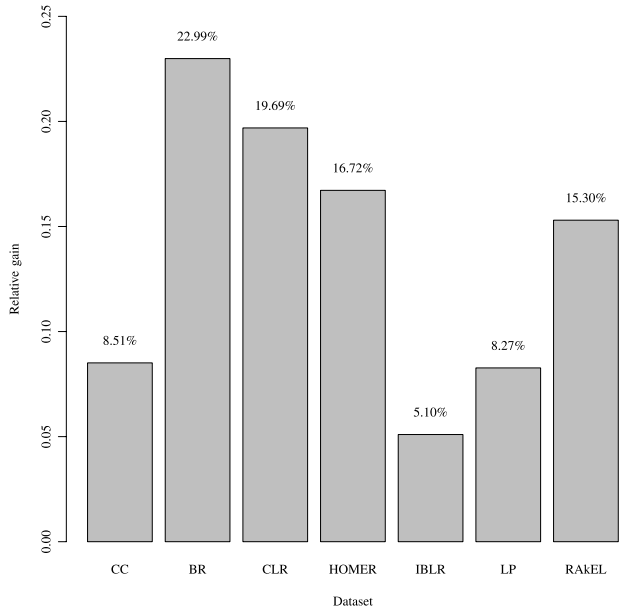

Fig. 4. Relative gain of LI-MLC in respect of the base execution time.

TABLE VI
EXACT P-VALUES FROM WILCOXON TEST

Algorithm	HL	F-Measure	Run time
BR	≥ 0.2	0.00714	7.25E-5
CC	≥ 0.2	0.09874	0.16880
CLR	≥ 0.2	0.06628	3.81E-6
HOMER	≥ 0.2	0.03850	7.25E-5
IBLR-ML	≥ 0.2	0.05594	0.00202
LP	≥ 0.2	0.01236	0.01597
RAkEL	≥ 0.2	0.00533	6.45E-4

evaluated with F-measure, these configurations was better in six and seven of 10 cases, respectively.

C. Statistical Study

Reading the previous tables, a general view of the improvements can be acquired. To know if this progression is significant from a statistical point of view, Table VI shows the p-values obtained by the Wilcoxon nonparametric statistical test for HL, F-score, and training time.

As can be observed, the significance of classification performance improvements will depend on the measure chosen.

1) With HL all p-values are above 0.2, which would mean that LI-MLC does not provide any benefit, which could be considered statistically significant.
2) Taking F-measure as reference, for CC, CLR, and IBLR-ML p-value < 0.1, and for BR, HOMER, LP, and RAkEL p-value < 0.05. Therefore, LI-MLC offers a statistically significant improvement with a 90% and 95% of confidence, respectively.

Regarding the execution time, all p-values are well below 0.05 except for CC. Thus, it can be said that LI-MLC significantly reduces the time necessary to train the classifiers, regardless of classification improvements.

Summarizing, LI-MLC offers a statistical significant reduction of execution time, as well as some significant improvements in classification results when they are evaluated with F-measure, a measure considered stronger than HL. These results can be justified for a number of reasons, discussed in the following.

D. On the Benefits of Label Reduction Performed by LI-MLC

All classifiers have a certain error ratio, as do ARs. Labels that are very common (have a large Supp in the data set) tend to produce a bias in classifiers, benefiting the majority classes (labels). The bias is reduced when LI-MLC hides these labels, obtaining models better able to classify samples in which less common labels appear. In some cases, the improvement in classification results led by LI-MLC is greater than the error introduced by the inference of the ARs.

For some methods like LP, reducing only one label halves the number of combinations generated by the data set transformation, resulting in a data set with far fewer classes, and therefore easier to process with multiclass classifiers. The ensemble methods based on LP transformation, such as RAkEL and HOMER, benefit from this enhancement. Given that for the bulk of the data sets, more than one rule is obtained, the reduction in the number of combinations is much more important, as will be the improvement in execution time.

