Knowledge-Based Systems 89 (2015) 385–397

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation

Francisco Charte [a,*], Antonio J. Rivera [b], María J. del Jesus [b], Francisco Herrera [a,c]

[a] Department of Computer Science and A.I., University of Granada, 18071 Granada, Spain
[b] Department of Computer Science, University of Jaén, 23071 Jaén, Spain
[c] Faculty of Computing and Information Technology – North Jeddah, King Abdulaziz University, 21589 Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 14 April 2015
Received in revised form 16 June 2015
Accepted 19 July 2015
Available online 23 July 2015

Keywords:
Multilabel classification
Imbalanced learning
Oversampling
Synthetic instance generation

ABSTRACT

Learning from imbalanced data is a problem which arises in many real-world scenarios, so does the need to build classifiers able to predict more than one class label simultaneously (multilabel classification). Dealing with imbalance by means of resampling methods is an approach that has been deeply studied lately, primarily in the context of traditional (non-multilabel) classification.

In this paper the process of synthetic instance generation for multilabel datasets (MLDs) is studied and MLSMOTE (Multilabel Synthetic Minority Over-sampling Technique), a new algorithm aimed to produce synthetic instances for imbalanced MLDs, is proposed. An extensive review on how imbalance in the multilabel context has been tackled in the past is provided, along with a thorough experimental study aimed to verify the benefits of the proposed algorithm. Several multilabel classification algorithms and other multilabel oversampling methods are considered, as well as ensemble-based algorithms for imbalanced multilabel classification. The empirical analysis shows that MLSMOTE is able to improve the classification results produced by existent proposals.

1. Introduction

Classification is one of the main supervised learning applications, an important field in Machine Learning [1]. The goal is to train a model using a set of labeled data samples, obtaining a classifier able to label new, never seen before, unlabeled samples. The datasets used in traditional classification have only one class per instance. By contrast, in multilabel datasets (MLDs) [2] each instance has more than one class assigned, and the total number of different classes (labels) can be huge.

In many real world scenarios, such as text classification [3] and fraud detection [4], the number of instances associated to some classes is much smaller (greater) than the amount of instances assigned to others. This problem, known as imbalanced learning, has been widely studied over the last decade [5] in the context of classic classification. It is also present in multilabel classification (MLC), since labels are unevenly distributed in most MLDs. To deal with imbalance in MLC, methods based on algorithmic adaptations [6–8], the use of ensembles [9,10], and resampling techniques [11–13] have been proposed.

Among the existent resampling techniques, those based on the generation of new samples (oversampling) have shown [14] to work better than others. The new samples can be clones of existent ones, or be synthetically produced as in SMOTE (Synthetic Minority Over-sampling Technique) [15]. Multilabel oversampling algorithms based on the cloning approach have been proposed in [12,13], being demonstrated its capability to deliver an improvement in classification results. A synthetic approach to produce new samples in MLDs is still to be faced. SMOTE is the most popular algorithm for this task in non-multilabel datasets, so it would be a good starting point.

Imbalance in MLC has been faced mainly through algorithmic adaptations and the use of ensembles, while the resampling approach is the least examined path until now. Nevertheless it is an interesting way and deserves to be taken into account, as the results in [12] have shown. Since oversampling algorithms seem to produce better results, designing a more advanced method to produce new data samples could be worth the effort. This is the motivation behind MLSMOTE (Multilabel Synthetic Minority Over-sampling Technique), a novel multilabel oversampling algorithm designed to create synthetic instances associated to minority labels.

The popular SMOTE algorithm takes all the samples belonging to the minority class, picks a random instance among the nearest

* Corresponding author. Tel.: +34 953 212 892; fax: +34 953 212 472.
E-mail addresses: francisco@fcharte.com (F. Charte), arivera@ujaen.es (A.J. Rivera), mjjesus@ujaen.es (M.J. del Jesus), herrera@ugr.es (F. Herrera).

http://dx.doi.org/10.1016/j.knosys.2015.07.019
0950-7051/© 2015 Elsevier B.V. All rights reserved.

neighbors of each one, and produces a new sample with the same minority class. Both the number of nearest neighbors and the amount of synthetic instances used for each minority sample can be adjusted. In a multilabel context there will always be more than one minority label, thus a strategy for choosing the appropriate instances has to be established. Moreover, the synthetic instances need a set of labels (labelsets) instead of being associated to an individual class. Therefore, a method to generate these synthetic labelsets has also to be settled.

The aim of this paper is to present MLSMOTE. As mentioned above, its goal is to produce synthetic instances associated to minority labels. In order to know which labels are minority, MLSMOTE leans on the multilabel imbalance measures proposed in [16]. The features of the synthetic instances are obtained by interpolation of values belonging to nearest neighbors, as in SMOTE. The labelsets of these new instances are also gathered from nearest neighbors, taking advantage of label correlation information in the neighborhood. For this task three different methods were studied, the intersection of the labels which appear in the neighbors, the union of those, and a third method based on a ranking of label occurrences. An extensive experimentation, structured in two different phases that will be detailed later, has been conducted. From the analysis of this experimentation it can be concluded that MLSMOTE, our multilabel synthetic minority oversampling technique, accomplishes a general improvement in classification results when compared with previous proposals with the same purpose.

The rest of this paper is organized as follows. In Section 2 the MLC and imbalanced learning problems are introduced. Section 3 provides a comprehensive review on the published approaches to work with multilabel imbalanced datasets. Section 4 provides all the details about the MLSMOTE algorithm, its parameters and implementation. In Section 5 the experimental framework used is defined, and the results obtained from experimentation are analyzed. Section 6 provides a final discussion and conclusions.

2. Preliminaries

The algorithm proposed in this paper has ties with two different topics, multilabel classification and imbalanced learning. In this section a brief introduction to both is provided, along with some specific details regarding imbalanced learning in the multilabel context.

2.1. Multilabel classification

In traditional classification the datasets are composed of a set of input features and a unique value in the output attribute, the class or label. In MLC [2] each sample may contain more than one value (class) in the output feature. Thus, the output of the classifier is not an individual label but a set of them. As stated in [2], a multilabel classifier will usually generate its prediction using two different methods. One is giving as output a bipartition, composed as a set of *true/false* values for each label. Another is returning a label ranking. In any case, most MLC solutions are built around one of two different approaches:

- Data transformation methods aim to convert the original dataset in order to use traditional classification algorithms to process it. A complete taxonomy of transformation methods for MLDs can be found in [17]. The two most popular ones are called Binary Relevance (BR) [18] and Label Powerset (LP) [19]. The former generates multiple binary datasets, one for each label, while the latter produces only one multiclass dataset, using as class the set of active labels in each sample.
- The goal of the method adaptation approach is to modify known classification algorithms to make them able to work with MLDs. There are many proposals in this field, from MLC trees like ML-TREE [20] or a multilabel kNN called ML-kNN [21] to multilabel neural networks [22,23] and SVMs [24]. There are also several methods based on ensembles of classifiers, such as RAkEL [25], CLR [26], HOMER [27], CC [28], and ECC [29], as well as other approaches to the problem, such as the use of Error-Correcting Codes [30].

In addition to new algorithms, MLC also demanded specific measures to evaluate classification results, as well as measures aimed to assess some MLDs peculiarities. In [2] the definition of most of them can be found. A recent review on the state-of-the-art multilabel learning algorithms, as well as evaluation measures, can be found in [31]. The measures used in this study will be defined later, in SubSection 2.3 (characterization measures) and Section 5 (evaluation measures).

2.2. Imbalance in traditional classification

In general, most classifiers underperform when used with imbalanced datasets. As stated in [32] the reason lies in their design, aimed to reduce the global error rate. This is a design which tends to benefit the most represented class in the dataset (majority class), labeling new instances with this class at the expense of the minority class. Moreover, imbalanced distribution of classes can complicate other common problems, such as noisy labels [33].

Three main approaches [34] have been proposed to face the imbalance problem. Data resampling follows the preprocessing approach, rebalancing the class distribution by deletion [35] or creation [15] of instances. Resampling techniques are classifier-independent solutions to the imbalanced learning problem, albeit some proposals for specific classifiers exist [36], and have shown their general effectiveness [14]. The other two approaches, algorithmic adaptations [37] and cost sensitive classification [38], are classifier dependent. The goal of the former is to modify existent classifiers taking into account the imbalanced nature of the datasets. The latter combines the design of adapted classifiers with some data preprocessing techniques. The present study is focused in the first approach. A general introduction and additional details about these approaches can be found in [39]. In some cases, resampling techniques are used along with ensembles of classifiers to tackle the imbalance problem. A general overview on ensemble methods is provided in [40]. The use of ensembles in imbalanced classification was recently reviewed in [41], and some specific algorithms are proposed in [42].

Most resampling algorithms consider one majority (minority) class only. Thus, undersampling techniques remove instances from the most frequent class only, whereas oversampling methods create instances from the least frequent one only. SMOTE works this way, generating new samples associated to the least frequent class. Firstly, the set of instances belonging to the minority class is obtained. For each instance in this set, SMOTE gathers a small batch of nearest neighbors. Typically the size of this group is 5. For each synthetic instance to produce, one of these neighbors is randomly picked. The features of the new sample are interpolated along the line which connects the reference and the neighbor instances. The class of the synthetic sample is always the minority class.

2.3. Imbalance in multilabel classification

The total number of distinct labels tends to be quite large in nearly all MLDs. The most usual cases are in the range from several dozens to a few hundreds of labels. There also are some extreme cases, with some MLDs having less than ten labels and others which use more than a thousand of them.

Despite the large set of labels appearing in many MLDs, each one of their samples is usually associated to a very small subset of them. As can be seen in Table 3 (see Section 5), for the MLDs used in our experimentation the average number of labels per sample, measure known as Cardinality (*Card*), is always below 5 with the exception of cal500. Intuitively, it is easy to deduct that some of the labels appear in many samples while others are scarcely represented. In general, the larger is the number of different labels, the higher will be the likelihood that some of them are underrepresented (overrepresented). Fig. 1 is a representation of the percentage of samples in which the 40 most common labels in each MLD appear. That most MLDs have two or three very frequent labels, whereas the rest of them are barely represented, can be observed. Overall, it can be seen that most labels are present in less than 5% of the instances. It must be taken into account that many of these MLDs have more than 40 labels. Those not appearing in Fig. 1 are much less frequent. Therefore, that a high imbalance among the labels exists can be visually inferred.

