Pattern Recognition 122 (2022) 108294

Contents lists available at ScienceDirect

Pattern Recognition

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

Multi-label sampling based on local label imbalance

Bin Liu [a,*], Konstantinos Blekas [b], Grigorios Tsoumakas [a]

[a] School of Informatics, Aristotle University of Thessaloniki, Thessaloniki 54124, Greece
[b] Department of Computer Science and Engineering, University of Ioannina, Ioannina 45110, Greece

ARTICLE INFO

Article history:
Received 30 December 2020
Revised 30 August 2021
Accepted 31 August 2021
Available online 2 September 2021

Keywords:
Multi-label learning
Class imbalance
Oversampling and undersampling
Local label imbalance
Ensemble methods

ABSTRACT

Class imbalance is an inherent characteristic of multi-label data that hinders most multi-label learning methods. One efficient and flexible strategy to deal with this problem is to employ sampling techniques before training a multi-label learning model. Although existing multi-label sampling approaches alleviate the global imbalance of multi-label datasets, it is actually the imbalance level within the local neighbourhood of minority class examples that plays a key role in performance degradation. To address this issue, we propose a novel measure to assess the local label imbalance of multi-label datasets, as well as two multi-label sampling approaches, namely Multi-Label Synthetic Oversampling based on Local label imbalance (MLSOL) and Multi-Label Undersampling based on Local label imbalance (MLUL). By considering all informative labels, MLSOL creates more diverse and better labeled synthetic instances for difficult examples, while MLUL eliminates instances that are harmful to their local region. Experimental results on 13 multi-label datasets demonstrate the effectiveness of the proposed measure and sampling approaches for a variety of evaluation metrics, particularly in the case of an ensemble of classifiers trained on repeated samples of the original data.

© 2021 Elsevier Ltd. All rights reserved.

1. Introduction

In multi-label data, each instance is associated with multiple binary output variables (labels), which allow the expression of much richer semantics compared to binary and multi-class data. The number of labels assigned to each instance is typically much smaller than the total number of output variables. In consequence, the number of instances relevant to each label is much less than the number of irrelevant ones. This gives rise to the problem of *class imbalance*, which has been recently recognized as a key challenge in multi-label learning [1–5].

There are two main types of methods for handling class imbalance in multi-label data: multi-label sampling and algorithm adaptation. The former reduce the imbalance level of multi-label data via adding or removing instances as a pre-processing step [1,2,6,7]. The latter make multi-label learning approaches resilient to class imbalance directly [3–5]. This work focuses on multi-label sampling methods, which can be coupled with any multi-label learning algorithm and are therefore more flexible.

A key challenge for multi-label sampling methods, which has not yet been properly addressed, is how to deal with the co-occurrence of multiple labels, which have varying frequencies, in the same training example. Multi-Label Random OverSampling (MLROS) [1] and Multi-Label Synthetic Minority Oversampling TEchnique (MLSMOTE) [2] relieve the imbalance level for each single minority (less frequent) label via duplicating or generating instances, but this can lead to other labels suffering more severe imbalance. Similarly, Multi-Label Random UnderSampling (MLRUS) [1] reduces the imbalance by focusing on the majority (higher frequent) label separately via removing instances, but this may increase the imbalance level of other labels as well. Multi-Label edited Nearest Neighbor (MLeNN) [6] removes examples associated with majority labels only, yet it is unable to process complex examples having minority and majority labels simultaneously. REsampling Multi-labEl datasets by Decoupling highly ImbAlanced Labels (REMEDIAL) [7] divides a complex example into two easier examples, of which one is associated with minority labels and another with majority labels. However, REMEDIAL brings in additional noise in the dataset, because the pair of new examples have identical features but different labels.

In essence, all the above sampling approaches for multi-label data focus on class imbalance at the *global* scale of the whole dataset. However, previous studies of binary and multi-class data have found that the main reason for the difficulty of a classifier to recognize the minority class is the distribution of class values in the *local* neighbourhood of the minority examples [8,9]. We hypothesize that in a similar vein, the *local distribution of the labels*

* Corresponding author.
E-mail addresses: binliu@csd.auth.gr (B. Liu), kblekas@cs.uoi.gr (K. Blekas), greg@csd.auth.gr (G. Tsoumakas).

https://doi.org/10.1016/j.patcog.2021.108294
0031-3203/© 2021 Elsevier Ltd. All rights reserved.

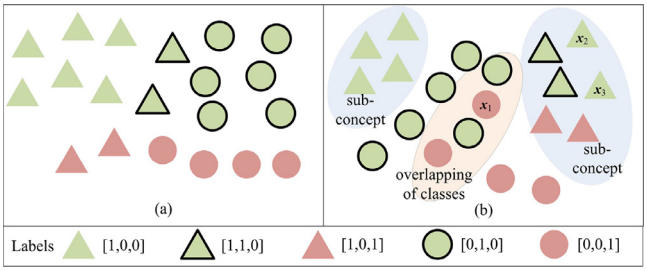


Fig. 1. Two 2-dimensional multi-label datasets (a) and (b) concerning points in a plane characterized by three labels, namely the shape of the points (triangles, circles), the border of the points (solid, none) and the color of the points (green, red). At the bottom we see the five different label combinations that exist in the datasets. The two datasets have the same global imbalance level per label because the number of relevant instances for the three labels is 10, 8 and 6 respectively in both (a) and (b). However, the local label distribution of (b) is more complex than (a) due to the appearance of sub-concepts and the overlapping of classes. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

is more important than the global imbalance level of each label to determine the hardness of a multi-label dataset to be learned. The correctness of this hypothesis will be verified in the experiments section.

Fig. 1 shows an example of two multi-label datasets with the same global level of label imbalance, but different local label distribution. In dataset (b), the triangle points are decomposed into two separated groups, one in the upper left corner and one at the right side, and each group refers to a sub-concept of the triangle class. Furthermore, the co-occurrence of red borderless points and green bordered points in the brown region leads to the overlapping of color classes (red or green) as well as the border style classes (with or without border). Therefore, dataset (b) has a more complex local label distribution and appears more challenging than (a) due to the presence of sub-concepts and class overlap.

This work addresses the class imbalance issue in multi-label data by focusing on the local distribution of the labels. We first present a measure for assessing the level of local imbalance in a multi-label dataset based on the local distribution of the labels. We then propose two twin multi-label sampling methods that take the local imbalance of the labels into account, namely Multi-Label Synthetic Oversampling based on Local label imbalance (MLSOL) and Multi-Label Undersampling based on Local label imbalance (MLUL). MLSOL creates new instances near difficult to learn examples, by using the local label imbalance within the seed instance selection and synthetic instance generation processes. MLUL eliminates difficult to learn examples as evaluated by both local label imbalance and the influence of the example on its reversed k nearest neighbours (RkNN). MLSOL and MLUL take all labels in one instance appropriately into account, by considering the influence of all informative labels. Finally, we embed MLSOL and MLUL, as well as other multi-label sampling methods, within a simple but flexible ensemble framework to further improve their performance and robustness. Experimental results on 13 multi-label datasets illustrate the validity of the proposed measure to assess the difficulty of multi-label dataset and demonstrate the effectiveness of the proposed sampling approaches.

This paper extends our previous work [10] as follows. First, we discuss existing measures to evaluate the imbalance level of multi-label data and propose a new measure based on local label imbalance, which is highly correlated with the difficulty of a multi-label dataset. Furthermore, we propose a multi-label undersampling method, MLUL, which is more efficient than oversampling methods and performs well for imbalance-aware base learners in terms of Macro-average F-measure. At last, we empirically validate the effectiveness of the proposed measure, investigate the reasons why our sampling approaches benefit more from the ensemble framework, and examine the influence of different parameter settings on the proposed methods for easy and difficult datasets, respectively.

The remainder of this paper is organized as follows. Section 2 offers a brief review of previous work. In Section 3, we introduce the new measure to assess the local imbalance level of multi-label data, propose the two sampling approaches based on local label distribution, and present the ensemble framework that can be coupled with multi-label sampling methods. Then, the experimental results along with their discussion are included in Section 4. Finally, the main conclusions of this work are given in Section 5.

2. Related work

2.1. Multi-label learning

Let $\mathcal{X} = \mathbb{R}^d$ be a d-dimensional input feature space, $L = \{l_1, l_2, \ldots, l_q\}$ a label set containing q labels and $\mathcal{Y} = \{0,1\}^q$ a q-dimensional label space. Let $D = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid 1 \leqslant i \leqslant n\}$ be a multi-label training set containing n instances. Each instance $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ consists of a feature vector $\boldsymbol{x}_i \in \mathcal{X}$ and a binary label vector $\boldsymbol{y}_i \in \mathcal{Y}$, where y_{ij} is the j-th element of $\boldsymbol{y}_i$ and $y_{ij} = 1(0)$ denotes that l_j is (not) associated with the i-th instance. The goal of multi-label learning is to learn a mapping function $h : \mathcal{X} \to \{0,1\}^q$ and (or) $f : \mathcal{X} \to \mathbb{R}^q$ that given an unseen instance $\boldsymbol{x} \in \mathcal{X}$, outputs a label vector $\hat{\boldsymbol{y}}$ with the predicted labels and (or) real-valued vector $\hat{\boldsymbol{fy}}$ with the corresponding relevance degrees to $\boldsymbol{x}$ respectively.

The main goal of most multi-label learning methods is to exploit the correlations among labels in order to improve prediction accuracy [11,12]. Multi-label learning methods are divided into three families based on the order of label correlations that they consider, namely first-order, second-order and high-order [13]. Binary Relevance (BR) [14] and Multi-Label k-Neareast Neighbour (MLkNN) [15] are first-order strategies that treat all labels independently and ignore label dependencies. Calibrated Label Ranking (CLR) [16] is a representative second-order approach that considers the correlation among pairs of labels via transforming the multi-label learning problem into several pair-wise label ranking problems. RAndom k-labELsets (RAkEL) [17] and Ensemble of Classifier Chains (ECC) [18] are two methods that exploit high order label correlations by treating label subsets as classes and by embedding labels in chain models, respectively.

2.2. Measuring the imbalance of multi-label data

The imbalance level of a single-label (binary, multi-class) dataset is typically measured by the imbalance ratio, which is computed as the proportion of the number of majority class instances to the number of minority class instances [19].

