Information Sciences 578 (2021) 344–363

Contents lists available at ScienceDirect

Information Sciences

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

Toward hierarchical classification of imbalanced data using random resampling algorithms

Rodolfo M. Pereira [a,b,*], Yandre M.G. Costa [c], Carlos N. Silla Jr. [a]

[a] Pontifícia Universidade Católica do Paraná (PUCPR), Curitiba, PR, Brazil
[b] Instituto Federal do Paraná (IFPR), Pinhais, PR, Brazil
[c] Universidade Estadual de Maringá (UEM), Maringá, PR, Brazil

ARTICLE INFO

Article history:
Received 26 February 2020
Received in revised form 3 July 2021
Accepted 7 July 2021
Available online 13 July 2021

Keywords:
Hierarchical datasets
Global approaches
Class imbalance
Random resampling algorithms

ABSTRACT

Although the class imbalance issue affects hierarchical datasets, the literature has few studies that deal with it, especially when it comes to methods to pre-process the training dataset as a whole. In this study, we present novel resampling algorithms to deal with imbalance in hierarchical datasets. To be able to process the imbalanced data, we first propose methods to retrieve the set of majority and minority label paths. The resampling algorithms were designed with respect to the depth of the label path prediction and may be used for hierarchical classification problems with single- and multiple-path labels. We propose two oversampling and two undersampling algorithms: HROS-FD/PD – Random Oversampling for Full/Partial Depth Hierarchical problems, and HRUS-PD/FD – Random Undersampling for Full/Partial Depth Hierarchical problems. While the resampling algorithms for full-depth problems deal with the data by simply processing the set of majority/minority paths, the partial depth approaches process the data in a leaf-node order way, recalculating the set of majority/minority paths at each step until reaching the root label. The experimental evaluation with 23 hierarchical datasets across different domains and characteristics, supported by statistical analysis, showed that the proposed resampling algorithms significantly improve the classification performance.

1. Introduction

Researchers in the field of machine learning and data mining often face challenges addressing problems involving classification tasks. Usually, the purpose of this type of task is to build a computational model that maps the sample features in relation to their labels to predict the labels of new and unseen examples.

Many researchers have focused on flat classification problems. By flat classification, we refer to binary, multi-class, or multi-label classification problems, that is, problems in which the samples are associated with two or more labels, but these labels do not have hierarchical relations with each other. However, many important real-world classification problems are naturally cast as hierarchical [1], in which the predicted classes are organized into a class hierarchy, such as protein function prediction [2], text categorization [3], sound signal classification [4], and medical image classification [5].

Many researchers face imbalanced class distribution issues, especially when dealing with real-world datasets, which makes the task of learning from imbalanced data an important challenge in the field [6]. The class imbalance problem may affect the predictions because the classifiers are usually focused on the minimization of the global error rate; thus, when

* Corresponding author.

https://doi.org/10.1016/j.ins.2021.07.033
0020-0255/© 2021 Elsevier Inc. All rights reserved.

dealing with imbalanced data, the algorithms tend to benefit the most frequent classes (known as majority classes). However, depending on the scenario, for example, in credit card fraud detection [7] and medical image classification [8], the main interest of the task is exactly in correctly labeling the rare patterns, that is, the less frequent classes (known as minority classes).

Several methods have been proposed in the literature to address imbalanced data distribution problems in the flat classification context [9–12]. Approaches based on data resampling, which can be further sub-categorized into oversampling and undersampling, are the most common and widely used solutions. While the first balances the dataset by creating or duplicating samples for the minority classes, the second is aimed at the removal of existing samples from the majority classes.

Dealing with the imbalance issue in hierarchical classification datasets is a challenging task, because there exist a wide range of hierarchical problems, which in turn have different types of properties, such as the type of label taxonomy, the depth of the prediction, and the number of paths associated with each sample [1]. Even though imbalance is a well-known issue in machine learning and data mining communities, there are few studies on this issue specifically for hierarchical classification problems. Moreover, to the best of our knowledge, there are no studies in the literature proposing resampling algorithms to deal with class imbalance in hierarchical datasets handling the label hierarchy as a whole.

Given the previously described context, the main contributions of this study are threefold:

- An approach to find sets of majority/minority label paths in a hierarchical dataset;
- Resampling algorithms for hierarchical classification problems with partial and full depth label prediction;
- A diversified set of hierarchical classification datasets allowing testbeds for the research community; and
- A formula to calculate the theoretical complexity score of hierarchical datasets.

Applying the proposed algorithms in twenty-three well-known datasets we will perform an experimental analysis with statistical rigor to investigate and discuss the following research questions: (i) The influence of the proposed resampling algorithms in the classification results; (ii) The most/least effective resampling algorithm; (iii) The influence of the resize rate in the classification results; (iv) The definition of a default value for the resize rate; (v) The influence of the dataset mean imbalance in the resampling and classification results; (vi) Which technique generate the best results (oversampling versus undersampling).

Applying the proposed algorithms to twenty-three well-known datasets, we performed an experimental analysis with statistical rigor to investigate and discuss the following research questions: (i) What is the influence of the proposed resampling algorithms on the classification results? (ii) Which is the most/least effective resampling algorithm? (iii) What is the influence of the resize rate on the classification results? (iv) How can we define the default value for the resize rate? (v) What influence does the dataset mean imbalance have on the resampling and classification results? And (vi) which technique generates the best results (over-sampling versus undersampling)?

The remainder of this paper is organized as follows: Section 2 presents the theoretical background, such as basic concepts of hierarchical classification problems and related works concerning how to handle imbalanced data. In Section 3, we describe the proposed approach to retrieve the majority/minority set of label paths and the novel resampling algorithms to deal with the imbalance issue in hierarchical datasets. In Section 4, we present the experimental protocol used in this study and the classification results before and after employing the proposed resampling methods. In Section 5, we present the discussion regarding the results supported by statistical analysis, and finally, in Section 6, we present concluding remarks and future work directions.

2. Theoretical background

This section presents the main concepts concerning hierarchical classification, such as problem definitions, classification approaches, evaluation metrics, and related works in the field of imbalanced data distribution and classification.

2.1. What is hierarchical classification?

Hierarchical classification can be seen as a particular type of classification problem, in which the output of the learning algorithm is defined over a specific class taxonomy. In Wu et al. (2005) [13] the taxonomy is considered a structured tree hierarchy defined over a partially order set $(C, \prec)$, where C is a finite set that enumerates all class concepts in the application domain, and the relation $\prec$ represents a "IS-A" relationship.

Hierarchical multi-label classification is a variant in which instances may belong to multiple classes at the same time, and these classes are organized in a hierarchy.

Fig. 1 shows an example of ten classes organized in a hierarchical structure. The root node is colored in gray, and all child nodes are yellow. We can observe that it is possible to have 16 different label paths though this label tree (ignoring the root node): A, B, C, D, E, F, G, H, I, A/B, A/B/C, D/E, D/E/G, D/E/H, D/F, and D/F/I.

According to Silla Jr. and Freitas (2011) [1], a hierarchical classification problem can be described as a three-tuple (Υ, Ψ, Φ), where.

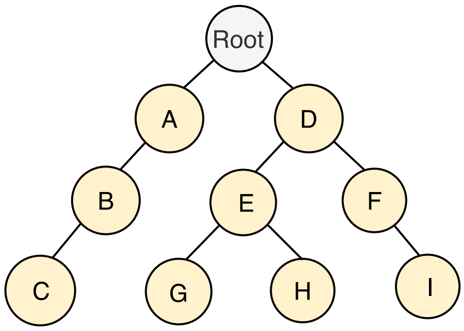


Fig. 1. An example of a hierarchical class structure.

- Υ specifies the type of representation of the labels hierarchy: direct acyclic graph (DAG) or tree;
- Ψ indicates whether a sample may be associated with a single or multiple paths in the class hierarchy; and
- Φ describes the label depth of the samples in the hierarchy: full depth labeling or partial depth labeling.

It is important to state that, in this study, we are only concerned with hierarchical classification problems in which the sample labels are organized in a tree taxonomy.

2.2. Classification approaches

According to Silla Jr. and Freitas (2011) [1], there are different approaches to tackling hierarchical classification problems, which can be categorized according to the classification process. The simplest way is to completely ignore the hierarchy between the classes, predicting each label path as a unique single label by itself.

In the local classifier approaches, the whole hierarchy is partially considered, because the information is used from a local perspective. Local classifier approaches can be further sub-categorized into three groups, focusing on how the local information is employed to build the classification model: Local Classifier per Parent Node (LCPN); Local Classifier per Level (LCL); and Local Classifier per Node (LCN) [1].

Finally, according to Freitas and Carvalho (2007) [14], in the Global Classifiers (GC) approach, a single classification model is built from the training data, taking into account the class hierarchy as a whole during the classification process.

Although the problem of hierarchical classification can be tackled by using local classifier approaches, creating a single global model for all label nodes has the benefit that the total size of the classification model is considerably smaller. In addition, dependencies between different labels with respect to class membership (e.g., a sample belonging to class B automatically belongs to class A/B) can be taken into account in a natural way [1]. Thus, in this study, we used a global approach to perform the hierarchical classification task.

2.3. Evaluation metrics

According to Cerri et al. (2015) [15], the metrics generally used in flat classification problems are inadequate for measuring the performance of hierarchical classifiers. Additionally, not considering the label hierarchy and facts that a sample may simultaneously belong to more than one label, conventional metrics ignore that the complexity of classification usually increases with the depth of the labels to be predicted.

Kiritchenko et al. (2004) [16] proposed two evaluation measures based on conventional precision and recall to consider hierarchical relationships between classes. Hierarchical precision and hierarchical recall were formally defined later in Kiritchenko *et al.* (2005) [17].