This imbalance ratio can be assessed with two of the measures proposed in [16] for this task. *IRLbl* is a measure obtained for each label in the dataset. It is defined as shown in Eq. (1) and assesses the individual label imbalance ratio. As can be seen in Eq. (2), *MeanIR* is obtained by averaging the *IRLbl* for all labels. In these equations D stands for the MLD, L for the full set of labels, L_l for the lth label in this set, and Y_i for the labelset associated to the ith instance in D. Both measures can be easily obtained with the mldr R package [43].

$$IRLbl(l) = \frac{\operatorname{argmax}_{l'=L_1}^{L_{|L|}} \left(\sum_{i=1}^{|D|} h(l', Y_i) \right)}{\sum_{i=1}^{|D|} h(l, Y_i)}, h(l, Y_i) = \begin{cases} 1 & l \in Y_i \\ 0 & l \notin Y_i \end{cases} \tag{1}$$

$$MeanIR = \frac{1}{|L|} \sum_{l=L_1}^{L_{|L|}} (IRLbl(l)). \tag{2}$$

The previous fact leads to a fundamental difference between imbalanced MLDs with respect to imbalance in traditional datasets, there is not only one minority (majority) class but a group of minority (majority) labels. Therefore, any algorithm designed to tackle imbalance in MLDs should take into account that it must consider several labels as targets, rather than only one.

Another intrinsic characteristic of imbalanced MLDs, which does not exist in traditional datasets, is the joint appearance of minority and majority labels in the same samples [44]. This casuistic adds another obstacle to the learning from imbalanced MLDs process. Furthermore, it makes it harder to find solutions based on resampling techniques. Fig. 2 shows how the labels in the tmc2007 dataset interact. The size of each arc is proportional to the frequency of the represented label. The innermost ring shows the color code associated to each label, and also indicates the number of instances in which it appears. The outermost ring provides a relative scale, along with the color codes corresponding to other labels present in the same set of instances. Color coded links depict the interactions among labels. The thickness of each link is proportional to the number of instances involved in each interaction. That more than a half of the instances in which the C22 label, one of the minority labels, appears are also associated to the C02 label, which is the most frequent label, can be noticed. The same circumstance affects to other minority labels in this MLD, as well as most of the remainder MLDs used frequently in the literature.

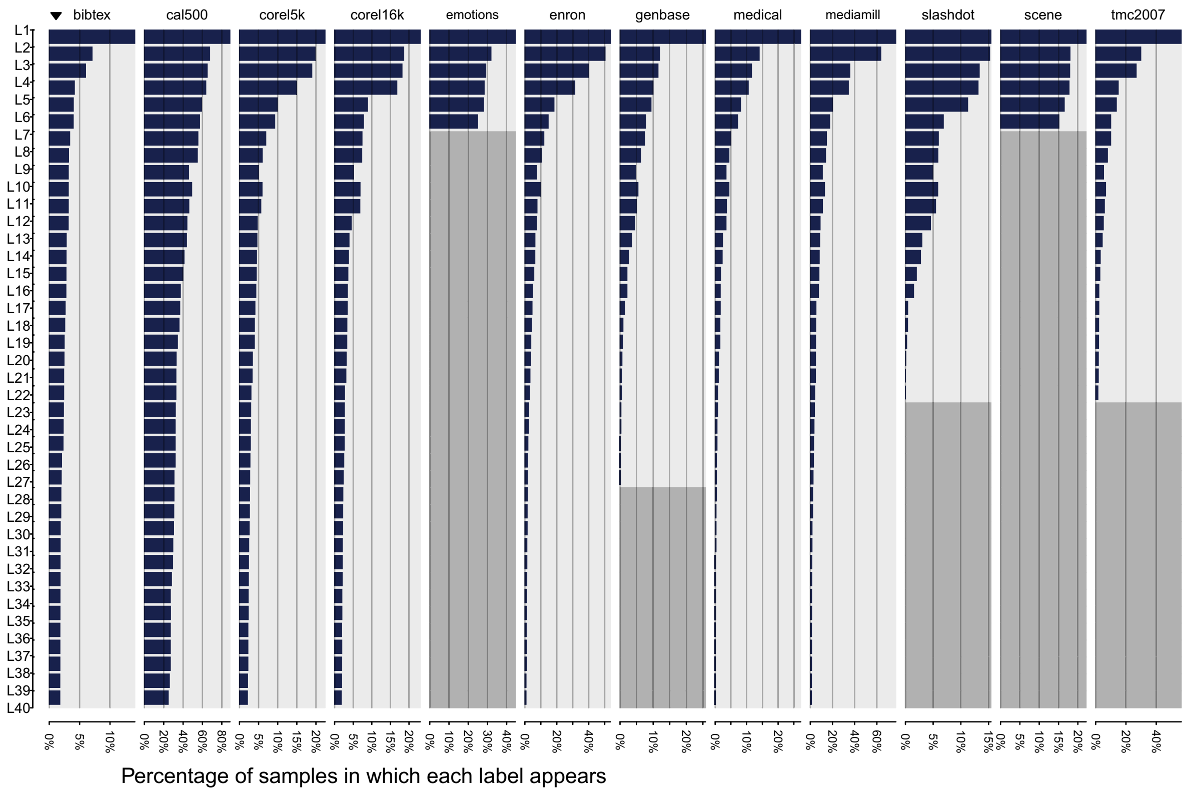

Fig. 1. Frequency of the 40 most common labels in each dataset.

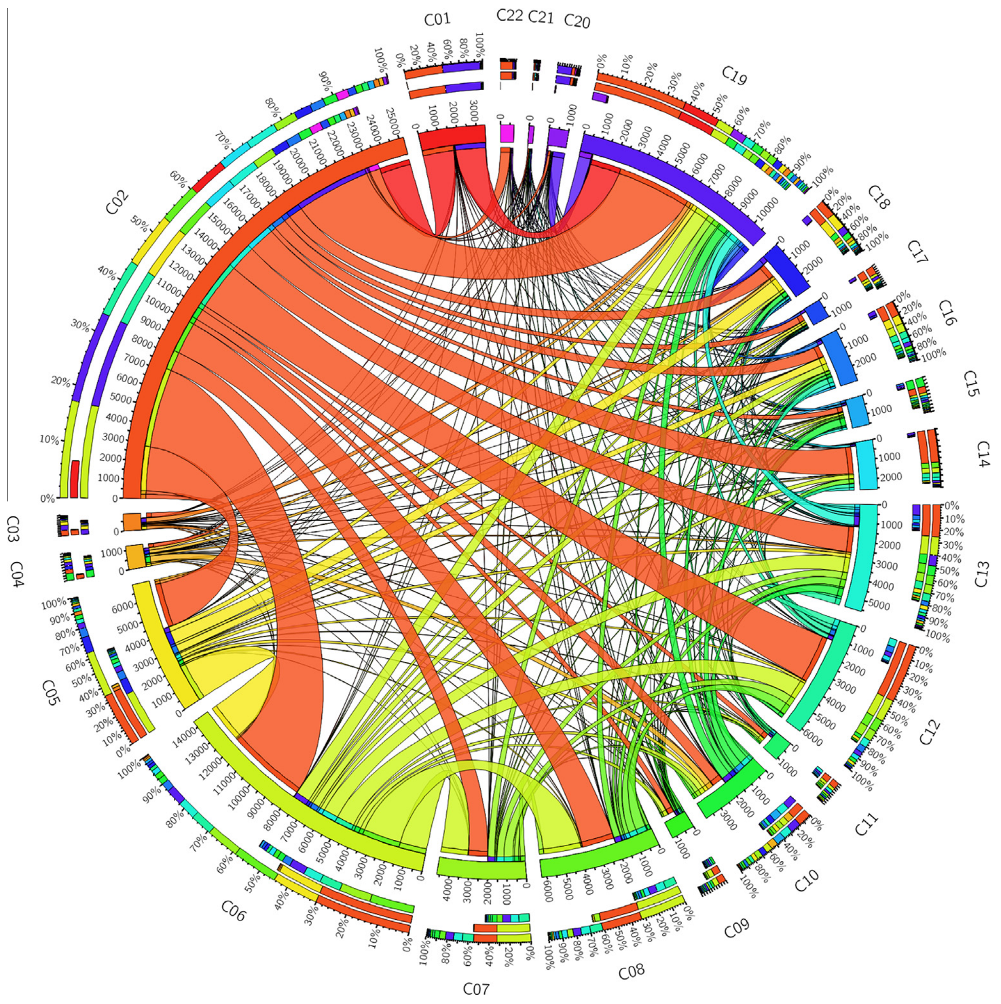

Fig. 2. Concurrence among labels in the tmc2007 MLD.

3. Related work: learning from imbalanced MLDs

Many published works claim the intrinsically imbalanced nature of MLDs, a fact experimentally stated in [12] by means of specific measures. In this section, the several published ways to accomplish classification with imbalanced MLDs are depicted, organized according to three well-known approaches: algorithmic adaptations, ensemble-based methods and resampling techniques.

3.1. MLC algorithmic adaptation proposals

Several solutions based on algorithmic adaptations have been proposed in late years to deal with imbalanced MLDs, among them:

- In [6] the authors face a highly imbalanced multilabel problem, as is the prediction of human proteins localizations. Their method, based on non-parametric probabilistic models, combines the use of covariance matrices to obtain label correlations and weighted coefficients associated to each label to fix the imbalance problem.
- The proposal in [7] is an adaptation of MIMLRBF [45], a multi-instance-multilabel classification algorithm based on radial basis artificial neural networks (RBFN). The proposed algorithm optimizes the original one to work with imbalanced MLDs in two ways. The number of units in the hidden layer is not constant, as in MIMLRBF, but calculated taking into account the number of samples per label. Additionally, the weights connecting this hidden layer with the output layer are adjusted applying an individual bias for each label.
- Also based on ANN, in [8] an iterative enrichment process is proposed. The authors initialize the ANN by clustering part of the data, after which a resampling over each cluster is performed in order to balance them in the euclidean space. Once initialized, the training consists in an enrichment method which removes samples and aggregates new data samples to improve the equilibrium of the network.
- In [46] a Min–Max-Modular network [47] is used in order to divide the classification task in several smaller tasks. Different strategies are tested to ensure that the imbalance in these smaller problems is lower than in the original one. The decomposition of the task in several smaller ones is faced randomly, through clustering, or using Principal Component Analysis [48]. The classification subproblems are always binary, and they are processed with SVM algorithms.