In multi-label learning, two measures that evaluate the imbalance of a particular label are *IRLbl* [1] and *ImR* [4,5]. Let $n_j^b = |\{(\boldsymbol{x}_i, y_{ij}) \mid y_{ij} = b, 1 \leqslant i \leqslant n\}|$ be the number of instances whose j-th label value is equal to $b \in \{0,1\}$. Let $G_j = \arg\max_{b\in\{0,1\}} n_j^b$ and $g_j = \arg\min_{b\in\{0,1\}} n_j^b$ denote the majority and minority class of l_j respectively. *IRLbl* and *ImR* are then formally defined as:

$$\begin{aligned} IRLbl_j &= n_{max}^1 / n_j^1 \\ ImR_j &= n_j^{G_j} / n_j^{g_j} \end{aligned} \tag{1}$$

where $j \in \{1, 2, \ldots, q\}$ and $n_{\max}^1 = \max_{h=1,\ldots,q} \{n_h^1\}$.

By considering the average and the coefficient of variation of *IRLbl* and *ImR* across all labels, four measures of the imbalance of

multi-label data have been proposed [1,4]:

$$\begin{aligned} MeanIR &= \frac{1}{|q|}\sum_{j=1}^{|q|} IRLbl_j \\ MeanImR &= \frac{1}{|q|}\sum_{j=1}^{q} ImR_j \\ CVIR &= \frac{1}{MeanIR}\sqrt{\sum_{j=1}^{q}\frac{\left(IRLbl_j - MeanIR\right)^2}{q-1}} \\ CVImR &= \frac{1}{MeanImR}\sqrt{\sum_{j=1}^{q}\frac{\left(ImR_j - MeanImR\right)^2}{q-1}} \end{aligned} \tag{2}$$

Labels whose *IRLbl* is larger (less) than *MeanIR* are called *majority* (*minority*) in *label* [1]. The coefficient of variation examines whether all labels suffer from a similar or different level of imbalance. For all the above measures, the higher the value, the more imbalanced the dataset.

When minority class is "1", which is the typical situation for multi-label datasets, *IRLbl* is linearly correlated with *ImR*:

$$ImR_j = n * IRLbl_j / n^1_{max} - 1 \tag{3}$$

Based on Eq. (3), the relation between the *IRLbl* and *ImR* based measures are:

$$\begin{aligned} MeanImR &= n * MeanIR / n^1_{max} - 1 \\ CVImR &= \frac{MeanIR * CVIR}{(MeanIR - n^1_{max}/n)} \end{aligned} \tag{4}$$

However, if the minority class is "0", then the two groups of measures are different because the denominator is n^1_j in $IRLbl_j$ but n^0_j in ImR_j

A recent measure of the imbalance in a multi-label data set that takes into account the occurrence of frequent and rare labels is *SCUMBLE* [7]. It is computed based on the Atkinson index and *IRLbl* as follows:

$$\begin{aligned} SCUMBLE &= \frac{1}{n}\sum_{i=1}^{n} SCUins_i \\ SCUins_i &= 1 - \frac{\sum_{j=1}^{q} y_{ij}\left(\prod_{j=1}^{q}\left(IRLbl_j\right)^{y_{ij}}\right)^{\frac{1}{\sum_{j=1}^{q} y_{ij}}}}{\sum_{j=1}^{q} y_{ij} IRLbl_j} \end{aligned} \tag{5}$$

The range of *SCUMBLE* is between 0 and 1, with higher values indicating more inconsistent frequencies of labels in the examples.

2.3. Handling the imbalance of multi-label data

Methods for dealing with the class imbalance issue in multi-label data can be divided in two groups: multi-label sampling and algorithm adaptation.

2.3.1. Sampling methods

Multi-label sampling methods relieve the global imbalance level of the whole dataset by manipulating the training instances in a pre-processing step. They are independent of the particular multi-label learning algorithm that will be subsequently applied to the dataset.

Multi-label undersampling methods delete instances to reduce the imbalance of the dataset. LP-RUS interprets each labelset (i.e. particular combination of label values) as class identifier and removes instances assigned with the most frequent labelset [20]. Instead of considering the whole labelset, MLRUS alleviates the imbalance of the dataset in the individual label aspect via omitting instances with majority labels randomly [1]. MLeNN employs an Edited Nearest Neighbor (ENN) based strategy to heuristically eliminate instances only assigned with majority labels and have similar labelset with their neighbors [6].

To achieve the balanced label distribution, multi-label oversampling approaches add instances to the dataset. LP-ROS, as a twin method of LP-RUS, replicates instances whose labelset appears the fewest times [20]. Similar to MLRUS, MLROS increases the frequency of minority labels via replicating instances relevant to minority labels to relieve the imbalance in view of individual labels [1]. To reduce the risk of overfitting caused by copying instances, some heuristic synthetic instance generation approaches have been explored. SMOTE [21] is extended to handle the multi-label dataset by employing three generation strategies that define different minority class samples used for creating new instances [22]. MLSMOTE randomly selects an instance containing minority labels, along with its neighbors, to generate synthetic instances. These instances are associated with the labels that appear in more than half of the seed instance and its neighbors [2].

REMEDIAL tackles the co-ocurrence of labels with different imbalance level in one instance, of which the level is assessed by *SCUMBLE*, by decomposing the sophisticated instance into two simpler examples, but may introduce extra confusions into the learning task, i.e. there are several pairs of instances with same features and different labels [7]. REMEDIAL can be used either as standalone sampling method or the prior part of another sampling technique. For example, RHwRSMT combines REMEDIAL with MLSMOTE [23].

2.3.2. Algorithm adaptation methods

Different from sampling methods, algorithm adaptation methods focus on the multi-label learning algorithm handling the class imbalance problem directly. A family of methods deals with the imbalance issue of multi-label learning via transforming the multi-label dataset to several binary/multi-class classification problems. COCOA converts the original multi-label dataset to one binary dataset and several multi-class datasets for each label, and builds imbalance classifiers with the assistance of sampling for each dataset [5]. SOSHF transforms the multi-label learning task to an imbalanced single label classification assignment via cost-sensitive clustering, and the new task is addressed by oblique structured Hellinger decision trees [3].

Another branch of approaches aims to modify current multi-label learning methods to handle the class imbalance problem. ECCRU3 makes ECC resilient to class imbalance by coupling it with undersampling and improving the exploitation of majority examples [4]. Apart from ECCRU3, modified models based on neural networks [24,25], SVM [26], hypernetwork [27] and BR [28–30] have been proposed as well.

Furthermore, other strategies, such as representation learning [31], constrained submodular minimization [32] and balanced pseudo-label [33] have been utilized to address the imbalance obstacle of multi-label learning as well.

3. The proposed measure and methods

We first present a new measure for evaluating the local imbalance of a multi-label dataset. Then, we propose two new multi-label sampling approaches based on the local label distribution. Subsequently, we introduce a simple but flexible framework for ensembling multi-label sampling approaches. Lastly, we analyze the computational complexity of the proposed approaches.

3.1. Local imbalance of multi-label data

As we illustrated in Fig. 1, the local label distribution rather than the global imbalance level is what makes a multi-label dataset challenging to learn. However, all existing measures for assessing the imbalance of multi-label datasets are based on the global imbalance level and ignore local information. Inspired by [8], we propose a measure that gauges the local imbalance of a multi-label dataset via considering the local label distribution of all instances.

At a global level, the minority and majority classes are fixed for each label. However, at a *local* level, the minority and majority classes in different regions may not be the same. Consider the dataset (b) in Fig. 1 as an example. For the first label, the local majority class is the triangle class ("1") in the left blue region, but the circle class ("0") in the middle brown region. Therefore, the local imbalance should not only assess the class imbalance level in the local area but also indicate the corresponding local minority and majority classes. To distinguish the global and local classes and avoid being wordy, we use *minority (majority) class* to denote the global minority (majority) class in the sequel. We define the *local imbalance* of an instance as the proportion of opposite class values in its local neighborhood. Specifically, for each instance $\boldsymbol{x}_i$, we first retrieve its k nearest neighbours $\mathcal{N}_i^k$ according to a distance function, such as the Euclidean distance. Then, for each label l_j we compute the proportion of neighbours having an opposite class with respect to the class of $\boldsymbol{x}_i$, as Eq. (6),

$$C_{ij} = \frac{1}{k} \sum_{\boldsymbol{x}_m \in \mathcal{N}_i^k} [\![y_{mj} \neq y_{ij}]\!] \tag{6}$$

where $[\![\pi]\!]$ is the indicator function that returns 1 if π is true and 0 otherwise. The value of C_{ij} is in [0,1], and $C_{ij} < 0.5$ indicates Y_{ij} is the local majority class in the neighbourhood region of $\boldsymbol{x}_i$ ($\mathcal{N}_i^k$), and local minority class otherwise. The closer to 0.5 C_{ij} is, the more imbalanced l_j is in the local area of $\boldsymbol{x}_i$. Furthermore, C_{ij} measures the difficulty of predicting l_j correctly for $\boldsymbol{x}_i$, regardless of whether Y_{ij} is the local minority or majority class. The local minority class instance is always more difficult than the local majority one, and a value of C_{ij} close to 0 (1) indicates a safe (hostile) neighborhood of similarly (oppositely) labelled examples. A value of $C_{ij} = 1$ can further be viewed as a hint that x_i is an outlier in this neighborhood with respect to l_j. We define a matrix $\boldsymbol{C} \in \mathbb{R}^{n \times q}$ to store the local imbalance of all instances for each label.

We define the local imbalance of the whole dataset, *LImb*, as the average of C_{ij} for the minority class of all instances and labels:

$$LImb = \frac{1}{q} \sum_{j=1}^{q} \frac{\sum_{i=1}^{n} C_{ij} [\![y_{ij} = g_j]\!]}{\sum_{i=1}^{n} [\![y_{ij} = g_j]\!]} \tag{7}$$

The larger the *LImb*, the more difficult the dataset to be learned.

3.2. Oversampling with MLSOL

We propose a new multi-label oversampling approach, Multi-Label Synthetic Oversampling based on Local label imbalance (MLSOL), that generates synthetic instances near those instances that are suffering high local imbalance and are more difficult to be predicted correctly. MLSOL combines the local imbalance of informative labels to pick up difficult seed instances. It assigns appropriate labels to the created synthetic instances so as to improve the frequency of difficult labels, without introducing noise for easy labels.