The hierarchical precision/recall considers that a sample belongs not only to its predicted labels but also to all its ancestor labels in the hierarchical taxonomy. Given a sample (x_i, L'_i), where x_i belongs to the space X of samples, L'_i is the set of predicted labels for x_i, and L_i is the set of true labels of x_i, the sets L_i and L'_i can be extended to contain their corresponding ancestor labels as $\widehat{L_i} = \bigcup_{l_k \in L_i} Ancestors(l_k)$ and $\widehat{L'_i} = \bigcup_{l_m \in L'_i} Ancestors(l_m)$, where $Ancestors(c_k)$ denotes the set of ancestors of class l_k [15].

Eqs. (1) and (2) show the hierarchical precision (hP) and recall (hR) metrics, respectively. These measures count the number of labels correctly predicted, in conjunction with the number of ancestor labels correctly predicted [15].

Eqs. (1) and (2) show the hierarchical precision (hP) and recall (hR) metrics, respectively. These measures count the number of correctly predicted labels, in conjunction with the number of correctly predicted ancestor labels [15].

$$hP = \frac{\sum_i |\widehat{L_i} \bigcap \widehat{L'_i}|}{\sum_i |\widehat{L'_i}|} \tag{1}$$

$$hR = \frac{\sum_i |\widehat{L_i} \bigcap \widehat{L'_i}|}{\sum_i |\widehat{L_i}|} \tag{2}$$

Fig. 2 shows examples of how to calculate the hR and hP measures considering the label hierarchy presented in Fig. 1. While Fig. 2(a) shows the real labels of a sample with solid black circles, in Fig. 2(b), we present three predictions for this sample, which are represented by solid purple circles and a red arrow showing the deepest predicted label.

Considering the definitions of hierarchical precision and recall, we may define the area under the precision-recall curve (AUPRC) for hierarchical classification algorithms. The AUPRC was originally defined in the two-class scenario as the area below the curve plotted with the precision (y-axis) and recall values (x-axis) considering the different classifier decisions thresholds. In a hierarchical problem, the hPs and hRs can be calculated individually by label path and then joined using an average calculation.

Fig. 3 presents a graphical example of the precision-recall curves for the label paths from the label tree presented in Fig. 1.

According to Davis and Goadrich (2006) [18], the precision-recall curve is more informative when there is a high-class imbalance in the data. Thus, in this study, we used the AUPRC to measure the classification results of the experiments described in Section 4.

2.4. Imbalance data classification

A large number of approaches have been proposed to deal with imbalance in binary classification problems, which can be mainly sub-categorized into three groups:

- Data-level solutions: The objective of these techniques is to re-balance the class distribution by resampling the dataset to diminish the effect of the class imbalance, that is, pre-process the dataset before the training phase (e.g.: [9,19]).
- Algorithmic level solutions: These solutions attempt to adapt the classification algorithms to strengthen the learning toward the minority class. Therefore, they can be defined as internal approaches that create new algorithms or modify existing ones to consider the class imbalance problem (e.g.: [20]).
- Cost-sensitive solutions: These solutions incorporate approaches at the data level, at the algorithmic level, or at both levels jointly, considering higher misclassification costs for the examples of the positive class with respect to the negative class, and therefore, try to minimize higher cost errors (e.g.: [21]).

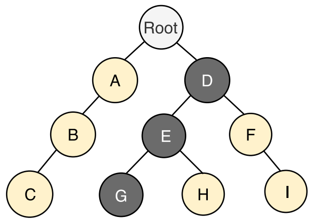
As the resampling method is the most well-known and commonly used technique to solve the imbalance issue, it is the focus of the present study.

The resampling methods can be subdivided into two categories: oversampling and undersampling, which are used to adjust the class distribution of a dataset, that is, the ratio between the different classes in the dataset. In undersampling, some instances from the majority class are removed, whereas to balance the classes' distribution, in oversampling, some instances from the minority class are duplicated or synthetically created to balance the classes distribution.

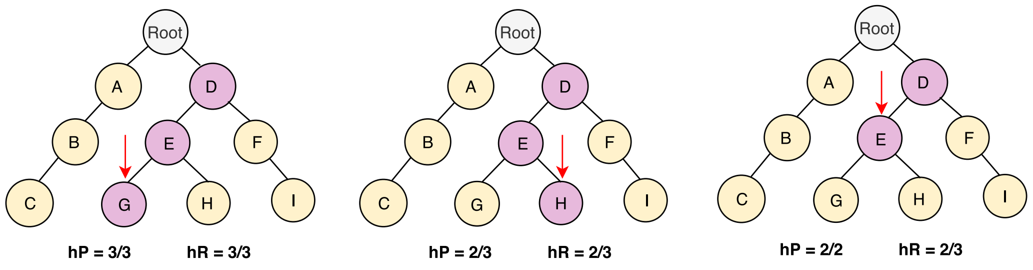
Although the resampling solutions were first defined and implemented for datasets with binary class distribution, they may also be applied to multi-class imbalance problems. According to Wang and Yao (2012) [22], to apply these solutions to multi-class problems, most attention in the literature has been devoted to class decomposition, that is, the conversion of a multiclass problem into a set of binary class sub-problems. Two common decomposing schemas are the one-versus-one (OVO) and one-versus-all (OVA). While the OVO technique, first used by Hastie and Tibshirani (1998) [23], proposed to train a classifier for each possible pair of classes, ignoring the examples that do not belong to the related classes, the OVA approach, introduced by Rifkin and Klantan (2004) [24], builds a single classifier for each class of the problem, considering the examples of the current class as positives and the remaining instances as negatives.

In this direction, a large number of resampling methods have been proposed to deal with imbalance data in flat classification problems [9,19,25–27]. The Synthetic Oversampling Technique (SMOTE) is one of the best-known resampling algorithms in the literature [26,28]. SMOTE was initially proposed for binary and multiclass classification scenarios. Charte et al. (2015) proposed MLSMOTE, which is an extension of SMOTE to handle multi-label classification problems [25]. SMOTE/MLSMOTE generates new synthetic samples for the minority class(es) by interpolating the nearest samples. After a given sample is selected from the minority class, one of its k-nearest neighbors is randomly selected, and a new sample is generated in the corresponding directions. Variations in SMOTE were also proposed for specific scenarios such as multi-instance (Informative-Bag-SMOTE [29]), or to consider the classes borderlines to avoid the generation of samples close to these areas, which is the case of Borderline-SMOTE, proposed by Han *et al.* (2005) [30].

Among various resampling methods, the random resampling algorithms are usually the first ones used by researchers to identify the impacts of data imbalance in datasets. Table 1 presents a brief review of the random resampling algorithms proposed for flat classification datasets, describing their main idea, problem type, and strategy (oversampling or undersam-



(a) Real Classes.



(b) Predicted Classes.

Fig. 2. Examples of hierarchical precision and recall measures.

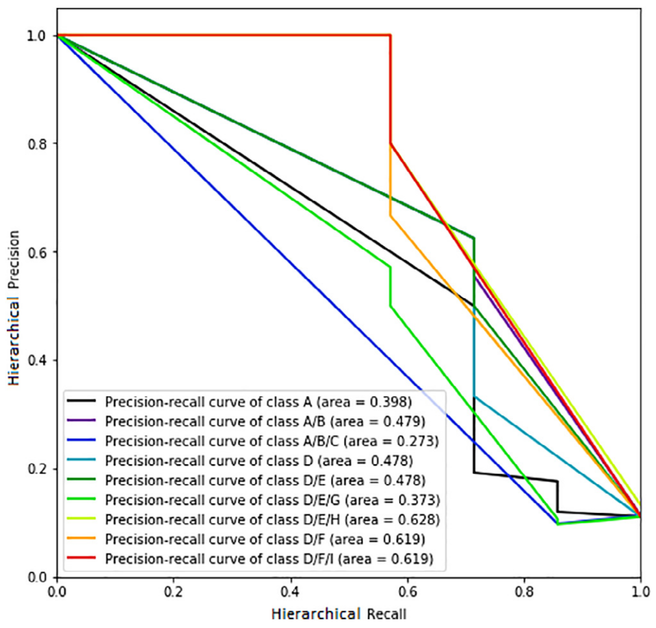


Fig. 3. An example of Precision-Recall Curves for the label paths from Fig. 1.

Table 1
Summary of Random Resampling Algorithms.

Algorithm	Problem Type	Main Idea	Strategy	Reference
ROS	Binary and Multi-class	Randomly duplicates samples from the minority class(es).	Oversampling	[27]
RUS	Binary and Multi-class	Randomly removes samples from the minority class(es).	Undersampling	[27]
LP-ROS	Multi-Label	Duplicates samples associated with the least frequent labelsets.	Oversampling	[31]
LP-RUS	Multi-Label	Removes samples from the most frequent labelsets.	Undersampling	[31]
MLROS	Multi-Label	Duplicates samples associated with the minority labels.	Oversampling	[32]
MLRUS	Multi-Label	Removes samples linked to the majority labels.	Undersampling	[32]

pling). We may observe that the random resampling methods are well established and defined in all classification problem scenarios, except for the hierarchical.

To the best of our knowledge, the issue of dealing with imbalanced data in hierarchical classification datasets was only directly addressed in our previous works (Pereira et al. (2018) [33] and Pereira *et al.* (2021) [34]). Furthermore, in the same study, we proposed an approach aimed at the conversion of the dataset labels to a strictly multi-label format, apply well-known multi-label resampling techniques, and then convert the dataset back to its hierarchical taxonomy. Although this approach can be considered a solution, it does not tackle the imbalance problem from a global perspective, as we attempt to address in the present work.

2.5. Measuring the imbalance in hierarchical datasets

The measurement of imbalance in datasets, known as imbalance ratio (IR), is usually obtained by computing the ratio between the number of samples in the majority classes and those associated with the minority classes. A high IR results in a highly imbalanced dataset.