Training a separate classifier for each label (BR method) in the data set seems a good idea, and it works well in many cases. However, this approach does not consider in its predictions the dependency among labels, valuable information as has been reported in [29] and [57]. ARs, on the other hand, collect these dependencies when there are strong implications as has been explained before, predicting the presence of a label when it rests more on label dependency than on the correlation between input attributes and their own label. Many MLC algorithms use BR as an underlying transformation, so an improvement in the base method also affects their results.

Regarding the time used by algorithms to build their models, the reduction in the number of labels performed by LI-MLC has a positive effect in almost all cases. The MLC algorithms benefit from a reduced set of labels, and the time spent applying LI-MLC is much lower than the time saved in building the simplified model. Table VI shows that all p-values are well under 0.05 (with the exception of CC), concluding that a significant statistical difference exists.

With MLC algorithms that build a binary classifier for each different label, such as BR and many ensemble proposals, LI-MLC produces at least a linear improvement in execution time with respect to the number of labels eliminated. When working with algorithms based on a combinatorial use of existent labels, the improvement could be much more important. In general, ensemble methods are benefited the most from the label space reduction, as it spans all the individual models, which compound the ensemble.

VII. CONCLUSION

In this paper, LI-MLC, a transformation method designed to reduce the number of labels in an MLD through the use of

ARs, has been presented. This approach can be used with any underlying MLC algorithm, allowing classifier training in less time, resulting in simpler models and improving classification results in many cases.

Furthermore, it has also been shown how it is possible to use certain statistical measures (CV, SK, and KR) that, unlike the Card and Dens measures by themselves, offer information useful for knowing when it is appropriate to use the proposed methodology. Without loss of generality, these measures can be used to characterize MLDs to obtain a general view of the distribution of labels in the data set.

The experimental results obtained over several MLDs with different classification algorithms, endorsed by the results from statistical tests, lead to the conclusion that it can be a useful approach for enhancing MLC.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers, whose comments contributed to improving this paper.

REFERENCES

[1] G. Tsoumakas, E. S. Xioufis, J. Vilcek, and I. Vlahavas. (2014, Jan. 1). *MULAN Multi-Label Dataset Repository* [Online]. Available: http://mulan.sourceforge.net/datasets.html

[2] Y. H. Li and A. K. Jain, "Classification of text documents," *Comput. J.*, vol. 41, no. 8, pp. 537–546, 1998.

[3] J. R. Bellegarda, "Large vocabulary speech recognition with multispan statistical language models," *IEEE Trans. Speech Audio Process.*, vol. 8, no. 1, pp. 76–84, Jan. 2000.

[4] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003.

[5] P. Mitra, C. A. Murthy, and S. K. Pal, "Unsupervised feature selection using feature similarity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 301–312, Mar. 2002.

[6] N. Spolaôr, E. A. Cherman, M. C. Monard, and H. D. Lee, "A comparison of multi-label feature selection methods using the problem transformation approach," *Electron. Notes Theoretical Comput. Sci.*, vol. 292, pp. 135–151, Mar. 2013.

[7] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. Data Eng.* [Online]. Available: http://dx.doi.org/10.1109/TKDE.2013.39

[8] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. Boston, MA, USA: Springer-Verlag, 2010, ch. 34, pp. 667–685.

[9] A. de Carvalho and A. Freitas, "A tutorial on multi-label classification techniques," in *Foundations of Computational Intelligence*, vol. 5, A. Abraham, A.-E. Hassanien, and V. Snášel, Eds. Berlin Heidelberg, Germany: Springer-Verlag, 2009, ch. 8, pp. 177–195.

[10] S. Godbole and S. Sarawagi, "Discriminative methods for multi-labeled classification," in *Proc. 8th Pacific-Asia Conf. Knowl. Discovery Data Mining*, LNCS 3056. 2004, pp. 22–30.

[11] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, and K. Brinker, "Multilabel classification via calibrated label ranking," *Mach. Learn.*, vol. 73, no. 2, pp. 133–153, 2008.

[12] M. Boutell, J. Luo, X. Shen, and C. Brown, "Learning multi-label scene classification," *Pattern Recognit.*, vol. 37, no. 9, pp. 1757–1771, 2004.