Firstly, we define some important variables based on the local imbalance matrix $\boldsymbol{C}$. To evaluate the hardness of instance $\boldsymbol{x}_i$, we define its weight as w_i, which characterizes the difficulty in correctly predicting the minority class values of this example by aggregating its C_{ij} for all labels. An initial straightforward way to do this is to simply sum these values for labels where the instance contains the minority class:

$$w_i = \sum_{j=1}^{q} C_{ij} [\![y_{ij} = g_j]\!] \tag{8}$$

However, there are two issues with Eq. (8). The first one is that we have also considered the outliers that are surrounded by opposite class instances. The second issue is that the global level of class imbalance of each label is not taken into account in this aggregation. The fewer the number of minority class samples, the higher the difficulty of correctly classifying the corresponding minority class. In contrast, Eq. (8) treats all labels equally. To address these two issues, we define a new matrix $\boldsymbol{S}$ that takes both global and local imbalance into account and ignores the impact of outliers:

$$S_{ij} = \begin{cases} \frac{C_{ij} [\![y_{ij} = g_j \wedge C_{ij} < 1]\!]}{\sum_{i'=1}^{n} C_{i'j} [\![y_{i'j} = g_j \wedge C_{i'j} < 1]\!]}, & \text{if } [\![y_{i'j} = g_j \wedge C_{i'j} < 1]\!] = 1 \\ -1, & \text{otherwise} \end{cases} \tag{9}$$

Adding the $C_{ij} < 1$ term to the indicator functions leads to omitting the influence of outliers. We consider that l_j is the *informative label* of $\boldsymbol{x}_i$ if $[\![y_{ij} = g_j \wedge C_{ij} < 1]\!] = 1$ ($\boldsymbol{x}_i$ is a non-outlier minority class example for l_j). $S_{ij} \neq -1$ only if l_j is the informative label of $\boldsymbol{x}_i$. Furthermore, the values of informative labels for all instances in $\boldsymbol{S}$ are normalized so that they sum to 1 per label, by dividing with the sum of the values of all non-outlier minority examples of that label. This increases the relative importance of the weights of labels with fewer samples. Finally, we arrive at the following proposed aggregation:

$$w_i = \sum_{j=1}^{q} S_{ij} [\![S_{ij} \neq -1]\!] \tag{10}$$

where $[\![S_{ij} \neq -1]\!]$ is equivalent to $[\![y_{ij} = g_j \wedge C_{ij} < 1]\!]$. The Eq. (10) combines all local imbalance of informative labels for $\boldsymbol{x}_i$. Weights of all instances are stored in $\boldsymbol{w} \in \mathbb{R}^n$.

Furthermore, we introduce the definition of the type of each instance-label pair, which would be utilized to determine the appropriate labels assigned to new instances that we will create. Following [8], we discretize the range $[0, 1]$ of C_{ij} to define four types of minority class instances, namely safe (*SF*), borderline (*BD*), rare (*RR*) and outlier (*OT*), according to their local imbalance :

- SF : $0 \leqslant C_{ij} < 0.3$. Safe instances are located in the region overwhelmed by minority examples.
- BD : $0.3 \leqslant C_{ij} < 0.7$. Borderline instances are located in the decision boundary between minority and majority classes.
- RR : $0.7 \leqslant C_{ij} < 1$. We further consider only those instances whose minority class neighbours are of type *RR* or *OT*. Otherwise there are some *SF* or *BD* examples in the proximity, which suggests that they should be considered as *BD*. Rare instances, accompanied by isolated minority class examples, are located in the majority class area and distant from the decision boundary.
- OT : $C_{ij} = 1$. Outliers are surrounded by majority examples.

For completeness, we add the *MJ* which associates the majority class case. Let $\boldsymbol{T} \in \{SF, BD, RR, OT, MJ\}^{n \times q}$ be the type matrix and T_{ij} be the type of y_{ij}.

The pseudo-code of MLSOL is shown in Algorithm 1 . Firstly, the auxiliary variables defined previously, as the weight vector $\boldsymbol{w}$ and type matrix $\boldsymbol{T}$, are calculated (lines 3–6 in Algorithm 1). Next, the loop describes the procedure of creating new instances (lines 8–13 in Algorithm 1). In each iteration, a synthetic instance is generated as follows: a seed instance $(\boldsymbol{x}_s, \boldsymbol{y}_s)$ is picked at the beginning, with the probability of selection being proportional to its weight (i.e. the more difficult instance has more chance to be selected). Then a reference instance $(\boldsymbol{x}_r, \boldsymbol{y}_r)$ is randomly chosen from the k nearest neighbours of the seed instance with equal probability. Finally, a synthetic example is generated based on the specific seed and reference instances and added into the dataset. The above iterative procedure is terminated when the expected number of new examples are created.

The detailed procedure of how to determine the features and labels of a synthetic instance based on the given seed instance $(\boldsymbol{x}_s, \boldsymbol{y}_s)$ and reference instance $(\boldsymbol{x}_r, \boldsymbol{y}_r)$ along with their types is shown in Algorithm 2 . The feature values of the synthetic instance

Algorithm 1: MLSOL.

input : multi-label data set: D, sampling ratio: p, number of nearest neighbour: k
output: new data set $D\prime$

```
GenNum ← |D| * p ; /* number of instances to generate */
D′ ← D ;
Find the kNN of each instance ;
Calculate C according to Eq.(6) ;
Compute S according to Eq.(9) ;
Compute w according to Eq.(10) ;
T ← InitTypes(C, k) ;          /* Initialize the type of instances */
while GenNum > 0 do
    (x_s, y_s) ←Select a seed instance (x_s, y_s) from D based on w;
    Choose a reference instance (x_r, y_r) from N_s^k;
    (x_c, y_c) ← CreateIns((x_s, y_s), T_s, (x_r, y_r), T_r);
    D′ ← D′ ∪ (x_c, y_c) ;
    GenNum ← GenNum − 1 ;
return D′ ;
```

Algorithm 2: CreateIns.

input : seed instance: $(\boldsymbol{x}_s, \boldsymbol{y}_s)$, types of seed instance: T_s, reference instance: $(\boldsymbol{x}_r, \boldsymbol{y}_r)$, types of reference instance: T_r
output: synthetic instance: $(\boldsymbol{x}_c, \boldsymbol{y}_c)$

```
1  for j ← 1 to d do
2      x_cj ← x_sj + Random(0, 1) * (x_rj − x_sj) ;   /* Random(0,1) return a random value ∈ [0, 1] */
3  d_s ← distance(x_c, x_s), d_r ← distance(x_c, x_r) ;
4  cd ← d_s/(d_s + d_r) ;
5  for j ← 1 to q do
6      if y_sj = y_rj then
7          y_cj ← y_sj ;
8      else
9          if T_sj = MJ then /* ensure y_sj being minority class */
10             s ⟷ r ;  /* swap indices of seed and reference instance */
11             cd ← 1 − cd ;
12         switch T_sj do
13             case SF θ ← 0.5 ; break;
14             case BD θ ← 0.75 ; break;
15             case RR θ ← 1 + 1e − 5 ; break;
16             case OT θ ← 0 − 1e − 5 ; break;
17         if cd ⩽ θ then
18             y_cj ← y_sj ;
19         else
20             y_cj ← y_rj ;
21 return (x_t, y_t) ;
```

$(\boldsymbol{x}_c, \boldsymbol{y}_c)$ are interpolated along the line which connects both input samples (lines 1–2 in Algorithm 2). Once $\boldsymbol{x}_c$ is confirmed, we compute the quantity $cd \in [0, 1]$, which indicates whether the synthetic instance is closer to the seed ($cd < 0.5$) or closer to the reference instance ($cd > 0.5$) (lines 3–4 in Algorithm 2).

With respect to label assignment, we employ a scheme considering the labels and types of the seed and reference instances, as well as the location of the synthetic instance. This scheme is able to create informative instances for locally imbalanced labels without bringing in noises for the rest of the labels. For each label l_j, y_{cj} is set as y_{sj} (lines 6–7 in Algorithm 2) if y_{sj} and y_{rj} belong to the same class. Otherwise, in the case where y_{sj} is the majority class ($T_{sj} = MJ$), the seed and reference instances are exchanged to guarantee that y_{sj} is always the minority class (lines 9–11 in Algorithm 2). Then, the threshold θ for cd is defined according to the type of the label in the seed instance T_{sj} (lines 12–16 in Algorithm 2), which would be responsible for identifying the instance (seed or reference) that will lend its label to the synthetic example. In the case of the first three types (*SF*, *BD*, *RR*), where the minority class (seed) example is surrounded by several majority class instances and may lead to a wrong classification decision, the cut-point of label assignment is closer to the majority class (reference) instance. For *SF*, we set $\theta = 0.5$ so that the label of the nearest (seed or reference) instance is assigned to the synthetic instance. The threshold takes value greater than 1 for *RR*, so as to ensure that the seed's class will remain minority, i.e. $y_{cj} \leftarrow y_{sj}$. Regarding *BD*, we set $\theta = \frac{1+0.5}{2} = 0.75$ that is the midpoint between two previous cases. Finally, for obtaining smoother decision boundary in class regions, the threshold will become less than zero ($\theta < 0$) in *OT* cases, ensuring that the synthetic instance will take the reference class (majority), i.e. $y_{cj} \leftarrow y_{rj}$.

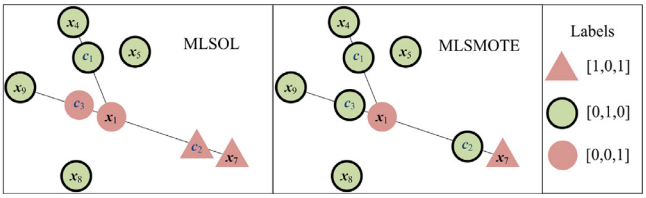


Fig. 2. The subset of dataset (b) concerning $\boldsymbol{x}_1$ and its kNNs as an example of MLSOL excelling MLSMOTE for label assignment for synthetic instances. $\boldsymbol{x}_1$ is the seed instance, $\boldsymbol{x}_4 - \boldsymbol{x}_9$ are candidate reference instances ($kNN(\boldsymbol{x}_1)$), and $\boldsymbol{c}_*$ are possible synthetic examples.