Pereira et al. (2018) [33] proposed Formula (3), which defines the imbalance ratio for a certain label path p as $IRLbP(p)$. In this context, p is the set of all possible label paths that have at least one occurrence, P_i is the *i-th* label path, and the dataset is represented as D.

$$IRLbP(p) = \frac{\max_{p\prime \in P}\left(\sum_{i=1}^{|D|} h(p\prime, P_i)\right)}{\sum_{i=1}^{|D|} h(p, P_i)} h(p, P_i) = \begin{cases} 1, & p \in P_i \\ 0, & p \notin P_i \end{cases} \tag{3}$$

In Formula (3), the value is 1 for the most frequent label path and a greater value for the others. The higher the *IRLbP*, the larger the imbalance level for the label path.

Furthermore, in Pereira et al. (2018) [33], Formula (4) is also defined to retrieve the mean imbalance of a hierarchical dataset (*HMeanIR*) based on the average of the imbalance per label path, which is presented by *IRLbP*.

$$HMeanIR = \frac{1}{|P|} \sum_{p=P_1}^{P_{|P|}} IRLbP(p) \tag{4}$$

3. The proposed resampling algorithms

Although we are only concerned with hierarchical classification problems in which the labels are organized in a tree-based taxonomy, to design the novel resampling algorithms, we have considered two other variants described in Silla Jr. and Freitas (2011) [1]: number of paths (defined in the literature as Ψ) and depth of the paths (defined in the literature as Φ). We propose two groups of resampling algorithms that consider the depth of the labels in the hierarchical classification problem. Each group was composed of an oversampling and an undersampling algorithm.

3.1. Finding the majority and minority classes

When resampling a dataset, we first define the target samples of the resampling process. When we apply an oversampling algorithm, the samples belonging to the minority classes must be increased, and when applying an undersampling algorithm, the instances from the majority classes are decreased.

In binary and multiclass datasets, the majority/minority classes can be identified by considering the most/less frequent labels among the samples. Moreover, for the identification of majority and minority labels in multi-label datasets, Charte et al. (2013) [31] defined and suggested the use of the IRLbl and MeanIR imbalance measures. Their idea was to consider the labels with an imbalance ratio (IRLbl) below the average imbalance ratio (MeanIR) as belonging to the set of majority classes (named as majority bag); otherwise, the classes belong to the minority bag.

Thus, before proposing a new resampling algorithm for hierarchical datasets, we first have to establish a mechanism to identify the majority and minority classes, considering class hierarchy. In hierarchical problems, the classes are represented by label paths in the tree taxonomy instead of individual labels, and we used the imbalance ratio per label path (IRLbP) and hierarchical mean imbalance ratio (HMeanIR), as proposed by Pereira et al. (2018) [33]. Our idea here is to find the majority and minority paths in the dataset based on their imbalance ratio, which were calculated using Formulas (3) and (4).

Algorithm 1 shows the pseudocode for the imbalanced ratio calculations. Furthermore, the pseudocode of the proposed methods to retrieve the set of majority and minority label paths in hierarchical datasets are presented in Algorithms 2 and 3, respectively.

Algorithm 1. Pseudocode for Imbalance Ratios Calculations

```
Inputs: D: The hierarchical dataset
Output: IRLbP: The imbalance ratio per label paths in D
HMeanIR: The average imbalance ratio in D
 1: labelPaths ← label paths from dataset D
 2: for each path in labelPaths do
 3:     countDict[path] ← number of samples in D labelled with path
 4: end for
 5: maxCount ← max number of labelPaths in countDict
 6: IRLbP ← empty dictionary
 7: for each path in labelPaths
 8:     pathCount ← countDict[path]
 9:     IRLbP[path] ← maxCount/pathCount
10: end for
11: HMeanIR ← $\left(\sum_{i=1}^{|labelPaths|} IRLbP[path_i]\right)/|labelPaths|$
12: return IRLbP, HMeanIR
```

Algorithm 1 receives as input the hierarchical dataset (*D*) and returns two outputs: IRLbP and HMeanIR. The looping from lines 1–3 counts the number of samples belonging to each label path from dataset *D*. It is important to state that we are considering all samples that are labeled with the given *path*, for example, if the *path* is "A/B", samples labeled with "A/B/C" and "A/B/D" are taken into account. In lines 5–10, the IRLbP is calculated, and then in line 11, the average imbalance ratio is obtained.

Algorithms 2 and 3 receive dataset *D* as input and output the set of majority (majPaths) or minority (minPaths) paths. First, in line 1, both algorithms use Algorithm 1 to obtain the imbalance ratios and then, in lines 2–7, the algorithms looped over the label paths, filtering those whose IRLbP are below HMeanIR (majority paths) or above HMeanIR (minority paths).

Algorithm 2. Pseudocode for Retrieving Majority Paths

```
Inputs: D: The hierarchical dataset
Output: majPaths: The label paths from the set of majority paths
1: IRLbP, HMeanIR ← Calculate Imbalance Ratios
2: majPaths ← empty list
3: for each path in labelPaths do
4:     if IRLblP[path] < HMeanIR then
5:         append path into majPaths
6:     end if
7: end for
8: return majPaths
```

Algorithm 3. Pseudocode for Retrieving Minority Paths

```
Inputs: D: The hierarchical dataset
Output: minPaths: The label paths from the set of minority paths
1: IRLbP, HMeanIR ← Calculate Imbalance Ratios
2: minPaths ← empty list
3: for each path in labelPaths do
4:     if IRLblP[path] > HMeanIR then
5:         append path into minPaths
6:     end if
7: end for
8: return minPaths
```

Given the target samples, which can be obtained with Algorithms 1–3, we are able to oversample or undersample the hierarchical dataset. In the following subsections, we present the details of the novel resampling algorithms. To contemplate the different kinds of hierarchical problems, we proposed two different oversampling/undersampling algorithms, considering the depth of label paths in the dataset (based on the Φ definition - full or partial depth), one oversampling/undersampling for each type.

3.2. Resampling full depth hierarchical classification problems

In this type of problem, the instances may be associated with one or more label paths, but always with full depth in the label tree. Considering this, we propose random oversampling/undersampling for full depth hierarchical classification problems (HROS-FD/HRUS-FD). The pseudocodes of these methods are presented in Algorithms 4 and 5, respectively.

Both algorithms are very similar and receive the dataset to oversample/undersample (represented as D) and the percentage of samples to increase or decrease (S) and output the resampled dataset (D'). The main idea of the algorithms is to obtain the set of majority/minority label paths and randomly remove/create samples from/for these paths until the number of removed/created samples reaches the percentage determined by the S parameter. As the procedure followed by HROS-FD and HRUS-FD is analogous, in the following, we will only describe HROS-FD in detail.

Algorithm 4. Pseudocode for HROS-FD

```
Inputs: $D$: The hierarchical dataset,
$S$: Percentage of samples to increase (default = 10%)
Output: $D'$: An oversampled dataset
 1: samplesToCreate $\leftarrow |D| \times S$
 2: minPaths $\leftarrow$ getMinorityPaths($D$)
 3: maxIncrease $\leftarrow$ samplesToCreate/ |minPaths|
 4: meanSize $\leftarrow$ calculate the average number of samples per labelPaths
 5: for labelPath in minPaths do
 6:     numSamples $\leftarrow$ samplesWithLabelPath($D$, labelPath)
 7:     increased $\leftarrow 0$
 8:     while increased < maxIncrease and numSamples < meanSize do
 9:         $D' \leftarrow$ randomly duplicate sample labeled with labelPath
10:         numSamples $\leftarrow$ numSamples + 1
11:         increased $\leftarrow$ increased + 1
12:     end while
13: end for
14: return $D'$
```

Algorithm 5. Pseudocode for HRUS-FD

```
Inputs: $D$: The hierarchical dataset,
$S$: Percentage of samples to increase (default = 10%)
Output: $D'$: An undersampled dataset
 1: samplesToRemove $\leftarrow |D| \times S$
 2: maxPaths $\leftarrow$ getMajorityPaths($D$)
 3: maxDecrease $\leftarrow$ samplesToRemove/ |maxPaths|
 4: meanSize $\leftarrow$ calculate the average number of samples per labelPaths
 5: for labelPath in minPaths do
 6:     numSamples $\leftarrow$ samplesWithLabelPath($D$, labelPath)
 7:     decrease $\leftarrow 0$
 8:     while decrease < maxDecrease and numSamples > meanSize do
 9:         $D' \leftarrow$ randomly remove sample labeled with labelPath
10:         numSamples $\leftarrow$ numSamples - 1
11:         decrease $\leftarrow$ decrease + 1
12:     end while
13: end for
14: return $D'$
```

First, we calculate the exact number of samples to be created (line 1) and then retrieve the set of minority paths using Algorithm 3 (line 2). To force a distribution of the number of samples to be increased among the minority label paths, in line 3, the maximum increase per label path is calculated. In line 4, the average number of samples per label path is obtained to establish another limit in the sample's duplication distribution. In the algorithm's main loop (lines 5–12), the samples from each minority path are randomly duplicated until reaching the maximum increase, which is determinate between the division between the total number of samples to increase and the number of minority paths, or until the mean size of the label paths in the dataset is reached.