[13] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski, "An extensive experimental comparison of methods for multi-label learning," *Pattern Recognit.*, vol. 45, no. 9, pp. 3084–3104, 2012.

[14] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Proc. 5th Eur. Conf. Principles Data Mining Knowl. Discovery*, LNCS 2168. Freiburg, Germany, 2001, pp. 42–53.

[15] F. De Comité, R. Gilleron, and M. Tommasi, "Learning multi-label alternating decision trees from texts and data," in *Proc. 3rd Int. Conf. Mach. Learn. Data Mining Pattern Recognit.*, LNCS 2734. Leipzig, Germany, 2003, pp. 35–49.

[16] M. Zhang and Z. Zhou, "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognit.*, vol. 40, no. 7, pp. 2038–2048, 2007.

[17] W. Cheng and E. Hüllermeier, "Combining instance-based learning and logistic regression for multilabel classification," *Mach. Learn.*, vol. 76, nos. 2–3, pp. 211–225, 2009.

[18] M.-L. Zhang, "Multilabel neural networks with applications to functional genomics and text categorization," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 10, pp. 1338–1351, Oct. 2006.

[19] M.-L. Zhang, "ML-RBF : RBF neural networks for multi-label learning," *Neural Process. Lett.*, vol. 29, no. 2, pp. 61–74, 2009.

[20] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Advances in Neural Information Processing Systems*, vol. 14. Cambridge, MA, USA: MIT Press, 2001, pp. 681–687.

[21] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Mach. Learn.*, vol. 85, no. 3, pp. 333–359, 2011.

[22] G. Tsoumakas and I. Vlahavas, "Random k-labelsets: An ensemble method for multilabel classification," in *Proc. 18th ECML*, LNCS 4701. Warsaw, Poland, 2007, pp. 406–417.

[23] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Effective and efficient multilabel classification in domains with large number of labels," in *Proc. ECML/PKDD Workshop MMD*, Antwerp, Belgium, 2008, pp. 30–44.

[24] A. Chan and A. A. Freitas, "A new ant colony algorithm for multi-label classification with applications in bioinfomatics," in *Proc. 8th Annu. Conf. Genetic Evol. Comput.*, 2006, pp. 27–34.

[25] J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik, "Kernel dependency estimation," in *Advances in Neural Information Processing Systems*, vol. 15. New York, NY, USA: Springer-Verlag, 2002, pp. 873–880.

[26] D. Hsu, S. Kakade, J. Langford, and T. Zhang, "Multi-label prediction via compressed sensing," in *Advances in Neural Information Processing Systems*, vol. 22. New York, NY, USA: Springer-Verlag, 2009, pp. 772–780.

[27] T. Zhou, D. Tao, and X. Wu, "Compressed labeling on distilled labelsets for multi-label learning," *Mach. Learn.*, vol. 88, nos. 1–2, pp. 69–126, 2012.

[28] G. Tsoumakas, A. Dimou, E. Spyromitros, V. Mezaris, I. Kompatsiaris, and I. Vlahavas, "Correlation based pruning of stacked binary relevance models for multi-label learning," in *Proc. 1st Int. Workshop Learn. Multi-Label Data*, Bled, Slovenia, 2009, pp. 101–116.

[29] E. A. Cherman, J. Metz, and M. Monard, "A simple approach to incorporate label dependency in multi-label classification," in *Proc. 9th MICAI*, LNCS 6438. Pachuca, Mexico, 2010, pp. 33–43.

[30] M.-L. Zhang and K. Zhang, "Multi-label learning by exploiting label dependency," in *Proc. 16th Int. Conf. Knowl. Discovery Data Mining*, Washington, DC, USA, 2010, pp. 999–1008.

[31] Y. Guo and S. Gu, "Multi-label classification using conditional dependency networks," in *Proc. 22th Int. Joint Conf. Artif. Intell.*, vol. 2. 2011, pp. 1300–1305.

[32] L. Tenenboim-Chekina, L. Rokach, and B. Shapira, "Identification of label dependences for multi-label classification," in *Proc. 2nd Int. Workshop Learn. Multi-Label Data*, Haifa, Israel, 2010, pp. 53–60.