Compared with MLROS and MLSMOTE, MLSOL performs a more comprehensive analysis by emphasizing on more difficult to learn instances and generating more diverse and well-labeled synthetic instances. For dataset (b) in Fig. 1, MLROS would randomly replicate the red data points containing minority label (l_3) with equal chance. Likewise, the probabilities of each red data point to be selected as seed instance by MLSMOTE are equal. On the other hand, MLSOL is more likely to choose $\boldsymbol{x}_1$ as seed instance, because it is surrounded by more opposite class neighbours for l_3. With respect to the synthetic generation process, as shown in Fig. 2, MLSMOTE assigns label vector [0,1,0] to all synthetic instances, as decided by their neighbors. Conversely, MLSOL generates more diverse instances via assigning them labels according to their location. Furthermore, the synthetic instances $\boldsymbol{c}_2$ and $\boldsymbol{c}_3$ generated by MLSMOTE introduce noise, while MLSOL copies the labels of the nearest instance to the new examples. In conclusion, MLSMOTE generates new instances biased to the dominant class in the local area. In contrast, MLSOL is characterized by an efficient exploration and exploitation of the feature and labels space.

3.3. Undersampling with MLUL

We propose a new Multi-Label UndersampLing method (MLUL) that makes minority class examples to be learned more easily via the removal of harmful examples. In MLUL, instead of picking up harmful examples to be deleted directly, we choose a subset of important examples and eliminate the rest.

In traditional undersampling approaches, majority class instances surrounding minority class examples are typically considered candidates for removal. However, this simple strategy be-

comes invalid in the case of multi-label data, where the same training example could be important to some of the labels but damaging for other labels. To deal with this issue, we need to consider two factors to evaluate the importance of an instance. The one is its local imbalance level, and the other one is the influence of the instance on its reverse nearest neighbours, RkNN. The RkNN of $\boldsymbol{x}_i$ is a group of instances where $\boldsymbol{x}_i$ belongs to their neighborhoods [34], i.e.:

$$RkNN(\boldsymbol{x}_i) = \{\boldsymbol{x}_m | \boldsymbol{x}_i \in \mathcal{N}_m^k), 1 \leqslant m \leqslant n\} \tag{11}$$

The local imbalance of an instance could be measured by $\boldsymbol{w}$, as defined in Eq. (10). With respect to the second factor, we follow the principle that an instance is detrimental (beneficial) to an instance belonging to its RkNN if they have opposite (same) class for a label. To this direction, we introduce the influence quantity u_i that measures the impact of instance $\boldsymbol{x}_i$ to its $RkNN(\boldsymbol{x}_i)$:

$$u_i = \sum_{j=1}^{q} \frac{\sum_{\boldsymbol{x_m} \in RkNN(\boldsymbol{x_i})} (-1)^{[\![y_{ij} \neq y_{mj}]\!]} S_{mj} [\![S_{mj} \neq -1]\!]}{|RkNN(\boldsymbol{x_i})|} \tag{12}$$

Influence is calculated by combining the influence degree of informative labels for $\boldsymbol{x}_i$. This is obtained by taking the algebraic sum of local imbalances S_{mj} in all members of group $RkNN(\boldsymbol{x}_i)$, where S_{mj} is added to or subtract from the accumulative influence, depending on whether $\boldsymbol{x}_i$ and $\boldsymbol{x}_m$ have equal or different class for label l_j.The influence can take positive ($u_i > 0$) or negative ($u_i < 0$) values, denoting that the impact of $\boldsymbol{x}_i$ on its $RkNN$ is useful or harmful, respectively. The larger the $|u_i|$, the more beneficial or harmful $\boldsymbol{x}_i$ is considered for its $RkNN$. Specifically, we consider the importance of an instance as:

$$v_i = w_i + u_i - \min\{w_m + u_m\}_{1 \leqslant m \leqslant n} \tag{13}$$

where the subtraction of the third term ensures that all values in $\boldsymbol{v} \in \mathbb{R}^n$ are non-negative.

The pseudo-code of MLUL is shown in Algorithm 3 . Firstly, the

Algorithm 3: MLUL.

input : multi-label data set: D, sampling ratio: p
output: new data set $D\prime$

1 $RetNum \leftarrow |D| * (1 - p)$; `/* number of instances to retain */`
2 $D\prime \leftarrow \emptyset$;
3 Find the kNN and RkNN of each instance ;
4 Calculate $\boldsymbol{C}$ according to Eq.(6) ;
5 Compute $\boldsymbol{S}$ according to Eq.(9) ;
6 Compute $\boldsymbol{w}$ according to Eq.(10) ;
7 Compute $\boldsymbol{u}$ according to Eq.(12);
8 Compute $\boldsymbol{v}$ according to Eq.(13);
9 **while** $RetNum > 0$ **do**
10 Choose an instance $(\boldsymbol{x}, \boldsymbol{y})$ from D based on $\boldsymbol{v}$;
11 $D\prime \leftarrow D\prime \cap (\boldsymbol{x}, \boldsymbol{y})$;
12 $D \leftarrow D \setminus (\boldsymbol{x}, \boldsymbol{y})$;
13 $RetNum \leftarrow RetNum - 1$;
14 **return** $D\prime$;

number of retained instances is computed based on the undersampling ratio p (line 1 in Algorithm 3). Then, several auxiliary variables and the importance of instances $\boldsymbol{v}$ are calculated (lines 2–8 in Algorithm 3). Subsequently, $RetNum$ instances are sampled without replacements, with the probability of selection being proportional to the importance it is associated with (Algorithm 3, lines 9–13). Finally, the sampled instance subset D' is retained and examples excluded from D' are discarded. In addition, we exemplify the advantage of MLUL over MLRUS with dataset (b) in Fig. 1. MLRUS would delete triangle points randomly, while MLUL would remove $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ with more probability than other triangle points because $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ are easier to be learned and hinder their RkNNs, i.e. the green triangle points with borderline for l_2 and red triangle points for l_3.

3.4. Ensemble of multi-label sampling (EMLS)

Ensemble methods constitue an effective strategy to increase the overall accuracy and overcome over-fitting problems, but have not been leveraged in multi-label sampling approaches. To improve the robustness of the proposed multi-label sampling methods, we develop an Ensemble framework for Multi-Label Sampling (EMLS), where any multi-label sampling approach and classifier could be embedded. In EMLS, M homogeneous multi-label learning models with identical hyper-parameters are independently trained, where each model is built upon a sampled dataset generated by a multi-label sampling method with a different random seed. There are many random operations in proposed and existing multi-label sampling methods [1,2], which guarantee the diversity of the training set of each model in the ensemble framework by employing a different random seed. In addition, when our proposed sampling methods are used in the ensemble framework, setting different sampling ratio p or number of neighbours k offers another way to diversify the sampled datasets. Then the bipartition threshold of each label is decided by maximizing F-measure on the training set, as COCOA [5] and ECCRU3 [4] do. Given a test example, the predicted relevance scores are calculated as the average of the relevance scores obtained from the M models, and the labels whose relevance score is larger than the corresponding bipartition threshold are predicted as "1", and "0" otherwise.

3.5. Complexity analysis

The complexity of searching kNN and RkNN of input instances is $O(n^2d + n^2k)$. The complexity of computing auxiliary variables, such as $\boldsymbol{w}$, $\boldsymbol{T}$ and $\boldsymbol{u}$ is $O(nkq)$. The complexity of sampling retained instances in MLUL is $O(pn)$. Therefore, the overall complexity of MLUL is $O(n^2d + n^2k + nkq + pn)$. The complexity of creating synthetic instances is $O(pn(q + d))$. Therefore, the overall complexity of MLSOL is $O(n^2d + n^2k + nkq + pn(q + d))$. kNN searching is the most time-consuming part for both MLUL and MLSOL. MLSOL is more time-consuming than MLUL due to the process of creating synthetic instances.

Let's define $\Theta_s(n, d, q, p)$ the complexity of a multi-label sampling approach, and $\Theta_t(n', d, q)$ and $\Theta_p(d, q)$ the complexity of training and prediction of multi-label learning method respectively where n' is the size of the output sampled dataset. The complexity of EMLS is $O\big(M\big(\Theta_s(n, d, q, p) + \Theta_t(n', d, q) + n'\Theta_p(d, q)\big)\big)$ for training and $O(M\Theta_p(d, q))$ for prediction. Generally, EMLS combined with undersampling methods is much more efficient than with oversampling approaches, because undersampling methods are usually faster and output less number of instances than oversampling methods.

4. Empirical analysis

4.1. Setup

Table 1 shows the 13 benchmark multi-label datasets used in our experimental study along with their global and local imbalance levels. All datasets are available online at Mulan[1]. In textual data sets with more than 1000 features, we applied a simple feature selection approach that retains the top 10% (bibtex, enron,

[1] http://mulan.sourceforge.net/datasets-mlc.html

Table 1
The 13 multi-label datasets used in this study. Columns n, d, q denote the number of instances, features and labels respectively, *LC* the label cardinality. The $k = 5$ for *LImb*.

Dataset	Domain	n	d	q	*LC*	*MeanIR*	*CVIR*	*MeanImR*	*CVImR*	*SCUMBLE*	*LImb*
bibtex	text	7395	183	159	2.402	12.5	0.4051	87.7	0.4097	0.0938	0.8816
cal500	music	502	68	174	26	20.6	1.087	22.3	1.129	0.3372	0.8485
corel5k	image	5000	499	347	3.517	117	1.128	522	1.13	0.3917	0.9725
enron	text	1702	100	52	3.378	57.8	1.482	107	1.496	0.3024	0.844
flags	image	194	19	7	3.392	2.255	0.7648	2.753	0.7108	0.0606	0.5163
genbase	biology	662	1186	24	1.248	20.6	1.269	78.8	1.286	0.0266	0.2112
medical	text	978	144	35	1.245	39.1	1.107	143	1.115	0.0415	0.7438
rcv1subset1	text	6000	472	101	2.88	54.5	2.081	236	2.089	0.2237	0.896
rcv1subset2	text	6000	472	101	2.634	45.5	1.715	191	1.724	0.2092	0.887
scene	image	2407	294	6	1.074	1.254	0.1222	4.662	0.1485	0.0003	0.2633
yahoo-Arts1	text	7484	231	25	1.654	25	2.444	101	2.468	0.0594	0.8523
yahoo-Business1	text	11,214	219	28	1.599	249	2.447	286	2.453	0.1252	0.8603
yeast	biology	2417	103	14	4.237	7.197	1.884	8.954	1.997	0.1044	0.5821

medical) or top 1% (rcv1subset1, rcv1subset2, yahoo-Arts1, yahoo-Business1) of the features ordered by number of non-zero values (i.e. frequency of appearance). Similar pre-processing was applied in [4,5] as well. We remove labels containing only one minority class instance, because when splitting the dataset into training and test sets, there may be only majority class instances of those extremely imbalanced labels in the training set.