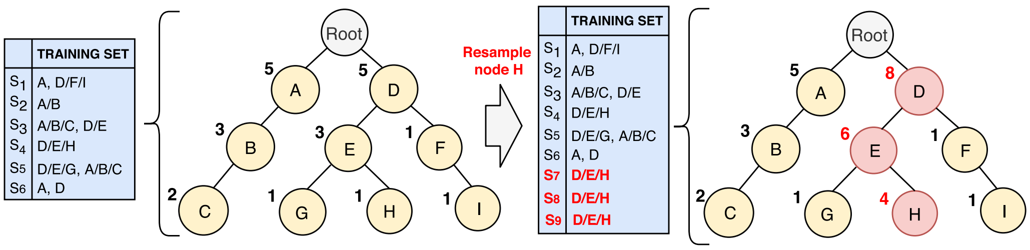
3.3. Resampling partial depth hierarchical classification problems

In this type of hierarchical classification problem, the instances may be associated with one or more label paths with full or partial depth in the label tree. The challenge of resampling this type of data is that when creating or removing samples from children nodes, the number of samples from parent label nodes will be indirectly increased or removed. This problem does not affect full-depth hierarchical classification problems because there are no samples labeled exclusively with internal nodes. Fig. 4 presents an example of this issue for a given training set with six samples (S_1-S_6) labeled with the same label tree shown in Fig. 1. We simulated the duplication of three samples labeled with node *H* to show its impact on the internal nodes. We observed that when we created these samples for node *H*, we indirectly created samples for nodes *E* and *D*.

To address this issue, we propose a technique to process the instances in a "leaf-node order", recalculating the majority/minority paths in each loop of the resampling process. Fig. 5 shows a visual example of the proposed method. For this specific example, the resampling process takes a total of three steps. The example dataset is composed of 85 samples and a label tree with nine nodes. We simulated the application of an oversampling method with an increase rate of 15%, that is, 12 samples.

Table 2 shows the variation in the imbalance ratio (IRLbP) during the resampling steps. The numbers in bold are greater than the HMeanIR, which is 8.45. In the first step, the proposed method processes nodes *C*, *G* and *I*, because they belong to the set of minority paths, randomly duplicating three samples from each of these label paths. The number of samples to be duplicated is calculated by dividing the 12%–15% increase rate by the total number of minority label paths in the dataset, which is five for this example. In step 2, only node *F* is resampled. It is important to observe that in step 1, node *B* belonged to the set of minority paths (with an IRLbP of 11.25); however, because node *C* was resampled, node *B* was also indirectly resampled (IRLbP changed to 7.29) and thus no longer belonged to the minority set (HMeanIR was 8.45). In the third and last steps, no nodes were resampled.

Considering the previous example, we propose random over/under-sampling for partial depth hierarchical classification problems (HROS-PD/HRUS-PD). The pseudocodes of these methods are presented in Algorithms 6 and 7, respectively. As the procedure followed by HROS-PD and HRUS-PD is similar, we will present a detailed explanation concerning HROS-PD and highlight the main differences in relation to HRUS-PD.



	TRAINING SET
S_1	A, D/F/I
S_2	A/B
S_3	A/B/C, D/E
S_4	D/E/H
S5	D/E/G, A/B/C
S6	A, D

	TRAINING SET
S_1	A, D/F/I
S_2	A/B
S_3	A/B/C, D/E
S_4	D/E/H
S5	D/E/G, A/B/C
S6	A, D
S7	D/E/H
S8	D/E/H
S9	D/E/H

Fig. 4. An example of the main issue when creating samples in leaf nodes. The numbers on the top left of the nodes symbolize the number of samples belonging to each node.

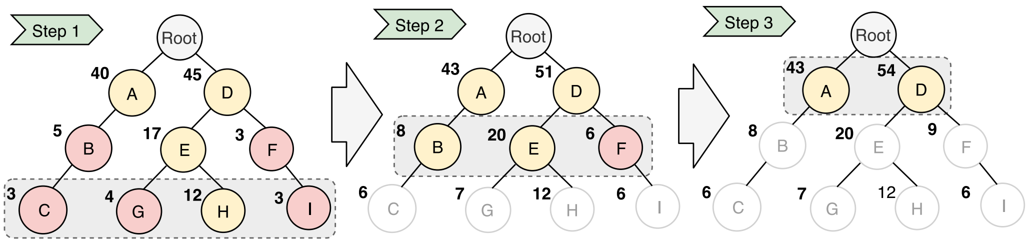


Fig. 5. An example of application of the HROS-PD in a dataset with 85 instances. The nodes marked with a circled dashed are being processed at the certain step and the red nodes represent the label paths belonging to the minority set.

Table 2
The imbalance ratios after each step from Figure 6. Numbers in bold are above the mean imbalance ratio – *HMeanIR*.

	A	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*
Step 1	1.13	**11.25**	**15.00**	1.00	2.65	**15.00**	**11.25**	3.75	**15.00**
Step 2	1.19	7.29	**8.50**	1.00	2.55	**8.50**	7.29	4.25	**8.50**
Step 3	1.26	7.71	**9.00**	1.00	2.70	6.00	7.71	4.50	**9.00**

Algorithm 6. Pseudocode for HROS-PD

```
Inputs: $D$: The hierarchical dataset,
S: Percentage of samples to increase (default = 10%)
Output: $D'$: An oversampled dataset
 1: samplesToCreate $\leftarrow |D| \times S$
 2: labelTree $\leftarrow$ retrieve label tree from $D$
 3: minPaths, HMeanIR $\leftarrow$ getMinorityPaths($D$)
 4: meanSize $\leftarrow$ calculate the average number of samples per labelPaths
 5: maxIncrease $\leftarrow$ samplesToCreate/ $|minPaths|$
 6: while labelTree is not empty do
 7:     for each leafNode in labelTree do
 8:         if leafNode in minPaths then
 9:             numSamples $\leftarrow$ samplesWithLabelPath($D$, leafNode)
10:             increased $\leftarrow$ 0
11:             while increased < maxIncrease and numSamples < meanSize do
12:                 $D'$ $\leftarrow$ randomly duplicate sample (last labeled with leafNode)
13:                 numSamples $\leftarrow$ numSamples + 1
14:                 increased $\leftarrow$ increased + 1
15:             end while
16:         end if
17:         remove leafNode from labelTree
18:     end for
19:     minPaths $\leftarrow$ getMinorityPaths($D$, HMeanIR)
20: end while
21: return $D'$
```

Algorithm 7. Pseudocode for HRUS-PD

```
Inputs: $D$: The hierarchical dataset,
S: Percentage of samples to increase (default = 10%)
Output: $D'$: An undersampled dataset
 1: samplesToRemove $\leftarrow |D| \times S$
 2: labelTree $\leftarrow$ retrieve label tree from $D$
 3: maxPaths, HMeanIR $\leftarrow$ getMajorityPaths($D$)
 4: meanSize $\leftarrow$ calculate the average number of samples per labelPaths
 5: maxDecrease $\leftarrow$ samplesToRemove/ $|maxPaths|$
 6: while labelTree is not empty do
 7:     for each leafNode in labelTree do
 8:         if leafNode in maxPaths then
 9:             numSamples $\leftarrow$ samplesWithLabelPath($D$, leafNode)
10:             decreased $\leftarrow$ 0
11:             while decreased < maxDecrease and numSamples > meanSize do
12:                 randomly remove sample (preferably last labeled with leafNode)
13:                 decreased $\leftarrow$ decreased + 1
14:                 numSamples $\leftarrow$ numSamples - 1
15:             end while
16:         end if
```

(*continued on next page*)

```
17:      end for
18:      remove leafNode from labelTree
19:      maxPaths ← getMajorityPaths($D$, HMeanIR)
20: end while
21: return $D'$
```

The first five lines of the algorithm represent the preparation phase and are similar to HROS-FD, with only a few differences. The first difference is in line 2, in which the dataset label tree is obtained. The second difference is when calculating the set of minority paths, in which the mean imbalancenss (HMeanIR) also have to be obtained by the algorithm. The main process of the algorithm is thought to be looping from lines 6 to 20. The idea is to resample each leaf node that belongs to the set of minority paths and re-calculating the set of minority paths in each looping step. The duplicating process is similar to that of HROS-FD. An important difference is that in line 12, in which a sample is randomly duplicated, the chosen sample is preferably lastly labeled with the target leaf node, that is, if the algorithm resamples an internal label path $x/y/z$, the sample chosen to be duplicated has also to be labeled with $x/y/z$, save exceptions in which there are no samples under this circumstance. This procedure avoids the fact that the duplication of samples from internal nodes affects the children nodes already processed by the algorithm.

An important issue in the proposed algorithm is that when re-calculating the set of minority paths (line 19 of Algorithms 6 and 7), the HMeanIR first obtained from the dataset (line 3 of Algorithms 6 and 7) has to be used as the threshold for the selection of the minority label paths in the looping of the resampling process (lines 6 to 20 of Algorithms 6 and 7). Although this may seem counterintuitive, it was a necessary design decision to cover the different types of hierarchical classification problems, especially the non-mandatory leaf node prediction cases. Let us elaborate on this design decision. During the first experiments, we tested to update HMeanIR while walking inside the label tree; however, the classification results were not satisfactory. While analyzing the resampled datasets, we realized that if we update HMeanIR in every step, we create many more samples in the leaf nodes than in the internal nodes. Because partial depth problems may also have samples labeled with only internal nodes, the creation of samples for these specific nodes is crucial, so that the learners can distinguish when to stop during the predictions.

3.4. Time complexity analysis

Table 3 shows the time complexity of the proposed resampling algorithms (HROS-FD/PD and HRUS-FD/PD) using the "big O" notation [35], that is, considering the superior limit for the execution time. Variable D computes the number of samples in the dataset, LP stands for the number of label paths, S is the number of samples that will be increased/decreased by the method, and L is the number of individual labels in the dataset, that is, the number of nodes in the label tree.

Analyzing the time complexity of the proposed methods, we may observe that the partial depth methods, that is, HROS-PD and HRUS-PD, have a higher time complexity than the full depth methods (HROS-PD and HRUS-PD). This occurs because, as the partial depth algorithms have to deal with the sub-paths, the main loop (lines 6–20 of Algorithms 6 and 7) have to walk through the label tree, recalculate the imbalance ratio, and generate possible new instances for each label in the hierarchy. In contrast, the full depth algorithms (HROS-FD and HRUS-FD) deal only with full label paths.

4. Experimental protocol and results

In this section, we present the datasets, algorithms, parameters, experimental setup, and classification results of the proposed resampling algorithms.