All these methods have been designed to be algorithm dependent, and their applications are mostly very specific to the domains described in the referenced papers.

3.2. Ensemble based proposals

Classification techniques relying on ensembles of classifiers have shown their strength in the MLC field, with algorithms such as RAkEL [25], ECC [29], and HOMER [27] among the best performers. This same approach has been also applied to solve the imbalance problem:

- The proposal made in [9] builds an heterogeneous ensemble (EML from now on) using RAkEL [25], ECC [29], CLR [26], MLkNN [21], and IBLR [49] as underlying MLC algorithms. The authors state that the use of ensembles will be positive to work with imbalanced MLDs. Some of the algorithms used in the proposed ensemble are ensembles of classifiers by themselves. The authors test several procedures to obtain a final prediction combining the individual ones, applying different thresholds and weights to each classifier by means of cross validation.
- Originally designed to face imbalance in traditional classification, the algorithm proposed in [10] is also useful for imbalanced MLC through the creation of ensembles with BR classifiers. The proposal, called BR-IRUS, grounds its strategy on training several classifiers for each label, using in each round all the minority samples but only a small random subset of the majority ones. This way several boundaries surrounding the minority space are obtained.

A major weakness of these proposals is usually their efficiency, as they demand the training of a large number of classifiers. This number would depend on the MLD's total number of labels, and some MLDs have several hundreds of them. Additionally they are algorithm-dependent solutions at some extent, since there is no freedom to chose among the existent MLC algorithms.

3.3. Resampling techniques proposals

Contrasting with the methods covered previously, the ones mentioned here belong to the classifier-independent algorithms group. This means they are not tied to any MLC algorithm, but are based on the preprocessing of the MLDs aiming to produce new, more balanced versions of them. Therefore, the application range of these methods is broader than the classifier-dependent ones. The following are some published proposals in this field:

- The proposal in [11] is an undersampling algorithm for multilabel text classification. The authors follow a one-vs-all way, generating an individual binary dataset for each label. All the samples which belong to a certain label are marked as positive, whereas the remainder ones will be marked as negative no matter what labels they contain. The number of the majority class instances in each binary dataset, which is usually the negative class, is reduced aiming to lower the bias of the classifiers. The proposed solution could be also considered a transformation method, since its output is intended to be processed with binary classification algorithms instead of MLC algorithms.
- Two undersampling and two oversampling algorithms are presented in [12], one of them based on some specific measures introduced in [16] and directed to assess the imbalance level in MLDs. As in other studies [14], the conducted experimentation discovers that the oversampling methods perform usually better than the undersampling ones. The two proposed oversampling methods, called LP-ROS and ML-ROS, are random oversampling algorithms, and both work by cloning instances associated to minority cases. What makes them different are the procedures followed to decide which cases are considered as minority. LP-ROS considers each label combination as class identifier, while ML-ROS individually evaluates the imbalance level of each label. Eventually, the latter approach proved to be the most successful. Despite the apparent simplicity of the random oversampling techniques, ML-ROS achieved a significant overall improvement of classification results with many MLDs.
- In [13], the authors analyze different strategies aimed to apply the original SMOTE algorithm to MLDs. They do it by means of three ways for selecting the seed instances, i.e. the instances which will be taken as reference to locate their nearest neighbors. Only one minority label is considered, what changes is the method used to choose the instances which belong to that label. The first method gives to SMOTE all the instances in which the minority label appears, following a one-vs-all approach as in the BR transformation method. The second way selects instances in which only the minority label is present, restricting the set of seed samples to those which are single-labeled. In the third strategy the seed samples are processed in batches, as many as different labelsets there are with the minority label in them. Eventually the latter method, which uses the UG strategy (SmoteUG from now on), proved to generate better results, whereas the other two produced a general degradation in classification performance.

Although ML-ROS [12] is a basic random oversampling algorithm, it is able to generate new instances for several minority labels. By contrast the analysis conducted in [13] relies in a more sophisticated oversampling algorithm such as SMOTE, but taking into account only one minority label. Thus, it ignores the intrinsic nature of MLDs. The following section shows how MLSMOTE obtains the best of these two approaches.

4. MLSMOTE multilabel synthetic instance generation

Instead of producing synthetic samples only from one class, MLSMOTE individually processes the set of instances in which each minority label appear. Each minority sample will be the seed for a new synthetic sample. These new instances need a set of features, as well as a synthetic labelset indicating which of the labels appearing in the reference sample and its neighbors will also be present in the synthetic instance. Thus, there are four main aspects to solve:

1. Minority instances selection: The algorithm assumes that most MLDs will have more than one minority label. Therefore, a criterion to know which labels are minority ones has to be established.
2. Nearest neighbor search: Once an instance belonging to a minority label has been selected, the algorithm has to search its nearest neighbors.
3. Feature set generation: Having selected one of the neighbors, the set of features for the synthetic instance is obtained through interpolation techniques.
4. Synthetic labelset production: Finally, given the multilabel nature of the problem at glance, a synthetic labelset has to be generated for the new instance.

The following subsections detail each one of the previously mentioned main aspects.

4.1. Minority instances selection

The generation of synthetic samples starts by selecting a minority instance as reference point. Hence, determining which instances will be considered as minority ones is needed. MLSMOTE relies in two aforementioned measures, *IRLbl*, defined in Eq. (1), and *MeanIR*, defined in Eq. (2).

The criterion followed to consider a label l as minority is that $IRLbl(l) > MeanIR$. In other words, that the frequency of the label l is below the average frequency of all labels in the MLD. This way the threshold used to cut the set of labels into two subsets, minority and majority labels, directly comes out of the MLD, instead of setting an ad hoc level or number of labels to be considered as minority.

Using these two measures, lines 3–6 in Algorithm 1 obtain for each minority label, those whose *IRLbl* is above *MeanIR*, a bag with all the instances in which it appears. Each sample in the bag will be taken as origin to search for nearest neighbors. It should be noted that the same instance could be selected several times, as it could be associated to more than one minority label.

4.2. Neighbor set selection

Once a minority instance has been chosen (*sample* in line 8 in Algorithm 1), the next step will be to select a set with its nearest neighbors. The size of this set is established by the *k* parameter. The process begins by obtaining the distances between *sample* and the other instances in *minBag*. The distance between two samples is measured aggregating the differences amid their corresponding features. For numeric attributes the euclidean distance is used, while for the nominal ones we rely on the Value Difference Metric (VDM) [50].

The number of nearest neighbors considered, assigned to *k*, is 5 by default, as recommended in the original implementation of SMOTE [15]. One of them is randomly picked (*refNeigh* in line 13 of Algorithm 1), and will be acting as the reference neighbor.

4.3. Feature set and label set generation

Having reached line 15 there are two samples selected, *sample* and *refNeigh*. The feature values of the synthetic instance will be interpolated (lines 24–31 in Algorithm 2) along the line which connects these two samples, for numeric attributes, or taking into account all the nearest neighbors, for nominal attributes.

Regarding the label set of the new instance, the method used until now in other oversampling proposals has been the cloning of the seed instance labelset. This technique completely disregards the information related to label correlations. Rather than build a global correlation model, as we did in [51], MLSMOTE tries to take advantage of the data about correlation that can be obtained from the neighborhood. Three possibilities were considered and empirically assessed:

- Intersection: Only the labels which appear in the reference sample and all its neighbors will be in the synthetic labelset.
- Union: All the labels which appear in the reference sample or any of its neighbors will be in the synthetic labelset.
- Ranking: A counting of the number of occurrences of each label in the reference sample and its neighbors is made, including in the synthetic labelset those present in half or more of the instances considered, as usual in most voting schemes.

At first, a preference for one of them did not exist. Therefore, the interest is in performing an all-against-all comparison to decide which is the best one. To do this, the MLDs mentioned in Section 5 were preprocessed with MLSMOTE using the three possible configurations. Then, the obtained datasets were used with the MLC algorithms indicated in the same Section 5 and average test results were retrieved. Two evidences can be drawn from those results. The first is that, with few exceptions, the intersection method for labelset generation is the worst performer. The second is the difficulty to know which of the other two methods is performing best, since both achieve many best cases and there are also many ties.

To select the most appropriate labelset generation method the Friedman test was applied. This test rejected the null-hypothesis, indicating that some statistical differences exist. Table 1 shows the average rankings obtained by each method for the three considered measures. As can be seen, the ranking method always obtains the lowest values, which means that it is positioned ahead of the others. To know the statistical significance of the differences in ranking the Bergmann's procedure was followed, obtaining the *p-values* shown in Table 2. The ranking-based method achieves significant differences against the intersection in both measures. Although the difference between Ranking and Union methods could not be considered statistically significant, the average ranking of the former is always the best. This leads to conclude that MLSMOTE performs better when the ranking method is used to generate the labelsets. Therefore, this will be the labelset generation method used from here on.

Therefore, the ranking approach proved to be the most effective, and it was implemented as shown in lines 35–38 in Algorithm 2.

The synthetic sample is eventually added to the MLD (line 17 in Algorithm 1), and the algorithm will then process the remainder instances of the same minority label first, and after that the rest of minority labels. Aiming to achieve the best possible balance, the *IRLbl* for each label is reassessed at the beginning of each cycle (line 4 in Algorithm 1). This way, if a minority label reaches the *MeanIR* value while processing, it will be excluded from the synthetic sample generation procedure.

Table 1
Friedman rankings for the labelset generation methods.

Algorithm	MacroFM	MicroFM
Ranking	1.8077	1.7385
Union	1.9231	1.9231
Intersection	2.2692	2.3385

Table 2
Adjusted *p*-values produced by Bergmann's procedure.