Four multi-label sampling methods, namely MLRUS, MLROS [1], MLSMOTE [2] and RHwRSMT [23], are used for comparison, among which the first one is an undersampling method, while the other three are oversampling methods. The ensemble versions of the proposed MLUL and MLSOL methods, denoted as EMLUL and EMLSOL, are compared with the ensemble versions of competing approaches, namely EMLRUS, EMLROS, EMLSMOTE and ERHwRSMT respectively. Furthermore, the base learning algorithm without employing any sampling method, denoted as *Default*, is also used for comparison purposes. In MLUL, MLSOL, MLSMOTE, RHwRSMT, the number of nearest neighbours k is set to 5 and the Euclidean distance is used to measure the distance between the examples. In RHwRSMT, the threshold for decoupling an instance is set to *SCUMBLE*. The sampling ratio p is selected from {0.01,0.5,0.1,0.15,0.2} and {0.1,0.3,0.5,0.7,0.9} for undersampling methods (MLRUS and MLUL) and oversampling methods (MLROS and MLSOL) via 3×2-fold cross-validation on the training set, respectively. The ensemble size M is set to 5 for all ensemble methods. For simplicity, we use a fixed p for sampling methods embedded in EMLS, i.e. p is set as 0.3 for MLSOL and 0.1 for MLUL, MLRUS and MLROS. In addition, six multi-label learning methods are employed as base learning methods, comprising four standard multi-label learning methods (BR [14], MLkNN [15], CLR [16], RAkEL [17]), as well as two state-of-the-art methods addressing the class imbalance problem (COCOA [5] and ECCRU3 [4]).

Three widely used imbalance aware evaluation metrics, namely Macro-averaged *F-measure*, Macro-averaged *AUC-ROC* (area under the receiver operating characteristic curve), and Macro-averaged *AUCPR* (area under the precision-recall curve), are leveraged to measure the performance of methods. For simplicity, we omit "macro-averaged" in further references to these metrics within the rest of this paper. To examine the statistical significance of the differences among the competing methods, the Friedman test, followed by the Wilcoxon signed rank test with Bergman-Hommel's correction at the 5% level is employed [35].

The experiments were conducted on a machine with $4 \times$ 10-core CPUs running at 2.27 GHz. We apply 5×2-fold cross validation with multi-label stratification [36] to each dataset and the average results are reported. The implementation of our approach is publicly available at Mulan's GitHub repository[2]. The default parameters are used for base learners. The numerical value results relating to Table 3–6 are available at the supplementary material.

4.2. Effectiveness of LImb

More difficult datasets usually lead methods to lower prediction accuracy. If a measure is negatively correlated with the prediction accuracy, it is positively correlated with the difficult of the dataset. Therefore, to investigate which imbalance measure can reveal the difficulty of a multi-label dataset, we calculate the Pearson correlation coefficients (ρ) between each global/local imbalance measure and the prediction accuracy of each base approach on the 13 datasets that are listed in Table 1. The results are presented in Table 2, where the ρ coefficients of which the absolute values are larger than the critical value 0.684[3] with $\alpha = 0.01$ and $N = 13$ are highlighted by boldface. At first, we notice that all measures are negatively correlated with prediction accuracy. In most cases, *LImb* is the most significantly correlated measure with ρ round -0.9, followed by *SCUMBLE* whose ρ is nearly -0.7. *SCUMBLE* is the most correlated measure in terms of AUC-ROC of CLR and COCOA, which both belong to the pairwise transformation strategy. Yet, the strength of these correlations are not significant. Overall, *LImb*, which reflects the local label imbalance, rather than the global label imbalance based measures, is the most effective measure to assess the difficulty of multi-label dataset. This empirically supports the assumption that local label distribution is more crucial to decide the hardness of multi-label datasets.

4.3. Results of sampling methods

Table 3 shows the average rank of each method as well as its significant wins/losses versus the rest of the methods in terms of the three evaluation metrics and the six base multi-label methods.

Initially, we focus on the average ranks and overall statistical test results across all base learners. MLSOL is the best method in terms of all metrics, achieving the top average rank and most significant wins without suffering any significant loss. MLROS and MLSMOTE are usually inferior to MLSOL, which verifies the effectiveness of utilizing local imbalance. These three methods outperform the default one. The two undersampling methods come next. The proposed MLUL outperforms MLRUS because MLUL retains more important instances via considering local imbalance. Compared with default approaches, MLUL is improved in F and

[2] https://github.com/tsoumakas/mulan/tree/master/mulan
[3] https://www.statisticssolutions.com/table-of-critical-values-pearson-correlation/

Table 2
The Pearson correlation coefficients (ρ) between global and local imbalance measures and prediction accuracy on 13 datasets. Parentheses denote the rank of the corresponding ρ in each row. Given the critical value 0.684 with $\alpha = 0.01$ and $N = 13$, the significant ρ not in the range of [-0.684,0.684] are boldfaced.

Metric	Base	*MeanIR*	*CVIR*	*MeanImR*	*CVImR*	*SCUMBLE*	*LImb*
F-measure	BR	−0.401(4)	−0.348(5)	−0.52(3)	−0.347(6)	**−0.692(2)**	**−0.959(1)**
	MLkNN	−0.428(6)	−0.493(4)	−0.552(3)	−0.487(5)	−0.648(2)	**−0.964(1)**
	CLR	−0.387(4)	−0.368(5)	−0.513(3)	−0.367(6)	**−0.696(2)**	**−0.962(1)**
	RAkEL	−0.412(4)	−0.385(5)	−0.537(3)	−0.383(6)	**−0.686(2)**	**−0.971(1)**
	COCOA	−0.433(4)	−0.377(5.5)	−0.56(3)	−0.377(5.5)	**−0.717(2)**	**−0.959(1)**
	ECCRU3	−0.428(6)	−0.493(4)	−0.552(3)	−0.487(5)	−0.648(2)	**−0.964(1)**
AUC-ROC	BR	−0.314(6)	−0.334(3.5)	−0.318(5)	−0.334(3.5)	−0.638(2)	**−0.852(1)**
	MLkNN	−0.318(6)	−0.412(3)	−0.359(5)	−0.404(4)	−0.674(2)	**−0.828(1)**
	CLR	−0.14(5)	−0.169(4)	−0.062(6)	−0.174(3)	−0.532(1)	−0.353(2)
	RAkEL	−0.337(6)	−0.348(4.5)	−0.41(3)	−0.348(4.5)	**−0.688(2)**	**−0.918(1)**
	COCOA	−0.181(5)	−0.247(4)	−0.154(6)	−0.255(3)	−0.645(1)	−0.487(2)
	ECCRU3	−0.318(6)	−0.412(3)	−0.359(5)	−0.404(4)	−0.674(2)	**−0.828(1)**
AUCPR	BR	−0.417(4)	−0.358(5)	−0.528(3)	−0.357(6)	−0.637(2)	**−0.945(1)**
	MLkNN	−0.449(4)	−0.435(5)	−0.572(3)	−0.427(6)	−0.645(2)	**−0.992(1)**
	CLR	−0.462(4)	−0.378(5)	−0.566(3)	−0.374(6)	**−0.693(2)**	**−0.967(1)**
	RAkEL	−0.424(4)	−0.406(5)	−0.56(3)	−0.403(6)	−0.666(2)	**−0.977(1)**
	COCOA	−0.461(4)	−0.413(5)	−0.587(3)	−0.41(6)	**−0.692(2)**	**−0.976(1)**
	ECCRU3	−0.449(4)	−0.435(5)	−0.572(3)	−0.427(6)	−0.645(2)	**−0.992(1)**

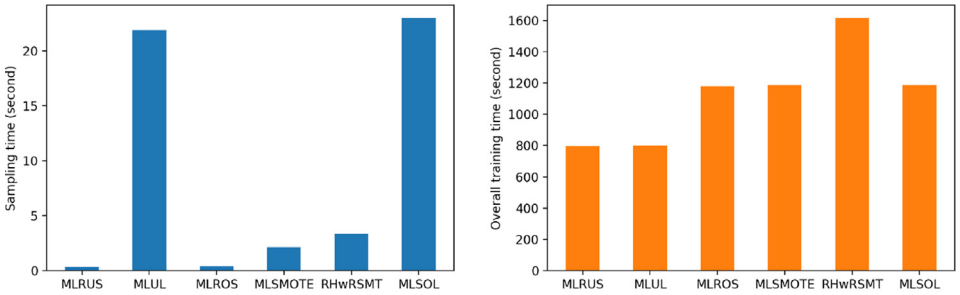
Table 3
Average rank of the compared sampling methods using 6 base learners in terms of three evaluation metrics. The parenthesis (n_1/n_2) indicates the corresponding method is significantly superior to n_1 methods and inferior to n_2 methods based on the Wilcoxon signed rank test with Bergman-Hommel's correction at the 5% level. The best methods are highlighted by boldface.