4.1. The datasets

To cover the different aspects of hierarchical classification problems, we performed computational experiments on 23 datasets: nine with full depth problems and 14 with partial depth problems. As some of these datasets were adapted and hence are somehow novel, presenting characteristics concerning the different taxonomies of hierarchical classification problems as defined by Silla Jr. and Freitas (2011) [1], they form a testbed for hierarchical classification researchers and, thus, may also be considered a contribution of this paper. It is important to state that all datasets, algorithm implementations, and detailed results can be obtained in this link.[1]

Tables 4 and 5 present a detailed description of the datasets with full depth and partial depth problems, respectively. The datasets marked with (*) and (**) are somewhat novel. The datasets marked with (*) were originally proposed as flat multi-label classification datasets in the literature, and in this work, were adapted to a hierarchical taxonomy. The datasets marked with (**) were extracted as single-label subsets from the original multi-label dataset and were adapted to a hierarchical taxonomy. In the following, we provide details about how these datasets were adapted.

[1] https://sites.google.com/view/hierarchical-imblearn/

Table 3
Time complexity of the multi-label resampling algorithms.

Resampling Method	Time Complexity
HROS-FD HRUS-FD	O(LP × (D + S))
HROS-PD HRUS-PD	O(L × (D + S + LP))

Table 4
Datasets with Full Depth Hierarchical Classification Problems.

Name	Paths	Domain	Train	Test	Attr.	Labels	Labels p/ level	Ref.
Enron*	Multiple	Text	988	660	1001	57	3, 40, 14	[36]
CAL500*		Music	351	151	68	164	7, 76, 78, 3	[37]
Emotions*			392	203	72	9	3, 6	[38]
Birds*		Biology	272	79	260	49	13, 17, 19	[39]
Actinopterygii	Single		15705	6739	15	30	2, 6, 12, 15	[40]
Diptera			15194	6528	33	29	4, 6, 9, 10	
Instrument		Audio	6583	2836	30	46	5, 10, 31	
Hglass		Glass	144	70	9	11	2, 3, 5, 1	[41]
ImCLEF07D		Image	10000	1006	80	24	4, 9, 11	[5]

Table 5
Datasets with Partial Depth Hierarchical Classification Problems.

Name	Paths	Domain	Train	Test	Attr.	Labels	Labels p/ level	Ref.
Cell-cycle	Multiple	Biology	2484	1281	78	180	4, 22, 70, 84	[42]
Church			2474	1281	24	180	4, 22, 70, 84	
Derisi			2450	1275	62	180	4, 22, 70, 84	
Eisen			1587	837	80	170	4, 22, 66, 78	
Exp			2488	1291	544	180	4, 22, 70, 84	
Gasch-1			2480	1284	174	180	4, 22, 70, 84	
Gasch-2			2488	1291	53	180	4, 22, 70, 84	
Phenotype			1009	582	64	168	4, 22, 66, 76	
Sequence			2580	1339	437	180	4, 22, 70, 84	
SPO			2437	1266	79	180	4, 22, 70, 84	
FMA-MFCC*		Music	33259	14274	13	97	12, 66, 19	[43]
FMA-SLLBP**	Single		15331	7000	59	135	25, 91, 19, 1	
FMA-SLSSD**			15631	6700	161	135	25, 91, 19, 1	
Diatoms		Image	2065	1054	371	398	82, 313, 3	[44]

Looking at the dataset characteristics, we can observe that FMA-MFCC is by far the largest, while Hglass is the smallest. Furthermore, Hglass is the dataset with the lowest number of attributes, whereas the Enron dataset has the most.

Fig. 6 presents graphics of the HMeanIR for the datasets used in the experimental analysis. We may observe that the datasets present a large variety of imbalance. The full depth classification problems datasets are much less imbalanced than the datasets with partial depth, with Emotions dataset reaching a HMeanIR of only 1.49. In contrast, Church, Derisi, Exp, Gasch-1, Gasch-2, Sequence, and SPO are the most unbalanced datasets, with HMeanIR close to 800.00.

It is important to note that all datasets presented in Tables 4 and 5 have labels in a tree taxonomy, because we are not dealing with directed acyclic graph taxonomies in this work.

4.1.1. New hierarchical datasets

As we proposed adaptations of existing multi-label datasets from the Laplace in order to represent the different hierarchical classification problems, we explain how these datasets were adapted.

Enron is an e-mail dataset in which messages were originally annotated with multiple subjects. The labels were hierarchically organized according to the subjects to transform the dataset in a hierarchical format. For instance, messages with emotional tone can be sub-categorized into humor, sarcasm, sadness, etc.

Birds is a dataset composed of ambient audio with multiple birds singing. As these birds have different or similar biological families, genera, and species, we hierarchically organized them according to their biological relationships. For instance, turdidae/catharus/guttatus and turdidae/catharus/ustulatus.

CAL500 is a music dataset labeled with a series of annotations, such as genres, emotions, and instruments. These labels are hierarchically organized according to their context. For instance, emotions were sub-categorized into emotion/positive/

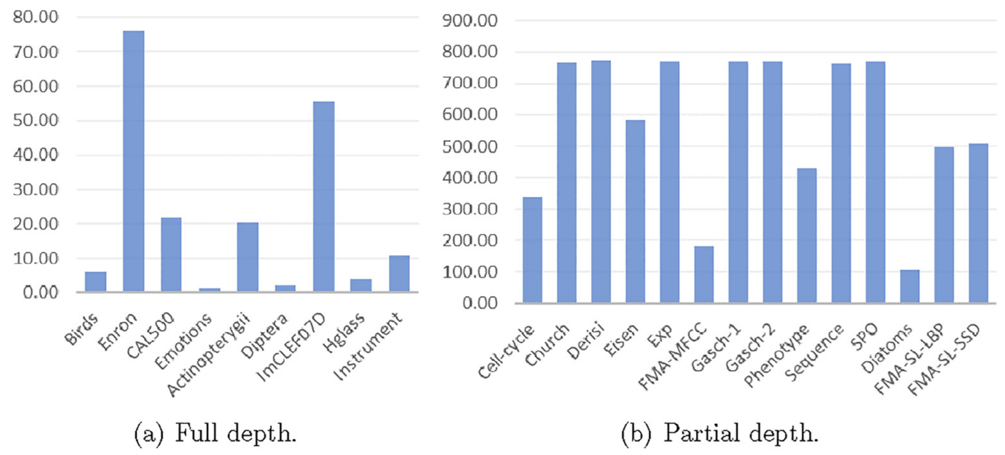


Fig. 6. Mean Imbalance Ratios (HMeanIR) for the Datasets.

happy, emotion/positive/exciting, emo/neutral/calming, etc. The genres were organized into rock/classic-rock, rock/hard-rock, etc.

Emotions is a music mood dataset and its hierarchical transformation is similar to the CAL500 emotion annotations, using positive, neutral, and negative categories.

FMA-MFCC was first introduced by Defferrard et al. (2017) [43] and is a music genre dataset extracted from the Free Music Archive (FMA) website. This dataset was also hierarchically annotated following the same reasoning as the CAL500 genres annotations.

Furthermore, FMA-SLLBP and FMA-SLSSD are subsets extracted from FMA. Although FMA is originally multi-labeled, FMA-SLLBP and FMA-SLSSD are composed of only the samples from FMA, which are single-labeled. The differences between these datasets are the types of features extracted from the songs, that is, Mel-Frequency Cepstral Coefficients (MFCC), Local Binary Patterns (LBP), or Statistical Spectrum Descriptors (SSD).

4.1.2. Hierarchical theoretical complexity score

Charte et al. (2016) [45] proposed a method for measuring the complexity of multi-label datasets, which is called the theoretical complexity score (TCS). The TCS is based on the number of features, labels, and different label sets in the dataset. The greater the TCS of a dataset, the more complex it is. The TCS metric can help explain why it is difficult to achieve satisfactory results for a given dataset. In fact, considering the resampling scenario, the TCS metric can also help to explain why the resampling algorithms are more or less effective in certain datasets.

Because the TCS metric was only defined for the multi-label classification scenario, in this work, we propose a new metric, called the hierarchical theoretical complexity score (HTCS), which is an adapted version of TCS considering the hierarchical aspects of the dataset. Eq. 5 shows the proposed metric. In HTCS, D represents the dataset, f is the number of features, LVL is the number of levels of the label tree, L_i is the number of labels in level i, and lps is the number of different sets of label paths in the dataset.

$$HTCS(D) = \ln\left(f + \sum_{i=1}^{LVL}(i \times L_i) + lps\right) \tag{5}$$

Table 6 shows the HTCS of the hierarchical datasets presented in Tables 4 and 5. We may observe that while Hglass is the least complex dataset, with an HTCS of 7.285, Exp is the most complex, with a HTCS of 20.102.

4.2. Classification algorithm and parameters

For the hierarchical classification task, we used the Clus-HMC framework,[2] which is a state-of-the-art hierarchical classification framework [46,47].

Clus-HMC is based on predictive cluster trees (PCT) and generates a single decision tree (DT) considering an entire class hierarchy. In Clus-HMC, DTs are seen as a hierarchy of clusters where the root node contains all the training instances, while the remaining are recursively divided into smaller groups as the hierarchy is traversed toward the leaves. The classification is performed using a distance-based metric that calculates how similar an instance is to a tree. The parameter configurations used in the algorithm were obtained after applying an extensive grid search, as proposed by Bergstra and Bengio (2012) [48], and are listed in Table 7.

[2] Available for download at https://dtai.cs.kuleuven.be/clus/

Table 6
HTCS of the datasets used in the experiments.