Hypothesis	MacroFM	MicroFM
Ranking vs Intersection	0.0255	0.0019
Union vs Intersection	0.0485	0.0179
Ranking vs Union	0.5107	0.2926

Algorithm 1. MLSMOTE algorithm's pseudo-code.

```
Inputs:
    D    ▷ Dataset to oversample
    k    ▷ Number of nearest neighbors
 1: $L \leftarrow$ labelsInDataset($D$)    ▷ Full set of labels
 2: $MeanIR \leftarrow$ calculateMeanIR($D, L$)
 3: for each $label$ in $L$ do
 4:     $IRLbl_{label} \leftarrow$ calculateIRperLabel($D, label$)
 5:     if $IRLbl_{label} > MeanIR$ then
 6:         ▷ Bags of minority labels samples
 7:         $minBag \leftarrow$ getAllInstancesOfLabel($label$)
 8:         for each $sample$ in $minBag$ do
 9:             $distances \leftarrow$ calcDistance($sample, minBag$)
10:             sortSmallerToLargest($distances$)
11:             ▷ Neighbor set selection
12:             $neighbors \leftarrow$ getHeadItems($distances, k$)
13:             $refNeigh \leftarrow$ getRandNeighbor($neighbors$)
14:             ▷ Feature set and labelset generation
15:             $synthSmpl \leftarrow$ newSample($sample$,
16:                         $refNeigh, neighbors$)
17:             $D = D + synthSmpl$
18:         end for
19:     end if
20: end for
```

Algorithm 2. Function: Generation of new synthetic instances.

```
21: **function** NEWSAMPLE(sample, refNeigh, neighbors)
22:     synthSmpl ← **new** Sample    ▷ New empty instance
23:     ▷ Feature set assignment
24:     **for each** feat **in** synthSmpl **do**
25:         **if** typeOf(feat) is numeric **then**
26:             diff ← refNeigh.feat − sample.feat
27:             offset ← diff ∗ randInInterval(0,1)
28:             value ← sample.feat + offset
29:         **else**
30:             value ← mostFreqVal(neighbors,feat)
31:         **end if**
32:         syntSmpl.feat ← value
33:     **end for**
34:     ▷ Label set assignment
35:     lblCounts ← counts(sample.labels)
36:     lblCounts + ← counts(neighbors.labels)
37:     labels ← lblCounts > $(k + 1)/2$
38:     synthSmpl.labels ← labels
39:     **return** synthSmpl
40: **end function**
```

4.4. MLSMOTE pseudo-code

Algorithm 1 shows the pseudo-code of the proposed MLSMOTE algorithm. The inputs to the algorithm are D, the MLD to be processed, and k, the number of neighbors to use. The output will be D including the synthetic samples generated by MLSMOTE.

The main body of the algorithm spans from line 3 to line 20, a loop that goes across all the labels in the dataset. For each label its *IRLbl* is obtained, and if it is above the MLD *MeanIR* the label is considered as minority. In this case all the samples in which the label appears are taken as seed instances, looking for their nearest neighbors and generating a synthetic sample.

Algorithm 2 shows the function which is in charge of generating new synthetic instances. This function needs as inputs the seed sample, its k nearest neighbors, and a random neighbor (one of the previous k neighbors) that will be taken as reference to interpolate the features values. As can be seen, the function can be divided in two parts. The first one (lines 24–33) produces the feature values of the synthetic sample, while the second one (lines 35–38) generates the synthetic labelset.

4.5. MLSMOTE formal definition

Let L be the set of labels in D. $\forall\lambda \in L,\ IRLbl_\lambda$ is defined. $\mathcal{F} = \prod_{i=1}^{n} F_i$ being the input feature space. Each instance in D can be expressed as $(f, O) \in \mathcal{F} \times \mathcal{P}(L)$. Taking $M = \{\lambda \in L : IRLbl_\lambda > MeanIR\}$, and $\forall\lambda \in M,\ S_\lambda = \{(f, O) \in D : \lambda \in O\}$. Assuming that, for $\lambda \in L$, NN_λ is a function that takes a sample and returns its nearest neighbors containing λ, let D' be the enlarged MLD with the synthetic instances included, defined as $D' = D \cup \{(\Phi_\lambda(s), \Lambda_\lambda(s)) : s \in S_\lambda,\ \lambda \in M\}$, where:

$$\Phi_\lambda :\ S_\lambda \to \mathcal{F}$$
$$(f,O) \mapsto h\, given\ by\ h_i = \begin{cases} rf_i + (1-r)g_i / r \in [0,1], (g,O') \in NN_\lambda(s) & F_i\ is\ numeric, \\ \underset{v\in F_i}{\operatorname{argmax}} \left|\{(g,O') \in NN_\lambda(s) : g_i = v\}\right| & F_i\ is\ notnumeric. \end{cases} \tag{3}$$

and

$$\Lambda_\lambda :\ S_\lambda \to \mathcal{P}(L)$$
$$(f,O) \mapsto \left\{ \mu \in L : \begin{array}{l} |\{(g,O') \in NN_\lambda(s) : \mu \in O'\}| > \frac{|NN_\lambda(s)|-1}{2} \wedge \mu \in O \\ or \\ |\{(g,O') \in NN_\lambda(s) : \mu \in O'\}| > \frac{|NN_\lambda(s)|+1}{2} \end{array} \right\} \tag{4}$$

4.6. MLSMOTE computational complexity

MLSMOTE iterates all the samples in the processed MLD m times, being m the number of minority labels and thus $m < |L|$. Only the s samples in which the processed minority label appears are taken as seeds. Usually $s \ll N$. For each seed its nearest neighbors among those s instances are searched, and a synthetic instance is generated. Therefore the computational complexity of MLSMOTE would be $O(ms^2)$.

In the next Section MLSMOTE will be experimentally compared with other oversampling algorithms, such as LP-ROS, ML-ROS and SmoteUG, previously described in Section 3.3. The computational complexity of these methods is as follows:

- LP-ROS: This algorithm generates clones of samples corresponding to the less frequent labelsets in an MLD. Theoretically the number of different labelsets could be $2^{|L|}$, but in practice this number is usually limited by N, the number of samples in the MLD. Since this methods generates $P\%$ of new instances, its complexity would be bounded by $O(NP)$.
- ML-ROS: The inner workings of these method are pretty similar to those followed by MLSMOTE. Firstly the m minority labels are found, and then the s samples in the MLD in which they appear are cloned. The number of clones is determined by the P parameter, as in LP-ROS. Hence, its computational complexity would be $O(msP)$.
- SmoteUG: Although this method is also based on SMOTE, it only takes into account one minority label. Therefore m will be 1. The algorithms searches for all the labelsets in which this minority label appears, and then looks for the instances containing each labelset. Any label in an MLD could appear in $2^{|L|-1}$ labelsets, and the number of instances is bounded by N. Since each seed sample will be compared to the remaining instances in the MLD, to get the nearest neighbors, the theoretical complexity of this method would be $O(2^{|L|-1}N^2)$.

5. Experimental study

In order to assess the benefits of MLSMOTE, an extensive experimental study was conducted. It is structured in two stages, using in all of them the same set of MLDs. These phases aim to answer two esencial questions, Is MLSMOTE able to improve the original classification results produced by MLC algorithms? and, How it performs against other published proposals? The two steps were conducted as follows:

- First, the influence of MLSMOTE on imbalance levels (Section 5.2) and the improvement produced by MLSMOTE over classification results (Section 5.3) are empirically assessed. Classification results after preprocessing are compared with those obtained from the MLDs without resampling, and how MLSMOTE influences each MLC algorithm is analyzed.
- Second, in Section 5.4, the results produced by MLSMOTE are compared against those obtained with other multilabel oversampling algorithms, such as LP-ROS, ML-ROS and SmoteUG,[1] all of them defined in Section 3.3, as well as against those produced by the BR-IRUS and EML algorithms mentioned in Section 3.2, which are specifically designed to face the imbalance problem.

In the next subsection the experimental framework used is described, while the following subsections analyze in detail the results obtained in each one of these stages.

[1] The EML [9], BR-IRUS [10], and SmoteUG [13] algorithms were carefully implemented following the descriptions given in their respective papers.

5.1. Experimental framework

The effectiveness of a resampling method such as MLSMOTE, which generates synthetic instances taking into account features and labels of other instances, should be very influenced by the MLDs' characteristics. Thus, testing the proposed algorithm against several MLDs with different traits is essential. Most of the MLDs used in a handful of published studies have been included in this experimentation. Their characteristics, including imbalance measures, are shown in Table 3. The MLDs' origin is offered in the rightmost column.

Each MLD was preprocessed with MLSMOTE, LP-ROS, ML-ROS and SmoteUG. All datasets were given as input to five MLC algorithms, applying a 2×5 folds cross validation scheme. Five well-known MLC algorithms were used to test the effect of the rebalancing produced by MLSMOTE. These are BR [19] (the basic binary relevance transformation), IBLR-ML [49] (an improvement of MLkNN [21]), HOMER [27] and RAkEL [25] (ensembles of LP classifiers) and CLR [26] (a pair-wise binary classifier). C4.5 was used as underlying classifier where needed, and default parameters were chosen for every classifier. The goal is to analyze the impact that MLSMOTE has over the most used transformation method (BR), one instance-based MLC classifier (IBLR-ML), and three of the MLC classifiers considered as state-of-the-art. Average results were obtained for each algorithm-dataset combination.

The outputs predicted for each classifier-dataset combination, always over test partitions, were assessed with two different evaluation measures. In this study context, assuming that some labels are in minority against the others, label-based evaluation measures are the most adequate. There are two ways to calculate this type of measures, known as macro-averaging and micro-averaging, shown in Eqs. (5) and (6). As stated in [63], macro-averaged measures are more sensitive to the performance of minority labels than micro-averaging.

$$MacroM = \frac{1}{|L|}\sum_{i=1}^{|L|} evalM(TP_i, FP_i, TN_i, FN_i) \tag{5}$$

$$MicroM = evalM\left(\sum_{i=1}^{|L|} TP_i, \sum_{i=1}^{|L|} FP_i, \sum_{i=1}^{|L|} TN_i, \sum_{i=1}^{|L|} FN_i\right) \tag{6}$$

Almost all evaluation measures, including accuracy, precision and recall, can be calculated following the two previous approaches. For the sake of space, F-measure (7), the harmonic mean of precision (8) and recall (9), was used. Therefore two measures are obtained for each run, MacroFM and MicroFM, as specific measures for imbalanced multilabel classification, where *evalM* is considered as the F-measure in Eq. (7), Y_i being the set of true labels and $h(x_i)$ the predicted ones for the *i*th sample of the MLD D

$$F-measure = 2 * \frac{Precision * Recall}{Precision + Recall}. \tag{7}$$

$$Precision = \frac{1}{|D|}\sum_{i=1}^{|D|} \frac{|Y_i \cap h(x_i)|}{|h(x_i)|}. \tag{8}$$

$$Recall = \frac{1}{|D|}\sum_{i=1}^{|D|} \frac{|Y_i \cap h(x_i)|}{|Y_i|}. \tag{9}$$

The significance of results obtained in each phase is established by the appropriate statistical tests. Two different situations can be found:

- When the analysis implies results from only two sources, for instance results before and after preprocessing, the Wilcoxon [64] non-parametric sign rank test is used. This is analogous to the parametric paired *T*-test.
- For multiple comparisons the statistical analysis is performed in two steps. Firstly the Friedman test is used to rank the methods, and to establish if any statistical differences exist. Secondly, a multiple comparison using Benjamini and Hochberg's FDR procedure [65] is completed. This test is a step-up procedure to control the false discovery rate in multiple comparison scenarios.