Metric	Base	Default	MLRUS	MLUL	MLROS	MLSMOTE	RHwRSMT	MLSOL
F-measure	BR	4.62(1/3)	4.46(1/2)	4.62(1/2)	3(2/0)	2.35(4/0)	7(0/6)	**1.96(4/0)**
	MLkNN	4.73(1/3)	5(1/3)	4.5(1/3)	2.92(4/2)	2.69(5/1)	7(0/6)	**1.15(6/0)**
	CLR	4.12(1/0)	4.12(1/0)	4.19(1/0)	3.69(1/0)	3.31(1/0)	7(0/6)	**1.58(1/0)**
	RAkEL	4.96(1/3)	4.81(1/3)	3.92(1/2)	2.92(4/0)	2.69(3/1)	7(0/6)	**1.69(5/0)**
	COCOA	3.46(1/0)	3.12(1/0)	**2.77(1/0)**	4.38(1/0)	4.46(1/0)	6.58(0/5)	3.23(0/0)
	ECCRU3	2.96(1/0)	2.81(1/0)	**2.58(1/0)**	4.58(1/0)	4.69(1/0)	6.38(0/5)	4(0/0)
	Ave(Total)	4.14(6/9)	4.05(6/8)	3.76(6/7)	3.58(13/2)	3.37(15/2)	6.83(0/34)	**2.27(16/0)**
AUC-ROC	BR	5(0/1)	4.23(0/1)	4.23(1/1)	3.58(1/0)	3.85(1/1)	5.96(0/4)	**1.15(5/0)**
	MLkNN	3.62(0/0)	5.12(0/0)	3.31(0/0)	4.31(0/0)	**2.85(1/0)**	5.54(0/2)	3.27(1/0)
	CLR	3.69(0/0)	4.46(0/0)	4.23(0/0)	**3(0/0)**	4.35(0/0)	5.15(0/0)	3.12(0/0)
	RAkEL	4.35(1/1)	4.27(1/1)	4.27(1/1)	3.35(1/0)	3.23(1/1)	7(0/6)	**1.54(5/0)**
	COCOA	4.58(0/2)	5.58(0/2)	4.58(0/1)	**2.5(3/0)**	3.77(0/0)	4.38(0/1)	2.62(3/0)
	ECCRU3	4.69(0/0)	4.69(0/0)	3.92(0/0)	**2.88(1/0)**	3.38(0/0)	5.54(0/1)	**2.88(0/0)**
	Ave(Total)	4.32(1/4)	4.73(1/4)	4.09(2/3)	3.27(6/0)	3.57(3/2)	5.6(0/14)	**2.43(14/0)**
AUCPR	BR	4.35(1/0)	3.58(1/0)	4.96(1/0)	3.69(1/0)	3.12(1/0)	6.23(0/6)	**2.08(1/0)**
	MLkNN	3.88(0/0)	4.96(0/0)	3.96(0/0)	4.54(0/0)	3.23(1/0)	5.15(0/2)	**2.27(1/0)**
	CLR	3.12(1/0)	4.23(0/0)	3.88(0/0)	**2.92(1/0)**	4.46(0/0)	5.77(0/2)	3.62(0/0)
	RAkEL	4.12(1/0)	4.23(1/0)	3.96(1/0)	3.19(1/0)	3.58(1/0)	7(0/6)	**1.92(1/0)**
	COCOA	4.92(0/0)	5(0/0)	4.15(0/0)	2.77(0/0)	4.04(0/0)	4.81(0/0)	**2.31(0/0)**
	ECCRU3	4.38(0/0)	4.5(0/0)	4.5(0/0)	3.38(0/0)	3.46(0/0)	4.73(0/0)	**3.04(0/0)**
	Ave(Total)	4.13(3/0)	4.42(2/0)	4.24(2/0)	3.42(3/0)	3.65(3/0)	5.62(0/16)	**2.54(3/0)**

AUC but performs even worse in AUCPR. Similarly, MLRUS deteriorates the accuracy of default methods in AUC and AUPR. The reduced accuracy of undersampling methods is mainly attributed to the inevitable information loss caused by removing instances associated with multiple labels [1]. Finally, RHwRSMT is the worst method, mainly because of the additional confusion yielded by REMEDIAL, i.e. there are several pairs of instances with the same features and disparate labels.

However, there are some exceptions for some base learners and metrics that should be noticed. Firstly, when COCOA and ECCRU3 are used as the base learner, MLUL is the best method, and undersampling approaches improve the two base learners, while oversampling methods are usually worse than the default ones in F-measure. The two imbalance-aware base learners utilize the bipartition threshold selection strategy that maximizes F-measure for each label on the training set. Nevertheless, the bipartition thresholds determined upon a sampled dataset would deviate from the original data distribution. The oversampling methods that add minority class instances have more impact than the undersampling ones that remove majority class instances on the accuracy of the bipartition threshold selection, as F-measure is the harmonic mean of precision and recall with respect to the minority class. Secondly, for MLkNN which is a neighbourhood-based base learner, the proposed MLSOL and MLUL that depend on the local label imbalance, as well as MLSMOTE which determines the labels of the synthetic instance by its neighbours, surpass other methods that totally ignore the local label information, except for MLROS, which is better than MLUL in F-measure. Lastly, MLROS is most effective on COCOA in AUC as well as CLR in AUC and AUCPR. Both COCOA and CLR divide the whole multi-label learning problem into several sub-problems involving pairs of labels, and each sub-problem is characterized by the imbalance level between the two labels, i.e. the ratio of frequencies of the two labels. By replicating instances associated with less frequent labels, MLROS diminishes the difference among the frequencies of labels and implicitly reliefs the imbalance between the labels.

Furthermore, the default approach does not suffer any significant loss in AUCPR, indicating that using single multi-label sampling methods are not very effective in AUCPR metric, which is considered as the most appropriate measure in the context of class imbalance. In addition, none of the sampling methods is able to significantly improve the performance of ECCRU3 and CLR in any



(a) Sampling Time (b) Overall Traingin Time with RAkEL

Fig. 3. Sampling and overall training time of sampling methods on Corel5k dataset.

measure, as ECCRU3 is an imbalance aware method and CLR is influenced by imbalance between labels.

In addition, Fig. 3(a) shows the sampling time of comprising methods on Corel5k dataset. MLRUS and MLROS simply replicating and deleting instances are the fastest approaches, followed by MLSMOTE and RHwRSMT that search kNN from subsets associated with minority labels. The proposed MLUL and MLSOL have a higher computational cost due to more additional time spending on the kNN retrieval from the whole dataset, as we analysed in Section 3.5. Nevertheless, when it comes to the overall training time including sampling data and training the base learner (RAkEL), as shown in Fig. 3(b), the difference between sampling methods is negligible. This is because the training time of the base learner, which depends on the size of the sampled dataset, dominates the overall computational cost. The undersampling approaches reducing the dataset are more efficient than the oversampling ones conducting enlargement. RHwRSMT is the most time-consuming method, becasue it creates a larger sampled dataset than other oversampling competitors via the instance decomposition.

Overall, MLUL is most effective on COCOA and ECCRU3 that directly tackle the class imbalance issue in F-measure, while MLSOL performs well in most of the other cases.

4.4. Results of ensemble methods

In this part, we examine the effectiveness of EMLS. We compare EMLS coupled with the six sampling approaches, as well as the Default approach. Average ranks and statistical test results are shown in Table 4.

Firstly, we observe that embedding a sampling method in the EMLS approach can significantly improve the performance of using the sampling method alone for all base learners and in all evaluation metrics. This verifies the known effectiveness of resampling approaches in reducing the error, in particular via reducing the variance component of the expected error [37].

According to the results, EMLUL is better in terms of the F-measure, while EMLSOL is superior in terms of the other two criteria, AUC-ROC and AUCPR. Both EMLSOL and EMLUL are not significantly inferior to the other five comparing methods. The single significant loss of EMLSOL is caused by EMLUL and the 5 significant losses of EMLUL are attributed to EMLSOL. For BR, MLkNN and RAkEL, EMLSOL achieves the best performance in terms of all metrics, with the exception that EMLSMOTE is the best one for MLkNN in terms of F-measure. For CLR depending on label pairwise transformation and the two imbalance aware base learners (COCOA and ECCRU3), EMLUL outperforms EMLSOL in terms of F-measure and is comparable with EMLSOL, i.e. EMLUL and EMLSOL have the same numbers of significant wins and the difference between their average ranks are tiny, in terms of AUC and AUCPR. In addition, as we analyzed in Section 3.5, EMLUL utilizes fewer instances to train base learners and therefore is more computationally efficient than EMLSOL. Besides, the ratios of training time consumed by EMLSOL to EMLUL on Corel5k data set shown in Fig. 4 empirically verify that the computational cost of EMLSOL is 2.5 times of EMLUL for MLkNN which is quadratic to the number of instances and 1.5 times for other base learners. Overall, concerning F-measure, EMLSOL is more beneficial to BR, MLkNN and RAkEL, and EMLUL is more suitable for CLR, COCOA and ECCRU3. In terms of AUC and AUCPR, although EMLSOL is usually the best approach for all base learners, EMLUL is able to achieve comparable performance for CLR, COCOA and ECCRU3 with lower computational cost.

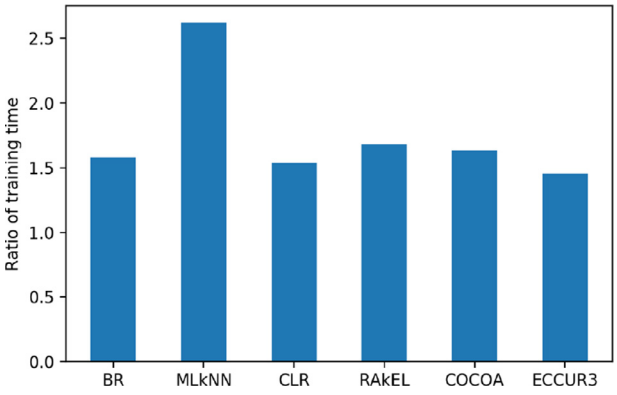


Fig. 4. The ratios of training time used by EMLSOL to EMLUL on Corel5k data set for six bases learners.

Other competitors are inferior to the proposed two methods. EMLROS and EMLRUS are usually the third and fourth methods, followed by EMLSMOTE. The top five ensemble approaches are better than Default, with two exceptions: EMLSMOTE is worse than the Default approach for COCOA and ECCRU3 in terms of F-measure. ERHwRSMT is the worst ensemble method, even worse than Default in some cases (i.e. for RAkEL, COCOA and ECCRU3 in F-measure). Furthermore, none of the ensemble methods is significantly better than COCOA and ECCRU3 in terms of F-measure, which is also due to the employment of the bipartition threshold optimization strategy on F-measure.

Diversity is an important factor of the effectiveness of ensemble approaches [38]. We therefore analyze the diversity of ensembling the different multi-label sampling methods with EMLS to shed light on the reasons for the superior performance achieved by our proposed methods. To assess the ensemble diversity, a widely used pairwise diversity measure, called *disagreement* [38], is employed. Although the original definition of disagreement aims to evaluate diversity among binary classifiers, it could be easily applied to the multi-label learning scenario via a simple transformation. Given a test set including n_t instances and two prediction matrices $Y^i \in \{0,1\}^{n_t \times q}$ and $Y^j \in \{0,1\}^{n_t \times q}$ obtained from two multi-

Table 4
Average rank of the compared ensemble of resampling methods using 6 base learners in terms of three evaluation metrics. The parenthesis (n_1/n_2) indicates the corresponding method is significantly superior to n_1 methods and inferior to n_2 methods based on the Wilcoxon signed rank test with Bergman-Hommel's correction at the 5% level. The "*" following the average rank denotes that the ensemble methods significantly outperform their corresponding signal sampling approaches. The best methods are highlighted by boldface.