Dataset	Number of Features	Labels p/ level				Sets of Label Path	HTCS
		1	2	3	4		
Hglass	9	2	3	5	1	6	7.285
actinopterygii	15	2	6	12	15	15	10.117
Emotions	72	3	6	0	0	26	10.243
diptera	33	4	6	9	10	14	10.555
instrument	30	5	10	31	0	31	11.606
ImCLEF07D	80	4	9	11	0	26	11.648
FMA-SL-LBP	59	25	91	19	1	125	14.497
FMA-MFCC	13	12	66	19	0	1125	14.894
Birds	260	13	17	19	0	116	14.959
FMA-SL-SSD	161	25	91	19	1	125	15.501
CAL500	68	7	76	78	3	351	16.085
Church	24	4	22	70	84	1655	16.977
Phenotype	64	4	22	66	76	789	17.140
Eisen	80	4	22	66	78	1114	17.723
Gasch-2	53	4	22	70	84	1661	17.773
Enron	1001	3	40	14	0	442	17.829
Derisi	62	4	22	70	84	1638	17.916
SPO	79	4	22	70	84	1639	18.159
Cell-cycle	78	4	22	70	84	1666	18.162
Diatoms	371	82	313	3	0	306	18.215
Gasch-1	174	4	22	70	84	1656	18.959
Sequence	437	4	22	70	84	1673	19.890
Exp	544	4	22	70	84	1661	20.102

4.3. Experimental setup

In this study, we used the *weighted* area under the AUPRC to measure the classification results (equation details shown in subSection 2.3) and tested seven resize rates commonly used by researchers in the resampling tasks: 5%, 10%, 15%, 20%, 25%, 30% and 35%.

The results presented in Tables 8–12 are the average of 10 executions of Clus-HMC after re-applying the proposed resampling algorithms in the training sets. In these executions, we used the same training/test split, as presented in Tables 4 and 5. The average of 10 executions is important because we have a random factor in the experiments, as we used random forest as the ensemble method, as presented in Table 7.

4.4. Results

Table 8 presents the classification results before and after applying the HROS-FD and HRUS-FD algorithms in the training sets of the full depth hierarchical datasets with single-path prediction. In this table, we highlight the Hglass dataset, which was the most benefited from the resampling algorithms and was able to reach an AUPRC of 0.9394 after applying HROS-FD with 10% of samples increase (0.1249 more than the original dataset). However, the Diptera dataset was the least affected by the resampling algorithms, with the best result achieving an increase of only 0.0079.

Table 9 shows the classification results for the datasets with full depth prediction and multiple paths. We may observe that, in general, the resampling algorithms were not very effective, with the best result being reached after applying HROS-FD with a 10% increase in the Emotions dataset, in which the results were 0.0436 greater than using the original dataset.

Table 10 shows the classification results before and after applying the HROS-PD and HRUS-PD in the partial depth prediction hierarchical datasets with single paths. These results highlight the fact that the best results with HROS-PD outperformed all HRUS-PD results. Another consideration that can be made is the poor performance of HRUS-PD on the Diatoms dataset.

Tables 11 and 12 present the classification results for the datasets with partial depth and multiple-path prediction. We may note in these tables that while the Cell-cycle was the dataset most benefited by the resampling methods, reaching an increase of 0.0429 using HROS-PD with a 10% resampling rate, the SPO dataset was the least affected by the resampling algorithms, achieving a maximum increase of only 0.0034.

5. Analysis and discussion

Observing the previously described results, four main questions are raised: (i) Can the proposed resampling algorithms increase the classification results? (ii) Which is the most/least effective resampling algorithm? (iii) Does resize rate influence the classification results? (iv) Can we define a default value for the resize rate? (v) Does the HMeanIR dataset influence the resampling and classification results? (vi) Which one is better: HROS or HRUS?.

Table 7
Clus-HMC execution parameters.

Parameter	Meaning	Value
Type	Type of problem (Tree or DAG)	Tree
ConvertToRules	Convert the trees into a set of rules	No
HSeparator	Character used as label separator in the dataset file	"/"
FTest	Stopping criterion for regression	{0.001, 0.005, 0.01, 0.05, 0.1, 0.125}
EnsembleMethod	Method used rightarrow combine the predictions	RForest
Iterations	Defines the number of trees in the ensemble	10
VotingType	Voting scheme for combining the predictions	Majority
EnsembleRandomDepth	Use different random depth for each tree selected	No
SplitSampling	The split used on the training set for the heuristic	None

Table 8
Results for the full depth hierarchical datasets with single paths.

		Actinopterygii	Diptera	ImCLEF07D	Hglass	Instrument
	Original	0.7570	0.6027	0.7151	0.8145	0.7656
HROS-FD	5%	0.7746	0.6003	0.7041	0.8394	0.7698
	10%	**0.7869**	0.6042	**0.7292**	**0.9394**	**0.7873**
	15%	0.7658	0.6094	0.7042	0.8880	0.7692
	20%	0.7530	0.5977	0.7069	0.8759	0.7584
	25%	0.7505	0.5969	0.6927	0.8485	0.7625
	30%	0.7502	0.5863	0.6835	0.8706	0.7602
	35%	0.7571	0.5953	0.6941	0.8646	0.7499
HRUS-FD	5%	0.7680	0.6086	0.7185	0.8222	0.7726
	10%	0.7541	0.6026	0.7210	0.7944	0.7633
	15%	0.7512	0.6033	0.7182	0.8910	0.7569
	20%	0.7516	0.6089	0.6937	0.8161	0.7222
	25%	0.7520	**0.6106**	0.6882	0.8214	0.7280
	30%	0.7397	0.6082	0.6880	0.7783	0.7152
	35%	0.7312	0.6042	0.6623	0.7868	0.7090

Table 9
Results for the full depth hierarchical datasets with multiple paths.

		Birds	Enron	CAL500	Emotions
	Original	0.4536	0.5693	0.4942	0.6843
HROS-FD	5%	0.4327	0.5720	0.4874	0.7114
	10%	0.4498	**0.5990**	0.4938	**0.7279**
	15%	0.4333	0.5789	0.5043	0.6951
	20%	0.4454	0.5656	**0.5174**	0.6980
	25%	0.4459	0.5516	0.4925	0.6888
	30%	0.4475	0.5578	0.4942	0.6931
	35%	0.4444	0.5628	0.4874	0.7101
HRUS-FD	5%	0.4308	0.5672	0.4874	0.6943
	10%	0.4224	0.5690	0.4945	0.7012
	15%	**0.4645**	0.5630	0.4941	0.6922
	20%	0.4301	0.5635	0.4945	0.6954
	25%	0.4365	0.5676	0.4874	0.6821
	30%	0.4106	0.5585	0.4945	0.6856
	35%	0.3952	0.5623	0.4947	0.6890

To answer the first question with statistical significance, we applied the Wilcoxon statistical test, stating that the *weighted* AUPRC of the classification is different after using the resampling methods. The test was applied for each of the seven different resize rates, and the *p-values* outputs for the full depth (HROS-FD and HRUS-FD) and partial depth algorithms (HROS-PD and HRUS-PD) are shown in Tables 13 and 14, respectively. Considering the threshold as 0.05, it is safe to say that all resampling algorithms improved the results in particular scenarios. While HROS-FD statistically improved the classification results of the single-path datasets when using a 10% increase rate, HRUS-FD improved the results of the single-path datasets with a decrease rate of 5%. Moreover, HROS-PD significantly improved the classification results for the datasets with multiple paths when using a 5% and 10% increase rate, and HRUS-PD improved the results for the multiple-path datasets with a decrease rate of 10%. Interestingly, it is possible to note that while the FD resampling algorithms performed better with single-path datasets, the PD methods were better in the multiple-path classification problems.

Table 10
Results for the partial depth hierarchical datasets with single paths.

		Diatoms	FMA-SL-LBP	FMA-SL-SSD
	Original	0.2582	0.3268	0.2700
HROS-PD	5%	0.2445	0.3173	0.2717
	10%	0.2646	0.3204	0.2776
	15%	0.2659	**0.3484**	**0.2918**
	20%	**0.2887**	0.3307	0.2754
	25%	0.2728	0.3277	0.2760
	30%	0.2544	0.3107	0.2668
	35%	0.2623	0.3159	0.2633
HRUS-PD	5%	0.2243	0.3271	0.2721
	10%	0.2401	0.3293	0.2726
	15%	0.2431	0.3200	0.2663
	20%	0.2393	0.3371	0.2716
	25%	0.2206	0.3376	0.2697
	30%	0.2333	0.3238	0.2659
	35%	0.2223	0.3248	0.2717

Table 11
Results for the partial depth hierarchical datasets with multiple paths (Part 1).

		Cell-cycle	Church	Derisi	Eisen	Exp	FMA-MFCC
	Original	0.1307	0.1222	0.1309	0.1483	0.1606	0.2803
HROS-PD	5%	0.1654	0.1347	0.1404	0.1590	0.1672	0.2890
	10%	**0.1736**	**0.1352**	**0.1586**	0.1719	0.1585	0.2628
	15%	0.1632	0.1304	0.1425	0.1603	**0.1753**	0.2592
	20%	0.1594	0.1264	0.1450	0.1513	0.1660	0.2588
	25%	0.1415	0.1279	0.1366	0.1557	0.1660	0.2572
	30%	0.1424	0.1230	0.1261	0.1642	0.1526	0.2533
	35%	0.1465	0.1204	0.1264	0.1625	0.1600	0.2560
HRUS-PD	5%	0.1458	0.1217	0.1330	0.1589	0.1634	0.2908
	10%	0.1425	0.1287	0.1307	0.1578	0.1636	0.2783
	15%	0.1471	0.1221	0.1312	**0.1797**	0.1553	**0.3022**
	20%	0.1481	0.1264	0.1310	0.1486	0.1642	0.2820
	25%	0.1422	0.1217	0.1311	0.1465	0.1585	0.2910
	30%	0.1446	0.1304	0.1307	0.1473	0.1597	0.2812
	35%	0.1411	0.1254	0.1229	0.1501	0.1486	0.2817

Table 12
Results for the partial depth hierarchical datasets with multiple paths (Part 2).