The tests were executed using the `wilcoxsign_test` and `pairwise.table` functions in R `coin` and `stats` statistical packages, and exact *p-values* are reported.

5.2. How MLSMOTE influences the imbalance levels

After applying MLSMOTE to the MLDs, the imbalance levels for each one were reassessed. The goal is to analyze how MLSMOTE has influenced the label distributions in the MLDs. Table 4 shows for each MLD the *MaxIR* and *MeanIR* measures before and after applying MLSMOTE, as well as the percentage of change for these measures in each case. It should be noted that the values shown in Table 4 were obtained from the training partitions used in experimentation, since MLSMOTE is applied only to training partitions, whereas the imbalance data previously shown in Table 3 correspond to the full datasets.

As can be observed in Table 4, the *MaxIR* has slightly increased (+0.29% to +3.04%) in four of the MLDs, while the remainder ones have experimented a remarkable reduction. This implies that the ratio between the most frequent label and the less frequent one has been improved in almost all MLDs. The *MeanIR*, the average imbalance level for all labels, has decreased for all MLDs with the exception of emotions. This dataset is not actually imbalanced, as

Table 3
Characteristics of the datasets used in the experimentation.

Dataset	Card	Number of			Imbalance ratio		Ref.
		Samples	Features	Labels	Max	Mean	
bibtex	2.402	7395	1836	159	20.43	12.50	[52]
cal500	26.044	502	68	174	88.80	20.58	[53]
corel5k	3.522	5000	499	374	1120.00	189.57	[54]
corel16k	2.867	13766	500	161	126.80	34.16	[55]
emotions	1.869	593	72	6	1.78	1.48	[56]
enron	3.378	1702	753	53	913.00	73.95	[57]
genbase	1.252	662	1186	27	171.00	37.31	[58]
mediamill	4.376	43907	120	101	1092.55	256.40	[59]
medical	1.245	978	1449	45	266.00	89.50	[60]
scene	1.074	2407	294	6	1.46	1.25	[19]
slashdot	1.181	3782	1079	22	194.67	19.46	[61]
tmc2007	2.158	28596	49060	22	41.98	17.13	[62]

Table 4
Imbalance level changes after preprocessing the MLDs.

Dataset	MaxIR			MeanIR		
	Before	After	%Δ	Before	After	%Δ
bibtex	22.81	15.66	−31.32	12.54	12.08	−3.68
cal500	133.19	118.74	−10.85	21.27	20.90	−1.78
corel5k	896.00	923.20	3.04	168.78	142.16	−15.77
corel16k	134.73	69.55	−48.38	34.32	25.67	−25.22
emotions	1.78	1.71	−4.21	1.48	1.52	2.80
enron	657.05	674.05	2.59	72.77	62.58	−14.01
genbase	136.80	137.20	0.29	32.41	28.95	−10.68
mediamill	1122.32	564.55	−49.70	257.46	159.15	−38.18
medical	212.80	214.90	0.99	72.17	69.56	−3.62
scene	1.46	1.25	−14.44	1.25	1.20	−4.21
slashdot	202.73	99.76	−50.79	20.03	11.24	−43.88
tmc2007	42.01	23.50	−44.06	17.14	12.34	−28.00

its original *MeanIR* denotes. For the most imbalanced MLDs the improvement is between −14% to −43%.

Generally speaking, from these results it can be drawn that MLSMOTE produces an amelioration on imbalance levels. Notwithstanding, the change in imbalance level will not necessarily implies better classification results. The decisive factor for obtaining better predictions will be how these new instances change the model built by the classifier. This is the fact assessed in the following section.

5.3. MLSMOTE vs base results comparison

The next step will be the analysis of results produced by the MLC classifiers before and after MLSMOTE is applied. Therefore, in this case there are only two sets of results, one produced by the classifiers from the MLDs without resampling and another one obtained from the same classifiers using the MLDs after being processed by MLSMOTE. All these results are shown in Table 5 (MacroFM) and Table 6 (MicroFM).

Table 7
Wilcoxon statistical tests analyzing differences after applying MLSMOTE.

	MacroFM		MicroFM	
	z-score	*p*-value	*z*-score	*p*-value
BR	−1.76640	0.04199	−2.16562	0.01465
CLR	−2.85499	0.00195	−2.55246	0.00342
HOMER	0.39223	0.66138	0.94136	0.83032
IBLR	−1.37387	0.09473	−0.82432	0.22314
RAkEL	−1.76640	0.04199	−2.16562	0.01465

In order to assess the statistical significance of the analyzed results, the Wilcoxon statistical test was applied stating that MacroFM/MicroFM is higher after preprocessing as alternative hypothesis. The *z-score* and *p-value* outputs of these tests are shown in Table 7. From its analysis the following facts can be deducted:

- The observed *p-values* denote that CLR, BR and RAkEL algorithms, as well as IBLR when MacroFM is used, significantly improved their results after applying MLSMOTE. CLR is the most benefited classifier (lowest *p-value* and *z-score*). As can be seen in Table 5 (MacroFM), for CLR the results have been improved in all cases, whereas in Table 6 (MicroFM) there is only one case without improvement. The behavior of BR and RAkEL after applying MLSMOTE is very similar, and all *p-values* are under the 0.1 threshold.
- Although the behavior of IBLR when evaluated with MicroFM has experimented a slight enhancement, it hardly can be

Table 5
Results before and after applying MLSMOTE – MacroFM.

Dataset	BR		CLR		HOMER		IBLR		RAkEL	
	Before	After	Before	After	Before	After	Before	After	Before	After
bibtex	0.3368	**0.3456**	0.3342	**0.3429**	**0.3042**	0.2989	0.2140	**0.2479**	0.3368	**0.3456**
cal500	0.2934	**0.3124**	0.3323	**0.3474**	**0.3316**	0.3139	0.2772	**0.2783**	0.2934	**0.3124**
corel16k	0.1336	**0.1347**	0.1003	**0.1040**	0.1363	**0.1408**	**0.1141**	0.0971	0.1277	**0.1288**
corel5k	0.1774	**0.1790**	0.1330	0.1330	0.1916	**0.1923**	0.1059	**0.1130**	0.1774	**0.1790**
emotions	0.5712	**0.5841**	0.5982	**0.6107**	0.5642	**0.5733**	0.6487	**0.6511**	0.5712	**0.5841**
enron	**0.4029**	0.3936	0.4198	**0.4272**	**0.3790**	0.3740	0.3458	**0.3580**	**0.4029**	0.3936
genbase	0.9890	**0.9895**	0.9848	**0.9853**	0.9780	**0.9839**	0.9655	**0.9688**	0.9890	**0.9895**
mediamill	**0.2774**	0.2737	0.2276	**0.2308**	0.2404	**0.2424**	0.2806	**0.2862**	**0.2774**	0.2737
medical	0.8166	0.8166	0.7942	0.7942	**0.7942**	0.7786	0.6404	0.6404	0.8166	0.8166
scene	0.6314	**0.6328**	0.6400	**0.6407**	0.6113	**0.6212**	0.7427	**0.7456**	0.6314	**0.6328**
slashdot	0.4038	**0.4044**	0.3982	0.3982	**0.3996**	0.3885	**0.2382**	0.1852	0.4038	**0.4044**
tmc2007	0.6015	**0.6165**	0.6073	**0.6242**	0.5968	**0.6003**	0.4668	**0.4924**	0.6015	**0.6165**

Best values are highlighted in bold.

Table 6
Results before and after applying MLSMOTE – MicroFM.

Dataset	BR		CLR		HOMER		IBLR		RAkEL	
	Before	After	Before	After	Before	After	Before	After	Before	After
bibtex	0.4021	**0.4097**	0.3371	**0.3484**	**0.3568**	0.3546	0.2628	**0.2701**	0.4021	**0.4097**
cal500	0.3488	**0.3662**	0.2977	**0.3619**	**0.3978**	0.3744	0.3184	**0.3388**	0.3488	**0.3662**
corel16k	0.1156	0.1156	**0.0846**	0.0832	0.1866	**0.1878**	**0.0504**	0.0484	0.1145	0.1145
corel5k	0.1096	**0.1105**	0.0706	**0.0707**	**0.1744**	0.1702	**0.0542**	0.0535	0.1096	**0.1102**
emotions	0.5845	**0.5958**	0.6072	**0.6174**	0.5766	**0.5818**	0.6730	**0.6756**	0.5845	**0.5958**
enron	**0.5334**	0.5324	0.5596	**0.5606**	**0.5265**	0.5199	0.4561	**0.4660**	**0.5334**	0.5324
genbase	0.9867	**0.9873**	0.9852	**0.9858**	0.9820	**0.9827**	0.9768	**0.9771**	0.9867	**0.9873**
mediamill	**0.5622**	0.5618	0.5928	**0.5938**	**0.5493**	0.5482	0.5987	**0.6000**	**0.5622**	0.5618
medical	0.8006	0.8006	0.7965	0.7965	**0.7994**	0.7955	0.6324	0.6324	0.8006	0.8006
scene	0.6190	**0.6231**	0.6254	**0.6275**	0.6010	**0.6132**	0.7366	**0.7398**	0.6190	**0.6231**
slashdot	0.4598	**0.5339**	0.4416	**0.4449**	0.4429	**0.4436**	**0.2042**	0.1470	0.4598	**0.5339**
tmc2007	0.7063	**0.7071**	0.7228	**0.7248**	**0.6982**	0.6956	**0.6447**	0.6378	0.7063	**0.7071**

Best values are highlighted in bold.

considered statistically significant if we set the *p-value* threshold to the usual 0.1 or 0.05. However, this black or white approach to *p-values* interpretation can be broadened to shades of gray. Statistical non-significance does not imply equivalence. MLSMOTE is performing well with both evaluation measures for IBLR. In fact, classification results for MacroFM have improved in 10 out of 12 MLDs, and for MicroFM in 8 out of 12.
- Finally, according to Wilcoxon test results, HOMER is not benefiting from the resampling. Although for MacroFM the results are better in 7 out of 12 MLDs, for MicroFM the ratio is the opposite.