[33] F. Tai and H. Lin, "Multi-label classification with principle label space transformation," in *Proc. 2nd Int. Workshop Learn. Multi-Label Data*, Haifa, Israel, 2010, pp. 45–52.

[34] S.-H. Park and J. Fürnkranz, "Multi-label classification with label constraints," in *Proc. ECML PKDD Workshop Preference Learn.*, Antwerp, Belgium, 2008, pp. 157–171.

[35] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, "Algorithms for association rule mining—A general survey and comparison," *ACM SIGKDD Explorat. Newslett.*, vol. 2, no. 1, pp. 58–64, 2000.

[36] C. Silverstein, S. Brin, and R. Motwani, "Beyond market baskets: Generalizing association rules to dependence rules," *Data Mining Knowl. Discovery*, vol. 2, no. 1, pp. 39–68, 1998.

[37] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur, "Dynamic itemset counting and implication rules for market basket data," in *Proc. Int. Conf. Manag. Data*, Tucson, AZ, USA, 1997, pp. 255–264.

[38] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data Mining Knowl. Discovery*, vol. 8, no. 1, pp. 53–87, 2004.

[39] J. Read, B. Pfahringer, and G. Holmes, "Multi-label classification using ensembles of pruned sets," in *Proc. 8th IEEE ICDM*, Pisa, Italy, Dec. 2008, pp. 995–1000.

[40] D. A. Cieslak and N. V. Chawla, "Start globally, optimize locally, predict globally: Improving performance on imbalanced data," in *Proc. 8th IEEE ICDM*, Pisa, Italy, Dec. 2008, pp. 143–152.

[41] R. A. Groeneveld and G. Meeden, "Measuring skewness and kurtosis," *Statistician*, vol. 33, no. 4, pp. 391–399, 1984.
[42] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, *et al.*, "Finding interesting associations without support pruning," *IEEE Trans. Knowl. Data Eng.*, vol. 13, no. 1, pp. 64–78, Jan./Feb. 2001.
[43] I. Katakis, G. Tsoumakas, and I. Vlahavas, "Multilabel text classification for automated tag suggestion," in *Proc. ECML PKDD Discovery Challenge*, Antwerp, Belgium, 2008, pp. 75–83.
[44] B. Klimt and Y. Yang, "The enron corpus: A new dataset for email classification research," in *Proc. ECML*, Pisa, Italy, 2004, pp. 217–226.
[45] K. Crammer, M. Dredze, K. Ganchev, P. P. Talukdar, and S. Carroll, "Automatic code assignment to medical text," in *Proc. Workshop Biol., Translational, Clinical Lang. Process.*, Prague, Czech Republic, 2007, pp. 129–136.
[46] P. Duygulu, K. Barnard, J. de Freitas, and D. Forsyth, "Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary," in *Proc. 7th Eur. Conf. Comput. Vis.*, Copenhagen, Denmark, 2002, pp. 97–112.
[47] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei, and M. I. Jordan, "Matching words and pictures," *J. Mach. Learn. Res.*, vol. 3, pp. 1107–1135, Feb. 2003.
[48] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 2, pp. 467–476, Feb. 2008.
[49] A. Wieczorkowska, P. Synak, and Z. Raś, "Multi-label classification of emotions in music," in *Intelligent Information Processing and Web Mining*, vol. 35. New York, NY, USA: Springer-Verlag, 2006, ch. 30, pp. 307–315.
[50] S. Diplaris, G. Tsoumakas, P. Mitkas, and I. Vlahavas, "Protein classification with multiple algorithms," in *Proc. 10th PCI*, Volos, Greece, 2005, pp. 448–456.
[51] C. G. M. Snoek, M. Worring, J. C. van Gemert, J. M. Geusebroek, and A. W. M. Smeulders, "The challenge problem for automated detection of 101 semantic concepts in multimedia," in *Proc. 14th Annu. ACM Int. Conf. Multimedia*, Santa Barbara, CA, USA, 2006, pp. 421–430.
[52] J. Read and P. Reutemann. (2014, Jan. 1). *MEKA Multi-Label Dataset Repository* [Online]. Available: http://meka.sourceforge.net/#datasets
[53] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.
[54] J. Luengo, S. García, and F. Herrera, "A study on the use of statistical tests for experimentation with neural networks: Analysis of parametric test conditions and non-parametric tests," *Expert Syst. Appl.*, vol. 36, no. 4, pp. 7798–7808, 2009.
[55] D. J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*. London, U.K.: Chapman & Hall, 2003.
[56] J. Alcala-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, *et al.*, "Keel data-mining software tool: Data set repository and integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, nos. 2–3, pp. 255–287, 2011.
[57] E. Alvares-Cherman, J. Metz, and M. C. Monard, "Incorporating label dependency into the binary relevance framework for multi-label classification," *Expert Syst. Appl.*, vol. 39, no. 2, pp. 1647–1655, 2012.