Metric	Base	Default	EMLRUS	EMLUL	EMLROS	EMLSMOTE	ERHwRSMT	EMLSOL
F-measure	BR	6.38(0/5)	4.23*(2/1)	3.46*(2/1)	3.69*(1/0)	3.08*(2/1)	5.62*(0/4)	**1.54*(5/0)**
	MLkNN	7(0/6)	3.38*(1/0)	3*(1/0)	2.85***(2/0)**	**2.69*(2/0)**	5.23*(1/2)	3.85*(1/0)
	CLR	6.15(0/6)	3.15*(1/0)	**2.81*(3/0)**	3.69*(1/0)	4.5*(1/1)	4.31*(1/1)	3.38*(1/0)
	RAkEL	6.15(0/5)	3.85***(2/0)**	3.38***(2/0)**	3.08***(2/0)**	3.46***(2/0)**	6.31*(0/5)	**1.77*(2/0)**
	COCOA	5.15(0/0)	2.92*(2/0)	**2.23*(3/0)**	3.62*(2/0)	5.23*(0/4)	5.38*(0/3)	3.46*(1/1)
	ECCRU3	5(0/0)	2.69*(1/0)	**2.31*(2/0)**	3.38*(1/0)	5.62*(0/2)	5.46*(0/3)	3.54*(1/0)
	Ave(Total)	5.97(0/22)	3.37(9/1)	**2.87(13/1)**	3.39(9/0)	4.1(7/8)	5.39(2/18)	2.92(11/1)
AUC-ROC	BR	6.23(0/5)	3.77*(2/2)	3*(3/1)	3.69*(2/1)	4.15*(2/1)	6.08*(0/5)	**1.08*(6/0)**
	MLkNN	5.62(0/3)	3.31*(2/0)	3.15***(3/0)**	5.38*(0/3)	3.23*(1/0)	5.54*(0/3)	**1.77*(3/0)**
	CLR	5.92(0/3)	4*(0/2)	**2.35*(4/0)**	3.12*(2/0)	4.65*(1/2)	5.5*(0/4)	2.46***(4/0)**
	RAkEL	5.77(1/5)	3.81*(2/2)	2.5*(4/1)	3.81*(2/1)	3.96*(2/2)	6.88*(0/6)	**1.27*(6/0)**
	COCOA	6.46(0/5)	3.69*(2/1)	2.38***(4/0)**	2.38*(3/0)	5.23*(1/3)	5.54*(0/4)	**2.31***(3/0)
	ECCRU3	6.31(0/4)	3.42*(3/0)	2.69***(3/0)**	2.38*(3/0)	5.08*(1/4)	6.15*(0/5)	**1.96*(3/0)**
	Ave(Total)	6.05(1/25)	3.67(11/7)	2.68(21/2)	3.46(12/5)	4.38(8/12)	5.95(0/27)	**1.81*(25/0)**
AUCPR	BR	6.12(0/5)	3.73*(2/2)	2.96*(4/1)	3.69*(2/1)	3.96*(2/2)	6.38*(0/5)	**1.15*(6/0)**
	MLkNN	6.15(0/5)	2.85*(2/0)	3.27*(2/0)	4.88*(1/1)	3.88*(2/1)	5.58*(0/4)	**1.38*(4/0)**
	CLR	6.19(0/4)	3.38*(2/0)	2.54***(3/0)**	2.62*(2/0)	5.12*(0/2)	6*(0/4)	**2.15*(3/0)**
	RAkEL	6.04(0/5)	3.5*(2/1)	2.5*(3/1)	3.69*(2/0)	4.42*(2/2)	6.54*(0/5)	**1.31*(5/0)**
	COCOA	6.54(0/5)	3.54*(3/2)	2.19***(4/0)**	2.73*(3/0)	5.31*(1/4)	5.69*(0/4)	**2*(4/0)**
	ECCRU3	6.38(0/4)	3.27***(3/0)**	2.38***(3/0)**	2.65***(3/0)**	5.15*(1/4)	6.08*(0/5)	**2.08*(3/0)**
	Ave(Total)	6.24(0/28)	3.38(14/5)	2.64(19/2)	3.38(13/2)	4.64(8/15)	6.05(0/27)	**1.68(25/0)**

Table 5
Average rank of disagreement (*disa*) of multi-label sampling ensemble approaches on 13 datasets.

Base	EMLRUS	EMLUL	EMLROS	EMLSMOTE	ERHwRSMT	EMLSOL
BR	3.08	2.77	4.15	4.23	5.62	**1.15**
MLkNN	2.96	2.81	4.23	4.42	5.35	**1.23**
CLR	2.88	2.65	4.31	4.31	5.69	**1.15**
RAkEL	2.81	2.58	4.38	4.38	5.77	**1.08**
COCOA	**1.77**	2	3.77	5.12	5.58	2.77
ECCRU3	2	**1.46**	3.77	5.42	5.19	3.15
Ave	2.24	2.67	3.54	4.46	4.77	**2.21**

Table 6
The result of Wilcoxon signed rank test with Bergman-Hommel's correction at the 5% level for EMLS with different strategies to generate diverse sampled datasets in terms of AUCPR.

Base	A=MLSOL			A=MLUL		
	EA	E_pA	E_kA	EA	E_pA	E_kA
BR	0/1	**2/0**	0/1	0/0	0/0	0/0
MLkNN	0/2	**1/0**	**1/0**	0/0	0/0	0/0
CLR	0/2	**1/0**	**1/0**	**1/0**	0/1	0/0
RAkEL	0/2	**1/0**	**1/0**	0/0	0/0	0/0
COCOA	0/2	**1/0**	**1/0**	0/0	0/1	**1/0**
ECCRU3	0/0	0/0	0/0	0/0	0/0	0/0

label learning methods f^i and f^j respectively, the disagreement of the two learners is calculated as the proportion of different predictions in the two matrices:

$$disa_{ij} = \frac{\sum_{m=1}^{n_t}\sum_{l=1}^{q}[\![Y^i_{ml} \neq Y^j_{ml}]\!]}{n_t q} \tag{14}$$

The disagreement of an ensemble model is computed as the average disagreement of every pair of multi-label base learners embedded in the ensemble framework:

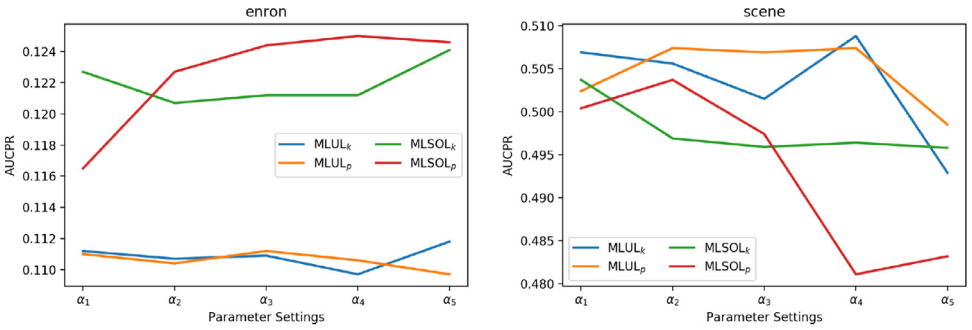
$$disa = \frac{\sum_{i=1}^{M}\sum_{j=1_{i\neq j}}^{M} disa_{ij}}{M(M-1)} \tag{15}$$

The *disa* measure takes values in [0,1]. The larger the *disa* mesaure, the more diverse the ensemble model.

Table 5 lists the average rank of disagreement of ensemble models with six sampling approaches on the 13 datasets. EMLSOL has the lowest *disa* value for standard base methods, while it is the third best method for imbalance aware base methods (COCOA and ECCRU3), ahead of the other three oversampling methods. Although MLSOL is worse than MLROS and MLSMOTE in some cases, EMLSOL always outperforms EMLROS and EMLSMOTE, because MLSOL generates more diverse synthetic instances. EMLUL and EMLRUS are more diverse than EMLROS and EMLSMOTE, which results in that EMLUL (EMLRUS) is better than (comparable with) EMLROS and EMLSOMTE, despite the inferior performance of the single undersampling method in most cases. The outputs of each model in ERHwRSMT are most consistent, which also results in its worse performance. Overall, our proposed sampling methods embedded within EMLS excel other competing approaches mainly because MLUL and MLSOL can benefit more from the ensemble framework via outputting more diverse sampled datasets.

4.5. Parameter analysis

An additional study has been made in order to investigate the influence of parameters, namely the number of neighbours k and sampling ratio p, on the proposed methods, MLSOL and MLUL. In the experiments, we use different settings for one parameter, while keeping others unchanged at the setting illustrated in Section 4.1. We vary $k = \{5, 6, 7, 8, 9\}$ for both sampling approaches, $p = \{0.01, 0.05, 0.1, 0.15, 0.2\}$ for MLUL, and $p = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ for MLSOL. The AUCPR results on the enron



(a) enron (b) scene

Fig. 5. Results of MLSOL and MLUL with various parameter settings on enron and scene datasets in terms of AUCPR.

dataset with *LImb*=0.844 and the scene dataset with *LImb*=0.2633 using BR as base learner are shown in Fig. 5.

In the enron dataset, with the increase of p, the performance of MLSOL improves, which indicates that generating more synthetic instances contributes to lessening the hardness of the difficult dataset. In the scene dataset, MLSOL achieves the best result when $p = .3$, but it fails when excess instances are added. MLUL can slightly improve the performance by removing small parts of harmful instances for both datasets. However, when too many instances are removed, it's inevitable to lose informative instances, which results in performance decrease. With respect to k, although the optimal settings of MLSOL and MLUL on the two datasets are different, $k = 5$ seems a relative good setting for both approaches on all datasets.