Overall, this first experimental stage determines that MLSMOTE is able to accomplish a general improvement in classification results when used along with CLR, RAkEL, BR and, to a less extent, IBLR. Even though there is not statistical significance when MicroFM is used, MLSMOTE in general has a positive influence over IBLR. Finally, MLSMOTE is not advisable if HOMER is going to be used as classifier, since the applied oversampling tends to deteriorate its predictions.

Delving into the classifiers behavior, IBLR and HOMER share a common characteristic, both take advantage of local information to do their work. IBLR uses the nearest neighbors to make its prediction, whereas HOMER relies in the same technique in order to cluster instances and obtain subsets of labels. In fact, HOMER and RAkEL are very closer algorithms, since both train several multiclass classifiers using subsets of labels. The main difference is in the method used to built these labels subsets. Where HOMER uses local information to accomplish this task, RAkEL does it randomly. From the analysis of results that the synthetic samples produced by MLSMOTE have a more positive influence over classifiers such as BR, CLR and RAkEL, whose behavior is not biased by the selection of only a few nearest instances, can be concluded.

Table 8
Oversampling methods to compare.

Algorithm	# Min. labels	Features	Labelset
LP-ROS [16]	NA	Cloned	Cloned
ML-ROS [12]	Several	Cloned	Cloned
SmoteUG [13]	1	Synthetic	Cloned
MLSMOTE	Several	Synthetic	Synthetic

5.4. MLSMOTE vs other imbalanced MLC proposals

Our goal in the second phase of experimentation is to compare the performance of MLSMOTE against other published proposals to face imbalance in MLC. In subSection 3.2 the EML and BR-IRUS algorithms were introduced, as ensemble-based approaches to the imbalanced MLC task. In subSection 3.3 three already published multilabel oversampling algorithms were also referenced. All of them generate new instances, albeit following different strategies. Table 8 summarizes these methods characteristics, stating how many minority labels are taken into account, and how the new instances features and labelset are generated.

EML and BR-IRUS are able to produce classification predictions by themselves, but MLSMOTE, ML-ROS, LP-ROS and SmoteUG are not. Since they are resampling algorithms, their output has to be given to an existent MLC. For this reason the first step will be to determine which MLC algorithms presents the best behavior for each resampling method. After that, classification results produced by all configurations will be compared.

5.4.1. Selecting the best classifier for each resampling method

Aiming to learn how the MLC algorithms behave while working with each resampling method, classification results produced by them after preprocessing the MLDs were ranked by the Friedman statistical test and the average rank for each algorithm was obtained (see Tables 9 and 10). The results obtained by the same classifiers before preprocessing have been also included. From these tables observation it can be derived the following:

- Unsurprisingly, BR is the classifier that achieves better results before preprocessing and also after applying some of the resampling methods, including SmoteUG and MLSMOTE. As stated in [66], BR is a not so simple approach to multilabel classification, being able to produce good classification results and being a competitive option against other MLC classifiers.
- The classifier with the best behavior for LP-ROS and ML-ROS oversampling algorithms is CLR, which appears better ranked than BR for the two evaluation measures with the exception of ML-ROS/MicroFM.
- HOMER and IBLR are the worst classifiers in all cases, both before and after resampling the MLDs.

Table 9
Average rankings for each classifier before and after applying resampling (MacroFMeasure).

Before resampling		MLSMOTE		ML-ROS		LP-ROS		SmoteUG	
Rank	Classifier	Rank	Classifier	Rank	Classifier	Rank	Classifier	Rank	Classifier
2.3750	BR	2.4167	BR	2.5000	CLR	1.7500	CLR	2.2917	BR
2.4583	RAkEL	2.4167	RAkEL	2.5417	BR	2.7083	BR	2.4583	RAkEL
2.8750	CLR	2.6667	CLR	2.6250	RAkEL	2.7917	RAkEL	2.5833	CLR
3.3750	HOMER	3.5000	HOMER	3.4167	HOMER	3.8333	HOMER	3.5000	HOMER
3.9167	IBLR	4.0000	IBLR	3.9167	IBLR	3.9167	IBLR	4.1667	IBLR

Table 10
Average rankings for each classifier before and after applying resampling (MicroFMeasure).

Before resampling		MLSMOTE		ML-ROS		LP-ROS		SmoteUG	
Rank	Classifier	Rank	Classifier	Rank	Classifier	Rank	Classifier	Rank	Classifier
2.3750	BR	2.3750	BR	2.3750	BR	2.4167	CLR	2.3750	BR
2.4583	RAkEL	2.4583	RAkEL	2.4583	RAkEL	2.5833	BR	2.4583	RAkEL
3.0000	CLR	2.7500	CLR	2.7083	CLR	2.5833	RAkEL	2.8333	CLR
3.2500	HOMER	3.4167	HOMER	3.3750	HOMER	3.5833	HOMER	3.4167	HOMER
3.9167	IBLR	4.0000	IBLR	4.0833	IBLR	3.8333	IBLR	3.9167	IBLR

5.4.2. Classification results comparison

The final experimental stage compares the five previous proposals with MLSMOTE, aiming to elucidate how it performs against them. EML and BR-IRUS are MLC classifiers by themselves, but all the other methods, including MLSMOTE, need a classifier to work with the preprocessed MLDs. The best classifier for each resampling method has been chosen, following the rankings discussed above. Classification results produced by these six methods are presented in Tables 11 and 12. Best values for each measure are highlighted in bold.

At first sight, that BR-IRUS and LP-ROS are clearly the methods producing the worst results can be observed. Furthermore, MLSMOTE achieves the highest number of wins (11), followed by EML (6) and SmoteUG and ML-ROS (5). Regarding the statistical analysis, once the Friedman test confirms that some statistical difference exists among the algorithms, pairwise comparisons are performed using Benjamini and Hochberg's procedure. The best ranked method, which for the two considered measures is MLSMOTE, is taken as control algorithm. The rankings produced by the Friedman test and corresponding *p-values* are shown in Tables 13 and 14. As can be seen in the those tables, MLSMOTE is clearly working better than the other five proposals, as always achieves the best ranking position. That a clear statistical difference between MLSMOTE with respect to BR-IRUS, LP-ROS and SmoteUG exists can be concluded. MLSMOTE also outperforms EML with significant differences when MacroFM is used. With MicroFM MLSMOTE achieves a better ranking than EML, but without statistically significant differences. Despite the *p-values* reported for ML-ROS, the ranking and number of best values indicate than, in average, MLSMOTE is also superior to it.

It must be noted that MicroFM is a evaluation measure very influenced by correct classification of majority labels. On the other hand, MacroFM is more sensitive to the minority ones. Therefore, the previous statistical results allow to conclude that MLSMOTE is benefiting the classification of minority labels more than EML, even though globally the results produced by EML are very good. This fact can be confirmed in the results shown in Tables 11 and 12. EML obtains best values with MLDs such as cal500, emotions and scene, whose common characteristic is their low *MeanIR*, while MLSMOTE works better with MLDs having higher imbalance levels, such as corel5k, corel16k and medical.

Another factor to be taken into account is the computing resources needed to run each solution, specifically memory and running time. As was pointed out in Section 3.2, both BR-IRUS

Table 11
Classification results MLSMOTE vs other proposals (MacroFM).

Dataset	MLSMOTE + BR	ML-ROS + CLR	LP-ROS + CLR	SmoteUG + BR	EML	BR-IRUS
bibtex	**0.3456**	0.3386	0.2927	0.3375	0.1265	0.0462
cal500	0.3124	0.3202	**0.3236**	0.2934	0.1291	0.2178
corel16k	**0.1347**	0.1033	0.1059	**0.1347**	0.0181	0.0349
corel5k	**0.1790**	0.1355	0.1366	0.1774	0.0133	0.0226
emotions	0.5841	0.6062	0.5684	0.5724	**0.6893**	0.5160
enron	0.3936	**0.4220**	0.3819	0.4029	0.1241	0.1028
genbase	**0.9895**	0.9800	0.9732	0.9890	0.7608	0.8664
mediamill	0.2737	0.2322	0.2020	**0.2861**	0.1728	0.0556
medical	**0.8166**	0.7865	0.6920	**0.8166**	0.2770	0.3958
scene	0.6328	0.6387	0.5930	0.6314	**0.7546**	0.3673
slashdot	0.4044	**0.4061**	0.3410	0.4044	0.3560	0.1142
tmc2007	0.6165	**0.6332**	0.5731	0.5933	0.5122	0.1370

Table 12
Classification results MLSMOTE vs other proposals (MicroFM).

Dataset	MLSMOTE + BR	ML-ROS + CLR	LP-ROS + CLR	SmoteUG + BR	EML	BR-IRUS
bibtex	**0.4097**	0.3457	0.2972	0.4024	0.2888	0.0331
cal500	0.3662	0.3348	0.3510	0.3488	**0.4106**	0.2238
corel16k	**0.1156**	0.0894	0.1060	**0.1156**	0.0544	0.0367
corel5k	**0.1105**	0.0764	0.0804	0.1096	0.0712	0.0205
emotions	0.5958	0.6152	0.5755	0.5860	**0.7015**	0.5178
enron	0.5324	**0.5552**	0.5038	0.5334	0.5542	0.1030
genbase	**0.9873**	0.9854	0.9782	0.9867	0.9854	0.6576
mediamill	0.5618	0.6006	0.5524	0.5686	**0.6277**	0.0633
medical	**0.8006**	0.7866	0.6970	**0.8006**	0.7581	0.1392
scene	0.6231	0.6240	0.5784	0.6190	**0.7468**	0.3658
slashdot	**0.5339**	0.4456	0.3232	0.4644	0.4942	0.1062
tmc2007	0.7071	**0.7250**	0.6298	0.7046	0.7065	0.1456

Table 13
Average ranking produced by Friedman statistical test and *p-values* (MacroFM).

Pos	Algorithm	Rank	*p-value*
1	MLSMOTE + BR	2.04167	**
2	ML-ROS + CLR	2.33333	0.24452
3	SmoteUG + BR	2.54167	0.11719
4	LP-ROS + CLR	3.83333	0.00146
5	EML	4.66667	0.01968
6	BR-IRUS	5.58333	0.00092

Table 14
Average ranking produced by Friedman statistical test and *p-values* (MicroFM).