Francisco Charte received the B.Eng. degree in computer science from the University of Jaén, Jaén, Spain, in 2010, and the M.Sc. degree in soft computing and intelligent systems from the University of Granada, Granada, Spain, in 2011.

He is currently a PreDoctoral Researcher with the University of Granada. His current research interests include machine learning with applications to multilabel classification, high dimensionality and imbalance problems, and association rule mining, as well as CPU/GPU algorithm parallelization techniques.

Antonio J. Rivera received the B.Sc. and Ph.D. degrees in computer science from the University of Granada, Granada, Spain, in 1995 and 2003, respectively.

He is a Lecturer of computer architecture and computer technology with the Computer Science Department, University of Jaén, Jaén, Spain. His current research interests include multilabel classification, imbalance problems, evolutionary computation, neural network design, time series prediction, and regression tasks.

María J. del Jesus received the M.Sc. and Ph.D. degrees in computer science from the University of Granada, Granada, Spain, in 1994 and 1999, respectively.

She is an Associate Professor with the Department of Computer Science, University of Jaén, Jaén, Spain. Her current research interests include fuzzy rule-based systems, genetic fuzzy systems, subgroup discovery, data preparation, feature selection, evolutionary radial basis neural networks, knowledge extraction based on evolutionary algorithms, and data mining.

Francisco Herrera (M'10) received the M.Sc. and Ph.D. degrees in mathematics from the University of Granada, Granada, Spain, in 1988 and 1991, respectively.

He is currently a Professor with the Department of Computer Science and Artificial Intelligence, University of Granada. He has been the supervisor of 28 Ph.D. students. He has published more than 240 papers in international journals. He is the co-author of the book *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases* (World Scientific, 2001). His current research interests include computing with words and decision making, bibliometrics, data mining, big data, cloud computing, data preparation, instance selection and generation, imperfect data, fuzzy rule based systems, genetic fuzzy systems, imbalanced classification, knowledge extraction based on evolutionary algorithms, memetic algorithms and genetic algorithms, and biometrics.

Dr. Herrera is currently an Editor-in-Chief of the *International Journal Progress in Artificial Intelligence* (Springer). He acts as an Area Editor of the *International Journal of Computational Intelligence Systems* and Associate Editor of the journals: the IEEE TRANSACTIONS ON FUZZY SYSTEMS, *Information Sciences, Knowledge and Information Systems, Advances in Fuzzy Systems*, and the *International Journal of Applied Metaheuristics Computing*, and he serves as a member of several journal editorial boards: *Fuzzy Sets and Systems, Applied Intelligence, Information Fusion, Evolutionary Intelligence*, the *International Journal of Hybrid Intelligent Systems, Memetic Computation*, and *Swarm and Evolutionary Computation*. He received the following honors and awards: the ECCAI Fellow in 2009, the IFSA Fellow in 2013, the Spanish National Award on Computer Science ARITMEL to the "Spanish Engineer on Computer Science" in 2010, the International Cajastur "Mamdani" Prize for Soft Computing in 2010, the IEEE TRANSACTIONS ON FUZZY SYSTEMS Outstanding 2008 Paper Award (bestowed in 2011), and the 2011 Lotfi A. Zadeh Prize Best Paper Award of the International Fuzzy Systems Association.