		Gasch-1	Gasch-2	Phenotype	Sequence	SPO
	Original	0.1544	0.1410	0.1256	0.1683	0.1342
HROS-PD	5%	0.1590	0.1492	0.1318	0.1655	0.1357
	10%	**0.1868**	**0.1654**	0.1303	0.1679	0.1359
	15%	0.1733	0.1549	0.1272	0.1598	0.1277
	20%	0.1636	0.1469	0.1237	0.1637	0.1265
	25%	0.1581	0.1466	0.1256	0.1761	**0.1376**
	30%	0.1596	0.1415	0.1294	**0.1886**	0.1264
	35%	0.1611	0.1480	0.1297	0.1597	0.1317
HRUS-PD	5%	0.1560	0.1411	0.1240	0.1665	0.1344
	10%	0.1585	0.1540	0.1260	0.1674	0.1347
	15%	0.1510	0.1399	0.1276	0.1675	0.1348
	20%	0.1569	0.1388	0.1344	0.1626	0.1337
	25%	0.1496	0.1301	0.1326	0.1620	0.1343
	30%	0.1521	0.1410	**0.1370**	0.1626	0.1346
	35%	0.1451	0.1368	0.1255	0.1587	0.1352

Moreover, although the Wilcoxon tests were not able to identify a significant improvement at threshold 0.05, we cannot affirm that HRUS-FD (using a decrease of 25%, 30%, and 35%), HROS-PD (using an increase rate of 25%), and HRUS-PD (using a decrease rate of 5%) certainly did not improve the results because their p-values are very close to the threshold. In fact, if we observe the classification results in Tables 8–12, we may note some improvements in terms of AUPRC for these cases.

To answer the second question, we also analyze Tables 13 and 14. We may note that HROS-PD was the only one that statistically outperformed the original classification results when using two different rates—5% and 10%. Furthermore, the

Table 13
P-values of the Wilcoxon signed-rank statistical test for the Full Depth Random Resampling Algorithms.

	HROS-FD		HRUS-FD	
(%)	Single Path	Multiple Path	Single Path	Multiple Path
5	0.3452	1.0000	**0.0431**	0.4652
10	**0.0431**	0.4652	0.3452	0.7150
15	0.3452	0.7150	0.8927	0.4652
20	0.5002	0.4652	0.3452	0.7150
25	0.5002	0.2733	0.5002	*0.0679*
30	0.5002	0.5930	*0.0796*	0.4652
35	0.6858	0.7150	*0.0796*	0.4652

Numbers in bold are below the threshold and in italics are close to the threshold.

Table 14
P-values of the Wilcoxon signed-rank statistical test for the Partial Depth Random Resampling Algorithms.

	HROS-PD		HRUS-PD	
(%)	Single Path	Multiple Path	Single Path	Multiple Path
5	0.2850	**0.0058**	1.0000	*0.0828*
10	0.2850	**0.0409**	1.0000	**0.0409**
15	0.1088	0.1307	0.1088	0.4769
20	0.1088	0.3281	1.0000	0.1549
25	0.1088	*0.0743*	0.5930	0.8589
30	0.1088	0.6566	0.1088	0.5751
35	0.2850	0.7897	0.2850	0.4236

Numbers in bold are below the threshold and in italics are close to the threshold.

HROS-PD was also close to the threshold when using a 25% increase rate. Finally, although in the single-path datasets, HROS-PD did not statistically overperform the original results using any increase rate, we may observe that in four of the seven p-values, the value was 0.1088, that is, only 0.05 above the threshold. In contrast, we may note that only HRUS-PD obtained an exact *p*-value 1.0 of, for three decreasing rates (5%, 10%, and 15%), which indicates that it is a less effective resampling method. Moreover, HROS-FD performed poorly in all scenarios, with the exception of single paths with a 10% increase rate.

The third question may be answered by looking at the different ranges of *p*-values presented in Tables 13 and 14. It is notable that the post-resampled results that overperformed the original classification results were obtained with resize rates of 5% and 10%, despite the resampling algorithm or type of dataset. Thus, we can confirm that the resize rate influences the classification results.

To answer the fourth question, we present in Table 15 the computation of the number of times the best result was achieved with each of the resize rates in Tables 8–12. We may observe that there is no absolute unanimity among the resize rates, as all of them achieved the best result in at least one resampling scenario. However, we may also note the higher incidence of 10%, which is the best resize rate by far among the oversampling algorithms, that is, HROS-FD and HROS-PD. Thus, based on the experiments, we may define a default resize rate of 10% (as shown in Algorithms 4–6 from Section 3). It is important to note that the definition of a heuristic method to pick the best resize rate for a given dataset is not a trivial task because each dataset has its intrinsic characteristics.

The answers to the fifth question are first grounded in Fig. 6 and Table 6. First, analyzing the HMeanIR, we may note that Enron, IMCLEF07D, Church, Derisi, Exp, Gasch-1, Gasch-2, Sequence, and SPO are the most imbalanced datasets, in contrast with Birds, Emotions, Diptera, Hglass, and Diatoms, which are the least imbalanced datasets. However, as shown in Tables 8–12, it can be observed that the most imbalanced datasets were not necessarily the most benefited by the resampling algorithms. As a counter example, let us consider the Hglass datasets, which have HMeanIR of only 4.04. In this case, the result with the original datasets (0.8145) had the largest improvements (0.9394) between all datasets (after applying HROS-FD with an increase of 10%). Therefore, although the dataset imbalance does influence the resampling and classification results, we may not conclude that this relation is necessarily inverse, that is, when the imbalance is low, the resampling algorithms will not significantly impact the classification results. Moreover, Table 6 explains how imbalance is not the only factor that influences the classification results. For example, we can observe that Hglass is the least complex dataset in terms of HTCS. This fact can enable the classifiers to learn more details about the instances when using resampling algorithms in the training set, because the dataset is theoretically "simple". On the other hand, we may also observe that the SPO dataset, which is one of the most complex datasets in terms of HTCS, was the least benefited by the resampling algorithms, even with a high HMeanIR.

The answer to the last question starts with the statement that, as can be seen in Tables 8–12, the proposed oversampling methods (HROS-FD and HROS-PD) outperformed the proposed undersampling techniques (HRUS-FD and HRUS-PD) in most of the classification scenarios. Thus, we may conclude that oversampling is somehow better than undersampling in

Table 15
Number of times that the best result was achieved with each resize rate.

Resize Rate	HROS-FD	HRUS-FD	HROS-PD	HRUS-PD
5%		2	2	1
10%	7	3	6	5
15%	1	2	3	5
20%	1		1	2
25%		1	1	2
30%			1	1
35%		1		

the context of this study. The experimental results achieved here can be supported by other studies from the literature, such as Buda et al. (2018) and Mohammed *et al.* (2020) [49,50], in which the undersampling techniques also had poor performance. In fact, the undersampling methods commonly cause a loss of information in the training set, mostly in random methods, because they can remove instances with feature sets that are crucial to the learning process. For example, the removal of instances located in the borderline of the feature spaces can prejudice the learner to distinguish the sample's classes. Moreover, analyzing the datasets in which the undersampling outperformed the oversampling, that is, Diptera, FMA-MFCC, Birds, Phenotype, and Eisen, we can observe that, in general, their HTCS (presented in Table 6), are relatively below or near the average, which may lead to the hint that theoretical complex datasets are not well fitted to the use of undersampling, corroborating the information loss factor.

6. Concluding remarks and future works

Addressing imbalanced data distribution is a difficult task for many classification algorithms. Resampling the training data toward a more balanced distribution is one of the most common and effective ways to combat this issue, independent of the classifier. Although this issue has been widely studied in the literature, the authors usually focus on flat classification contexts, ignoring scenarios with a hierarchy between the labels. In addition, no study has proposed a resampling algorithm capable of pre-processing a hierarchical dataset as a whole. Considering this gap, we propose novel methods to handle imbalance in hierarchical classification problems in this study.

We proposed four novel resampling algorithms: random oversampling and undersampling for hierarchical datasets with full depth prediction (HROS-FD and HRUS-FD), and random oversampling and undersampling for hierarchical datasets with partial depth prediction (HROS-PD and HRUS-PD). These algorithms are, to the best of our knowledge, the first ones to deal with imbalance in hierarchical datasets, handling and resampling the data in a direct way.

To retrieve the set of majority/minority label paths in a given hierarchical dataset, we proposed the use of the IRLbP and HMeanIR, previously defined in the literature. While the full depth resampling algorithms deal with the data in unique looping though the majority/minority label paths, the partial depth algorithms handle the data walking thought the label tree in a leaf-nodes order, resampled their instances, and re-calculated the IRLbPs to retrieve updated sets of majority/minority paths.

Experimental results with 23 datasets with characteristics ranging from partial/full depth prediction to single/multiple paths showed that the proposed algorithms can statically improve the classification results in relation to *weighted* AUPRC for all scenarios.

It should be noted that the resampling algorithms proposed in this study can be considered as a baseline and starting point for a promising branch of investigations. The main approaches used in the algorithms show a way to handle an imbalanced hierarchical dataset considering its different aspects. Using these approaches, novel heuristic resampling can be further investigated. In future works, we intend to investigate the adaptation and/or creation of novel resampling algorithms considering specific heuristics, such as synthetic instance creation and nearest neighbor removal.

Finally, it is important to state that the proposed resampling approaches are focused only on hierarchical datasets with tree-based taxonomies, excluding the problems with directed acyclic graphs (DAG) taxonomy.

CRediT authorship contribution statement

Rodolfo M. Pereira: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Resources, Validation, Visualization, Writing - original draft. **Yandre M.G. Costa:** Supervision, Validation, Writing - review & editing. **Carlos N. Silla Jr.:** Project administration, Supervision, Validation, Writing - review & editing.