Pos	Algorithm	Rank	*p-value*
1	MLSMOTE + BR	2.08333	**
2	EML	2.79167	0.86243
3	SmoteUG + BR	2.83333	0.05310
4	ML-ROS + CLR	2.87500	0.28280
5	LP-ROS + CLR	4.41667	0.00061
6	BR-IRUS	6.00000	0.00061

and EML have to train multiple classifiers, using all or part of the training data. Thus, the memory needed increases, as do the time used to build the classifiers. On the contrary, the results reported above for MLSMOTE are obtained by a single MLC algorithm, trained once. Overall, the solution based on MLSMOTE produces better classification results than the other approaches, while using less resources.

Overall, MLSMOTE seems to be the most forceful option to face imbalanced multilabel classification. Despite the good position for EML when using MicroFM (2nd best), it falls to fifth position with MacroFM. Something similar happens to ML-ROS, which achieves 2nd position when using MacroFM, but it falls to fourth with MicroFM. MLSMOTE obtains the best result for both measures.

6. Concluding remarks

In this paper MLSMOTE, a multilabel synthetic minority oversampling technique, has been presented, along with three strategies for synthetic labelset generation. A review on how the imbalance problem in the multilabel classification has been faced in the past is provided, and how MLSMOTE differs from previous approaches has been discussed.

From the conducted experimentation we can conclude that MLSMOTE, using Rank as labelset generation method, achieves a statistically significant improvement against the results obtained without preprocessing for the best performers MLC algorithms, such as BR, RAkEL and CLR. Using this recommended configuration, MLSMOTE is able to reduce the imbalance level in MLDs, thus improving the predictions made by the MLC algorithms. These results has been endorsed by the proper statistical tests. Although the results produced by IBLR have also improved, in general MLSMOTE would not be advisable for MLC algorithms relying on local information, such as IBLR and HOMER.

Furthermore, MLSMOTE has been compared with other multilabel oversampling algorithms, as well as against imbalance-aware MLC algorithms. The experimentation results show that MLSMOTE outperforms all other oversampling methods, accomplishing a statistically significant difference against most of them. Regarding the ensemble-based MLC algorithms, MLSMOTE performed far better than BR-IRUS. Compared with the EML method, which could be considered the best one in its category, MLSMOTE performed slightly better when assessed using MicroFM, and was remarkably superior using the MacroFM evaluation measure. It must be highlighted the remarkable advantage of MLSMOTE over EML while working with highly imbalanced MLDs. This fact encourage us to recommend the use of MLSMOTE with this kind of datasets.

Acknowledgments

We would like to thank the anonymous reviewers, whose suggestions contributed to improve this paper. The work of F. Charte was supported by the Spanish Ministry of Education under the FPU National Program (Ref. AP2010-0068). This work was partially supported by the Spanish Ministry of Science and Technology under Projects TIN2011-28488 and TIN2012-33856, and the Andalusian regional Projects P10-TIC-06858 and P11-TIC-7765.

References

[1] R. Duda, P. Hart, D. Stork, Pattern Classification, second ed., John Wiley, 2000.
[2] G. Tsoumakas, I. Katakis, I. Vlahavas, Mining multi-label data, in: O. Maimon, L. Rokach (Eds.), Data Mining and Knowledge Discovery Handbook, Springer, US, Boston, MA, 2010, pp. 667–685, http://dx.doi.org/10.1007/978-0-387-09823-4_34 (Chapter 34).
[3] L. Zhuang, H. Dai, X. Hang, A novel field learning algorithm for dual imbalance text classification, in: Fuzzy Systems and Knowledge Discovery, LNCS, vol. 3614, 2005, pp. 39–48.
[4] T. Fawcett, F. Provost, Adaptive fraud detection, Data Min. Knowl. Discov. 1 (1997) 291–316.
[5] N.V. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, SIGKDD Explor. Newsl. 6 (1) (2004) 1–6, http://dx.doi.org/10.1145/1007730.1007733.
[6] J. He, H. Gu, W. Liu, Imbalanced multi-modal multi-label learning for subcellular localization prediction of human proteins with both single and multiple sites, PloS one 7 (6) (2012) 7155, http://dx.doi.org/10.1371/journal.pone.0037155.
[7] C. Li, G. Shi, Improvement of learning algorithm for the multi-instance multi-label RBF neural networks trained with imbalanced samples, J. Inf. Sci. Eng. 29 (4) (2013) 765–776.
[8] G. Tepvorachai, C. Papachristou, Multi-label imbalanced data enrichment process in neural net classifier training, in: IEEE Int. Joint Conf. on Neural Networks, 2008. IJCNN, 2008, pp. 1301–1307. http://dx.doi.org/10.1109/IJCNN.2008.4633966.
[9] M.A. Tahir, J. Kittler, A. Bouridane, Multilabel classification using heterogeneous ensemble of multi-label classifiers, Pattern Recogn. Lett. 33 (5) (2012) 513–523, http://dx.doi.org/10.1016/j.patrec.2011.10.019.
[10] M.A. Tahir, J. Kittler, F. Yan, Inverse random under sampling for class imbalance problem and its application to multi-label classification, Pattern Recogn. 45 (10) (2012) 3738–3750, http://dx.doi.org/10.1016/j.patcog.2012.03.014.
[11] S. Dendamrongvit, M. Kubat, Undersampling approach for imbalanced training sets and induction from multi-label text-categorization domains, in: New Frontiers in Applied Data Mining, LNCS, bol. 5669, Springer, 2010, pp. 40–52, http://dx.doi.org/10.1007/978-3-642-14640-4_4.
[12] F. Charte, A.J. Rivera, M.J. del Jesus, F. Herrera, Addressing imbalance in multilabel classification: measures and random resampling algorithms, Neurocomputing 163 (9) (2015) 3–16, http://dx.doi.org/10.1016/j.neucom.2014.08.091.
[13] A.F. Giraldo-Forero, J.A. Jaramillo-Garzón, J.F. Ruiz-Muñoz, C.G. Castellanos-Domínguez, Managing imbalanced data sets in multi-label problems: a case study with the SMOTE algorithm, in: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, LNCS, vol. 8258, Springer, 2013, pp. 334–342, http://dx.doi.org/10.1007/978-3-642-41822-8_42.
[14] V. García, J. Sánchez, R. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowl. Based Syst. 25 (1) (2012) 13–21, http://dx.doi.org/10.1016/j.knosys.2011.06.013.
[15] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357, http://dx.doi.org/10.1613/jair.953.
[16] F. Charte, A. Rivera, M.J. Jesus, F. Herrera, A first approach to deal with imbalance in multi-label datasets, in: Proc. 8th Int. Conf. Hybrid Artificial Intelligent Systems, Salamanca, Spain, HAIS'13, LNCS, 2013, vol. 8073, pp. 150–160. http://dx.doi.org/10.1007/978-3-642-40846-5_16.
[17] A. de Carvalho, A. Freitas, A tutorial on multi-label classification techniques, in: Found. Computational Intell, vol. 5, 2009, pp. 177–195 (Chapter 8). http://dx.doi.org/10.1007/978-3-642-01536-6_8.
[18] S. Godbole, S. Sarawagi, Discriminative methods for multi-labeled classification, in: Advances in Knowl. Discovery and Data Mining, vol. 3056, 2004, pp. 22–30. http://dx.doi.org/10.1007/978-3-540-24775-3_5.
[19] M. Boutell, J. Luo, X. Shen, C. Brown, Learning multi-label scene classification, Pattern Recogn. 37 (9) (2004) 1757–1771, http://dx.doi.org/10.1016/j.patcog.2004.03.009.
[20] Q. Wu, Y. Ye, H. Zhang, T. Chow, S.-S. Ho, ML-TREE: a tree-structure-based approach to multilabel learning, IEEE Trans. Neural Netw. Learn. Syst. (2014), http://dx.doi.org/10.1109/TNNLS.2014.2315296.
[21] M. Zhang, Z. Zhou, ML-KNN: a lazy learning approach to multi-label learning, Pattern Recogn. 40 (7) (2007) 2038–2048, http://dx.doi.org/10.1016/j.patcog.2006.12.019.
[22] M.-L. Zhang, Multilabel neural networks with applications to functional genomics and text categorization, IEEE Trans. Knowl. Data Eng. 18 (10) (2006) 1338–1351, http://dx.doi.org/10.1109/TKDE.2006.162.
[23] M.-L. Zhang, ML-RBF: RBF neural networks for multi-label learning, Neural Process. Lett. 29 (2009) 61–74, http://dx.doi.org/10.1007/s11063-009-9095-3.
[24] A. Elisseeff, J. Weston, A kernel method for multi-labelled classification, Advances in Neural Information Processing Systems, vol. 14, MIT Press, 2001, pp. 681–687.
[25] G. Tsoumakas, I. Vlahavas, Random k-labelsets: an ensemble method for multilabel classification, in: Proc. 18th European Conf. on Machine Learning, Warsaw, Poland, ECML'07, vol. 4701, 2007, pp. 406–417. http://dx.doi.org/10.1007/978-3-540-74958-5_38.
[26] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, K. Brinker, Multilabel classification via calibrated label ranking, Mach. Learn. 73 (2008) 133–153, http://dx.doi.org/10.1007/s10994-008-5064-8.
[27] G. Tsoumakas, I. Katakis, I. Vlahavas, Effective and efficient multilabel classification in domains with large number of labels, in: Proc. ECML/PKDD Workshop on Mining Multidimensional Data, Antwerp, Belgium, MMD'08, 2008, pp. 30–44.
[28] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Mach. Learn. 85 (2011) 333–359, http://dx.doi.org/10.1007/s10994-011-5256-5.
[29] J. Read, B. Pfahringer, G. Holmes, Multi-label classification using ensembles of pruned sets, in: Proc. 8th IEEE Int. Conf. on Data Mining, Pisa, Italy, ICDM'08, 2008, pp. 995–1000.