Then, we examine two alternative ways, setting various p and k for MLUL and MLSOL, to increase the diversity of EMLS. Specifically, E$_k$MLSOL (E$_k$MLUL) contains five MLSOL (MLUL) models and the k of each one is set 5, 6, 7, 8, 9, respectively. Analogously, E$_p$MLSOL (E$_p$MLUL) denotes the embedded MLSOLs (MLULs) with $p = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ ($p = \{0.01, 0.05, 0.1, 0.15, 0.2\}$). The statistical test results are shown in Table 6. The E$_p$MLSOL and E$_k$MLSOL manage to enhance the diversity of the EMLS, therefore they significantly outperform the EMLSOL. Besides, the higher p promotes the performance of MLSOL for difficult datasets, which also contributes to the improvement of E$_p$MLSOL. In contrast, there are a few significant differences between the three strategies for MLUL. MLUL is relatively insensitive to the parameters changing, hence using various p and k hardly increase the diversity of MLUL.

5. Conclusion

In this study, we presented a local label distribution based measure to assess the local imbalance level of multi-label dataset. Based on the local imbalance concept, we proposed two multi-label sampling methods considering all informative labels, in order to make the multi-label dataset easier to be learned. MLSOL selects difficult seed instances and generates more diverse and well-labeled synthetic instances. MLUL removes harmful instances that are easier and hinder their RkNNs. Furthermore, we employed MLSOL and MLUL within a simple ensemble framework, which exploits the random aspects of our approaches during the instance selection and synthetic instance generation.

The analysis of the relation between measures and performances of six multi-label learning methods on 13 benchmark multi-label datasets shows the effectiveness of the local label distribution based measure. In addition, experimental results demonstrate the advantage of the proposed methods on the compared multi-label sampling approaches, especially within the ensemble framework due to their ability to produce diverse sampled datasets. Specifically, MLUL is the best option for COCOA and ECCRU3 that have addressed the imbalance issue by themselves in F-measure, while MLSOL is usually outstanding in other situations. When it comes to the ensemble of sampling method, EMLUL is more suitable for CLR depending on the pairwise label transformation and imbalance aware base learners, whereas EMLSOL is most effective on other standard multi-label learning approaches, namely BR, MLkNN and RAkEL.

In the future, we would like to extend our approaches for multi-view data whose property is described in diverse aspects and weakly labelled data associated with incomplete labels.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Bin Liu is supported from the China Scholarship Council (CSC) under the Grant CSC No.201708500095.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2021.108294

References

[1] F. Charte, A.J. Rivera, M.J. del Jesus, F. Herrera, Addressing imbalance in multi-label classification: measures and random resampling algorithms, Neurocomputing 163 (2015) 3–16.
[2] F. Charte, A.J. Rivera, M.J. Del Jesus, F. Herrera, MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation, Knowl Based Syst 89 (2015) 385–397.
[3] Z.A. Daniels, D.N. Metaxas, Addressing Imbalance in Multi-Label Classification Using Structured Hellinger Forests, in: AAAI, 2017, pp. 1826–1832.
[4] B. Liu, G. Tsoumakas, Making Classifier Chains Resilient to Class Imbalance, in: ACML, 2018, p. 280295. Beijing
[5] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, Towards class-imbalance aware multi-label learning, in: IJCAI, 2015, pp. 4041–4047.
[6] F. Charte, A.J. Rivera, M.J. Del Jesus, F. Herrera, MLeNN: A first approach to heuristic multilabel undersampling, in: IDEAL, Springer International Publishing, 2014, pp. 1–9.
[7] F. Charte, A.J. Rivera, M.J. del Jesus, F. Herrera, Dealing with difficult minority labels in imbalanced mutilabel data sets, Neurocomputing 326–327 (2019) 39–53.
[8] K. Napierala, J. Stefanowski, Types of minority class examples and their influence on learning classifiers from imbalanced data, J Intell Inf Syst 46 (3) (2016) 563–597.
[9] J.A. Sáez, B. Krawczyk, M. Woźniak, Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets, Pattern Recognit 57 (2016) 164–178.
[10] B. Liu, G. Tsoumakas, Synthetic Oversampling of Multi-Label Data based on Local Label Distribution, ECML-PKDD, 2019. Würzburg
[11] S.Y. Li, Y. Jiang, N.V. Chawla, Z.H. Zhou, Multi-Label learning from crowds, IEEE Trans Knowl Data Eng 31 (7) (2019) 1369–1382.

[12] R. Wang, S. Kwong, X. Wang, Y. Jia, Active k-labelsets ensemble for multi-label classification, Pattern Recognit 109 (2021).
[13] M.L. Zhang, Z.H. Zhou, A review on multi-label learning algorithms, IEEE Trans Knowl Data Eng 26 (8) (2014) 1819–1837.
[14] M.R. Boutell, J. Luo, X. Shen, C.M. Brown, Learning multi-label scene classification, Pattern Recognit 37 (9) (2004) 1757–1771.
[15] M.-L. Zhang, Z.-H. Zhou, ML-KNN: A lazy learning approach to multi-label learning, Pattern Recognit 40 (7) (2007) 2038–2048.
[16] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, K. Brinker, Multilabel classification via calibrated label ranking, Mach Learn 73 (2) (2008) 133–153.
[17] G. Tsoumakas, I. Katakis, I. Vlahavas, Random k-labelsets for multilabel classification, IEEE Trans Knowl Data Eng 23 (7) (2011) 1079–1089.
[18] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Mach Learn 85 (3) (2011) 333–359.
[19] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans Knowl Data Eng 21 (9) (2009) 1263–1284.
[20] F. Charte, A. Rivera, M.J. del Jesus, F. Herrera, A first approach to deal with imbalance in multi-label datasets, in: HAIS, 2013, pp. 150–160.
[21] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, Journal of Artificial Intelligence Research 16 (2002) 321–357.
[22] 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: CIARP, 2013, pp. 334–342.
[23] F. Charte, A.J. Rivera, M.J. del Jesus, F. Herrera, REMEDIAL-HwR: tackling multilabel imbalance through label decoupling and data resampling hybridization, Neurocomputing 326–327 (2019) 110–122.
[24] G. Tepvorachai, C. Papachristou, Multi-label imbalanced data enrichment process in neural net classifier training, in: IJCNN, 2008, pp. 1301–1307.
[25] C. Li, G. Shi, Improvement of learning algorithm for the multi-instance multi--label RBF neural networks trained with imbalanced samples, Journal of Information Science and Engineering 29 (4) (2013) 765–776.
[26] P. Cao, X. Liu, D. Zhao, O. Zaiane, Cost Sensitive Ranking Support Vector Machine for Multi-label Data Learning, in: HIS, 2017, pp. 244–255. Cham
[27] K.W. Sun, C.H. Lee, Addressing class-imbalance in multi-label learning via two-stage multi-label hypernetwork, Neurocomputing 266 (2017) 375–389.
[28] K. Chen, B.-L. Lu, J.T. Kwok, Efficient Classification of Multi-label and Imbalanced Data using Min-Max Modular Classifiers, in: IJCNN, 2006, pp. 1770–1775.
[29] S. Dendamrongvit, M. Kubat, Undersampling approach for imbalanced training sets and induction from multi-label text-categorization domains, in: PAKDD, 2009, pp. 40–52.
[30] M.A. Tahir, J. Kittler, F. Yan, Inverse random under sampling for class imbalance problem and its application to multi-label classification, Pattern Recognit 45 (10) (2012) 3738–3750.
[31] L. Li, H. Wang, Towards Label Imbalance in Multi-label Classification with Many Labels, arXiv preprint arXiv:1604.01304 (2016).
[32] B. Wu, S. Lyu, B. Ghanem, Constrained Submodular Minimization for Missing Labels and Class Imbalance in Multi-label Learning, in: AAAI, 2016, pp. 2229–2236.
[33] W. Zeng, X. Chen, H. Cheng, Pseudo labels for imbalanced multi-label learning, in: DSAA, 2014, pp. 25–31.
[34] Y. Tao, D. Papadias, X. Lian, Reverse kNN search in arbitrary dimensionality, in: VLDB, 2004, pp. 744–755.
[35] A. Benavoli, G. Corani, F. Mangili, Should we really use post-Hoc tests based on mean-ranks? Journal of Machine Learning Research 17 (2016) 1–10.
[36] K. Sechidis, G. Tsoumakas, I. Vlahavas, On the Stratification of Multi-label Data, in: ECML-PKDD, 2011, pp. 145–158.
[37] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, 2016.
[38] Z.-H. Zhou, Ensemble methods: foundations and algorithms, Chapman and Hall/CRC, 2012, doi:10.1201/b12207.



Bin Liu received an M.S. degree in computer science from Chongqing University of Posts and Telecommunications, China in 2016. He is currently pursuing a Ph.D. degree in computer science from Aristotle University of Thessaloniki, Greece. His research interests include multi-label learning and class imbalance.



Konstantinos Blekas received the Diploma degree in Electrical Engineering in 1993 and the Ph.D. degree in Electrical and Computer Engineering in 1997, both from the National Technical University of Athens. He is currently associate professor at the Department of Computer Science & Engineering, University of Ioannina, Greece. He has co-authored more than 80 refereed journal and conference articles. His research interests include machine learning, pattern recognition, intelligent agents and reinforcement learning with applications to robotics and autonomous systems, computer vision, and medicine.



Grigorios Tsoumakas is an Associate Professor of Machine Learning and Knowledge Discovery at the School of Informatics of the Aristotle University of Thessaloniki (AUTH) in Greece. He received a degree in Computer Science from AUTH in 1999, an MSc in Artificial Intelligence from the University of Edinburgh, United Kingdom, in 2000 and a PhD in Computer Science from AUTH in 2005. His research expertise focuses on supervised learning techniques (ensemble methods, multi-target prediction) and natural language processing (semantic indexing, keyphrase extraction, summarization). He has published more than 100 research papers and according to Google Scholar he has more than 14,000 citations and an h-index of 43. Dr. Tsoumakas is a senior member of the ACM and an action editor of the Data Mining and Knowledge Discovery journal. His honors include receiving the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD) 10-Year Test of Time Award in 2017. He is an advocate of applied research that matters and has worked as a machine learning engineer, researcher and consultant in several national, international and private sector funded R&D projects. In February 2019 he co-founded Medoid AI, a spin-off company of the Aristotle University of Thessaloniki developing AI products and custom solutions based on machine learning technology.