[30] C.-S. Ferng, H.-T. Lin, Multilabel classification using error-correcting codes of hard or soft bits, IEEE Trans. Neural Netw. Learn. Syst 24 (11) (2013) 1888–1900, http://dx.doi.org/10.1109/TNNLS.2013.2269615.
[31] M. Zhang, Z. Zhou, A review on multi-label learning algorithms, IEEE Trans. Knowl. Data Eng. 26 (8) (2014) 1819–1837, http://dx.doi.org/10.1109/TKDE.2013.39.
[32] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intell. Data Anal. 6 (5) (2002) 429–449.
[33] T. Khoshgoftaar, J. Van Hulse, A. Napolitano, Supervised neural network modeling: an empirical investigation into learning from imbalanced data with labeling errors, IEEE Trans. Neural Netw. Learn. Syst 21 (5) (2010) 813–830, http://dx.doi.org/10.1109/TNN.2010.2042730.
[34] V. López, A. Fernández, S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics, Inf. Sci. 250 (2013) 113–141, http://dx.doi.org/10.1016/j.ins.2013.07.007.
[35] S.B. Kotsiantis, P.E. Pintelas, Mixture of expert agents for handling imbalanced data sets, Ann. Math. Comput. Teleinform. 1 (2003) 46–55.
[36] M. Lin, K. Tang, X. Yao, Dynamic sampling approach to training neural networks for multiclass imbalance classification, IEEE Trans. Neural Netw. Learn. Syst 24 (4) (2013) 647–660, http://dx.doi.org/10.1109/TNNLS.2012.2228231.
[37] A. Fernández, V. López, M. Galar, M.J. del Jesus, F. Herrera, Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches, Knowl. Based Syst. 42 (2013) 97–110, http://dx.doi.org/10.1016/j.knosys.2013.01.018.
[38] F. Provost, T. Fawcett, Robust classification for imprecise environments, Mach. Learn. 42 (2001) 203–231, http://dx.doi.org/10.1023/A:1007601015854.
[39] H. He, Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications, Wiley-IEEE, 2013.
[40] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, An overview of ensemble methods for binary classifiers in multi-class problems: experimental study on one-vs-one and one-vs-all schemes, Pattern Recogn. 44 (8) (2011) 1761–1776, http://dx.doi.org/10.1016/j.patcog.2011.01.017.
[41] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: Bagging, boosting, and hybrid-based approaches, IEEE Trans. Syst. Man Cybern., Part C: Appl. Rev. 42 (4) (2012) 463–484, http://dx.doi.org/10.1109/TSMCC.2011.2161285.
[42] M. Galar, A. Fernández, E. Barrenechea, F. Herrera, EUSBoost: enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling, Pattern Recogn. 46 (12) (2013) 3460–3471, http://dx.doi.org/10.1016/j.patcog.2013.05.006.
[43] F. Charte, F.D. Charte, How to work with multilabel datasets in R using the mldr package. http://dx.doi.org/10.6084/m9.figshare.1356035.
[44] F. Charte, A. Rivera, M.J. Jesus, F. Herrera, Concurrence among imbalanced labels and its influence on multilabel resampling algorithms, in: Proc. 9th Int. Conf. Hybrid Artificial Intelligent Systems, Salamanca, Spain, HAIS'14, vol. 8480, LNCS, 2014.
[45] M.-L. Zhang, Z.-J. Wang, Mimlrbf: {RBF} neural networks for multi-instance multi-label learning, Neurocomputing 72 (16-18) (2009) 3951–3956, http://dx.doi.org/10.1016/j.neucom.2009.07.008.
[46] K. Chen, B.-L. Lu, J. Kwok, Efficient classification of multi-label and imbalanced data using min–max modular classifiers, in: Int. Joint Conf. Neural Networks, 2006, pp. 1770–1775. http://dx.doi.org/10.1109/IJCNN.2006.246893.
[47] B.-L. Lu, M. Ito, Task decomposition and module combination based on class relations: a modular neural network for pattern classification, IEEE Trans. Neural Networks 10 (5) (1999) 1244–1256, http://dx.doi.org/10.1109/72.788664.
[48] I. Jolliffe, Principal Component Analysis, John Wiley & Sons, Ltd., 2005.
[49] W. Cheng, E. Hüllermeier, Combining instance-based learning and logistic regression for multilabel classification, Mach. Learn. 76 (2-3) (2009) 211–225, http://dx.doi.org/10.1007/s10994-009-5127-5.
[50] C. Stanfill, D. Waltz, Toward memory-based reasoning, Commun. ACM 29 (12) (1986) 1213–1228, http://dx.doi.org/10.1145/7902.7906.
[51] F. Charte, A. Rivera, M. del Jesus, F. Herrera, LI-MLC: a label inference methodology for addressing high dimensionality in the label space for multilabel classification, IEEE Trans. Neural Networks Learn. Syst. 25 (10) (2014) 1842–1854, http://dx.doi.org/10.1109/TNNLS.2013.2296501.
[52] I. Katakis, G. Tsoumakas, I. Vlahavas, Multilabel text classification for automated tag suggestion, in: Proc. ECML PKDD'08 Discovery Challenge, Antwerp, Belgium, 2008, pp. 75–83.
[53] D. Turnbull, L. Barrington, D. Torres, G. Lanckriet, Semantic annotation and retrieval of music and sound effects, IEEE Audio, Speech, Lang. Process. 16 (2) (2008) 467–476, http://dx.doi.org/10.1109/TASL.2007.913750.
[54] P. Duygulu, K. Barnard, J. de Freitas, D. Forsyth, Object recognition as machine translation: learning a lexicon for a fixed image vocabulary, in: Proc. 7th European Conf. on Computer Vision-Part IV, Copenhagen, Denmark, ECCV'02, 2002, pp. 97–112. http://dx.doi.org/10.1007/3-540-47979-1_7.
[55] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D.M. Blei, M.I. Jordan, Matching words and pictures, J. Mach. Learn. Res. 3 (2003) 1107–1135.
[56] A. Wieczorkowska, P. Synak, Z. Raś, Multi-label classification of emotions in music, in: Intelligent Information Processing and Web Mining, vol. 35, AISC, 2006, pp. 307–315 (Chapter 30). http://dx.doi.org/10.1007/3-540-33521-8_30.
[57] B. Klimt, Y. Yang, The Enron Corpus: A new dataset for email classification research, in: Proc. ECML'04, Pisa, Italy, 2004, pp. 217–226. http://dx.doi.org/10.1007/978-3-540-30115-8_22.
[58] S. Diplaris, G. Tsoumakas, P. Mitkas, I. Vlahavas, Protein Classification with Multiple Algorithms, in: Proc. 10th Panhellenic Conference on Informatics, Volos, Greece, PCI'05, 2005, pp. 448–456. http://dx.doi.org/10.1007/11573036_42.
[59] C.G.M. Snoek, M. Worring, J.C. van Gemert, J.M. Geusebroek, A.W.M. Smeulders, The challenge problem for automated detection of 101 semantic concepts in multimedia, in: Proc. 14th Annu. ACM Int. Conf. on Multimedia, Santa Barbara, CA, USA, MULTIMEDIA'06, 2006, pp. 421–430. http://dx.doi.org/10.1145/1180639.1180727.
[60] K. Crammer, M. Dredze, K. Ganchev, P.P. Talukdar, S. Carroll, Automatic Code Assignment to Medical Text, in: Proc. Workshop on Biological, Translational, and Clinical Language Processing, Prague, Czech Republic, BioNLP'07, 2007, pp. 129–136.
[61] J. Read, P. Reutemann, MEKA multi-label dataset repository. <http://meka.sourceforge.net/#datasets>.
[62] A.N. Srivastava, B. Zane-Ulman, Discovering recurring anomalies in text reports regarding complex space systems, in: Aerospace Conference, IEEE, 2005, pp. 3853–3862, http://dx.doi.org/10.1109/AERO.2005.1559692.
[63] L. Tang, S. Rajan, V.K. Narayanan, Large scale multi-label classification via metalabeler, in: Proc. 18th Int. Conf. on World Wide Web, WWW '09, 2009, pp. 211–220. http://dx.doi.org/10.1145/1526709.1526738.
[64] D.J. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, Chapman & Hall, 2003.
[65] Y. Benjamini, D. Yekutieli, The control of the false discovery rate in multiple testing under dependency, Ann. Stat. (2001) 1165–1188.
[66] O. Luaces, J. Díez, J. Barranquero, J.J. del Coz, A. Bahamonde, Binary relevance efficacy for multilabel classification, Prog. Artif. Intell. 1 (4) (2012) 303–313.

Francisco Charte received his B.Eng. degree in Computer Science from the University of Jaén in 2010 and his M.Sc. and Ph.D. in Computer Science from the University of Granada in 2011 and 2015, respectively. He is currently a researcher at the University of Granada (Spain). His main research interests include machine learning with applications to multi-label classification, high dimensionality and imbalance problems, and association rule mining, as well as CPU/GPU algorithm parallelization techniques.

Antonio J. Rivera received his B.Sc. degree and his Ph.D. in Computer Science from the University of Granada in 1995 and 2003, respectively. He is a lecturer of Computer Architecture and Computer Technology with the Computer Science Department at the University of Jaén (Spain). His research interests include areas such as 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, 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 (http://decsai.ugr.es/~herrera) received his M.Sc. in Mathematics in 1988 and Ph.D. in Mathematics in 1991, both from the University of Granada, Spain. He is currently a Professor in the Department of Computer Science and Artificial Intelligence at the University of Granada. He has been the supervisor of 36 Ph.D. students. He has published more than 300 papers in international journals. He is coauthor of the books: "Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases" (World Scientific, 2001) and "Data Preprocessing in Data Mining" (Springer, 2015). He currently acts as Editor in Chief of the international journals "Information Fusion" (Elsevier) and Progress in Artificial Intelligence (Springer). He acts as editorial board member or associate editor of a dozen of journals, among others: International Journal of Computational Intelligence Systems, IEEE Transactions on Fuzzy Systems, IEEE Transactions on Cybernetics, Information Sciences, Knowledge and Information Systems, Fuzzy Sets and Systems, Applied Intelligence, Knowledge-Based Systems, Memetic Computation, and Swarm and Evolutionary Computation. He received the following honors and awards: ECCAI Fellow 2009, IFSA Fellow 2013, 2010 Spanish National Award on Computer Science ARITMEL to the "Spanish Engineer on Computer Science", International Cajastur "Mamdani" Prize for Soft Computing (Fourth Edition, 2010), IEEE Transactions on Fuzzy System Outstanding 2008 and 2012 Paper Award (bestowed in 2011 and 2015 respectively), 2011 Lotfi A. Zadeh Prize Best paper Award of the International Fuzzy Systems Association, 2013 AEPIA Award to a scientific career in Artificial Intelligence (September 2013). 2014 "Natural de Jaén" Award by the University of Jaén, and 2014 XV Andaluca Research Prize Maimnides By the regional government of Andaluca). He belongs to the list of the Highly Cited Researchers in the areas of Engineering and Computer Sciences: http://highlycited.com/. His current research interests include bibliometrics, computing with words in decision making, information fusion, evolutionary algorithms, evolutionary fuzzy systems, biometrics, data preprocessing, data mining, cloud computing and big data.