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.

Acknowledgments

We thank the Brazilian Research Support Agencies: Coordination for the Improvement of Higher Education Personnel (CAPES), National Council for Scientific and Technological Development (CNPq) and the Araucaria Foundation (FA) for their financial support.

References

[1] C.N. Silla Jr., A.A. Freitas, A survey of hierarchical classification across different application domains, Data Min. Knowl. Disc. 22 (1–2) (2011) 31–72.
[2] H. Tang, Y. Wang, S. Tang, D. Chu, C. Li, A randomized clustering forest approach for efficient prediction of protein functions, IEEE Access 7 (2019) 12360–12372.
[3] R.A. Stein, P.A. Jaques, J.F. Valiati, An analysis of hierarchical text classification using word embeddings, Inf. Sci. 471 (2019) 216–232.
[4] F. Saki, N. Kehtarnavaz, Real-time hierarchical classification of sound signals for hearing improvement devices, Appl. Acoust. 132 (2018) 26–32.
[5] I. Dimitrovski, D. Kocev, S. Loskovska, S. Džeroski, Hierarchical annotation of medical images, Pattern Recogn. 44 (10–11) (2011) 2436–2449.
[6] F. Thabtah, S. Hammoud, F. Kamalov, A. Gonsalves, Data imbalance in classification: Experimental evaluation, Inf. Sci. 513 (2020) 429–441.
[7] S. Kumar, H. A. Rowley, X. Wang, J. J. M. Rodrigues, Hierarchical classification in credit card data extraction, uS Patent 9213907 (December 2015).
[8] R.M. Pereira, D. Bertolini, L.O. Teixeira, C.N. Silla Jr, Y.M. Costa, Covid-19 identification in chest x-ray images on flat and hierarchical classification scenarios, Comput. Methods Programs Biomed. 194 (2020) 105532.
[9] P. Vuttipittayamongkol, E. Elyan, Neighbourhood-based undersampling approach for handling imbalanced and overlapped data, Inf. Sci. 509 (2020) 47–70.
[10] R. M. Pereira, Y. M. G. Costa, C. N. Silla Jr, MLTL: A multi-label approach for the tomek link undersampling algorithm, Neurocomputing C (383) (2020) 95–105.
[11] K. Shin, J. Han, S. Kang, MI-MOTE: Multiple imputation-based minority oversampling technique for imbalanced and incomplete data classification, Inf. Sci. (2021).
[12] V.H. Barella, L.P.F. Garcia, M.C.P. de Souto, A.C. Lorena, A.C.P.L.F. de Carvalho, Assessing the data complexity of imbalanced datasets, Inf. Sci. 553 (2021) 83–109.
[13] F. Wu, J. Zhang, V. Honavar, Learning classifiers using hierarchically structured class taxonomies, in: Proceedings of The International Symposium on Abstraction, Reformulation, and Approximation, 2005, pp. 313–320.
[14] A. Freitas, A. Carvalho, A tutorial on hierarchical classification with applications in bioinformatics, in: Research and trends in data mining technologies and applications, IGI Global, 2007, pp. 175–208.
[15] R. Cerri, G.L. Pappa, A.C.P. Carvalho, A.A. Freitas, An extensive evaluation of decision tree–based hierarchical multilabel classification methods and performance measures, Comput. Intell. 31 (1) (2015) 1–46.
[16] S. Kiritchenko, S. Matwin, F. Famili, Hierarchical text categorization as a tool of associating genes with gene ontology codes, in: Proceedings of the Second European Workshop on Data Mining and Text Mining in Bioinformatics, Pisa, Italy, 2004, pp. 30–34.
[17] S. Kiritchenko, S. Matwin, F. Famili, Functional annotation of genes using hierarchical text categorization, in: Proceedings of the ACL Workshop on Linking Biological Literature, Detroit, USA, 2005, pp. 1–4.
[18] J. Davis, M. Goadrich, The relationship between precision-recall and roc curves, in: Proceedings of the International Conference on Machine Learning, ACM, 2006, pp. 233–240.
[19] A. Guzmán-Ponce, J.S. Sánchez, R.M. Valdovinos, J.R. Marcial-Romero, Dbig-us: A two-stage under-sampling algorithm to face the class imbalance problem, Expert Syst. Appl. 168 (2021) 114301.
[20] W. Wang, D. Sun, The improved adaboost algorithms for imbalanced data classification, Inf. Sci. 563 (2021) 358–374.
[21] O.S. Sitompul, E.B. Nababan, et al, Biased support vector machine and weighted-SMOTE in handling class imbalance problem, Int. J. Adv. Intell. Inf. 4 (1) (2018) 21–27.
[22] S. Wang, X. Yao, Multiclass imbalance problems: Analysis and potential solutions, IEEE Trans. Syst. Man Cybern. Part B (Cybern.) 42 (4) (2012) 1119–1130.
[23] T. Hastie, R. Tibshirani, Classification by pairwise coupling, Adv. Neural Inf. Process. Syst. 11 (1) (1998) 507–513.
[24] R. Rifkin, A. Klautau, In defense of one-vs-all classification, J. Mach. Learn. Res. 5 (Jan) (2004) 101–141.
[25] F. Charte, A. Rivera, M. del Jesus, F. Herrera, MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation, Knowl.-Based Syst. 89 (2015) 385–397.
[26] N. Chawla, K. Bowyer, L. Hall, P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[27] G. Batista, R. Prati, M. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD Expl. Newsl. 6 (1) (2004) 20–29.
[28] A. Fernández, S. Garcia, F. Herrera, N.V. Chawla, SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary, J. Artif. Intell. Res. 61 (2018) 863–905.
[29] C. Mera, J. Arrieta, M. Orozco-Alzate, J. Branch, A bag oversampling approach for class imbalance in multiple instance learning, in: Proceedings of the Iberoamerican Congress on Pattern Recognition, Springer, 2015, pp. 724–731.
[30] H. Han, W.-Y. Wang, B.-H. Mao, Borderline-SMOTE: A new oversampling method in imbalanced datasets learning, in: International Conference on Intelligent Computing, Hefei, China, 2005, pp. 878–887.
[31] F. Charte, A. Rivera, M.J. del Jesus, F. Herrera, A first approach to deal with imbalance in multi-label datasets, in: Proceedings of The International Conference on Hybrid Artificial Intelligence Systems, 2013, pp. 150–160.
[32] F. Charte, A. Rivera, M. del Jesus, F. Herrera, Addressing imbalance in multilabel classification: Measures and random resampling algorithms, Neurocomputing 163 (2015) 3–16.
[33] R.M. Pereira, Y.M.G. Costa, C.N. Silla Jr., Dealing with imbalanceness in hierarchical multi-label datasets using multi-label resampling techniques, in: IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI), 2018, pp. 818–824.
[34] R.M. Pereira, Y.M. Costa, C.N. Silla, Handling imbalance in hierarchical classification problems using local classifiers approaches, Data Min. Knowl. Disc. (2021) 1–58.
[35] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, 3rd Edition., MIT Press, 2009.
[36] B. Klimt, Y. Yang, Introducing the Enron Corpus, in: Proceedings of the Conference on Email and Anti-Spam (CEAS), 2004, pp. 1–2.
[37] D. Turnbull, L. Barrington, D. Torres, G. Lanckriet, Towards musical query-by-semantic-description using the cal500 data set, in: Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2007, pp. 439–446.
[38] K. Trohidis, G. Tsoumakas, G. Kalliris, I. P. Vlahavas, Multi-label classification of music into emotions, in: Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Vol. 8, 2008, pp. 325–330.
[39] F. Briggs, Y. Huang, R. Raich, K. Eftaxias, Z. Lei, W. Cukierski, S.F. Hadley, A. Hadley, M. Betts, X.Z. Fern, et al, The 9th annual mlsp competition: New methods for acoustic classification of multiple simultaneous bird species in a noisy environment, in: Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP), IEEE, 2013, pp. 1–8.
[40] A. R. S. Parmezan, V. M. Souza, G. E. Batista, Towards hierarchical classification of data streams, in: Iberoamerican Congress on Pattern Recognition, Springer, 2018, pp. 314–322.

[41] J. Metz, A.A. Freitas, M.C. Monard, E.A. Cherman, A study on the selection of local training sets for hierarchical classification tasks, Encontro Nacional de Inteligncia Artif. (2011) 572–583.
[42] A. Clare, R.D. King, Predicting gene function in saccharomyces cerevisiae, Bioinformatics 19 (2) (2003) 42–49.
[43] M. Defferrard, K. Benzi, P. Vandergheynst, X. Bresson, FMA: A dataset for music analysis, in: Proceedings of The International Society for Music Information Retrieval Conference (ISMIR), Suzhou, China, 2017, pp. 316–323.
[44] I. Dimitrovski, D. Kocev, S. Loskovska, S. Džeroski, Hierarchical classification of diatom images using ensembles of predictive clustering trees, Ecol. Inf. 7 (1) (2012) 19–29.
[45] F. Charte, A. Rivera, M.J. del Jesus, F. Herrera, On the impact of dataset complexity and sampling strategy in multilabel classifiers performance, in: International Conference on Hybrid Artificial Intelligence Systems, Springer, 2016, pp. 500–511.
[46] J. Wehrmann, R. Cerri, R. Barros, Hierarchical multi-label classification networks, in: Proceedings of the International Conference on Machine Learning, 2018, pp. 5225–5234.
[47] G.T. Pereira, P.H. Gabriel, R. Cerri, Hierarchical classification of transposable elements with a weighted genetic algorithm, in: Proceedings of the EPIA Conference on Artificial Intelligence, Springer, 2019, pp. 737–749.
[48] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J. Mach. Learn. Res. 13 (Feb) (2012) 281–305.
[49] M. Buda, A. Maki, M.A. Mazurowski, A systematic study of the class imbalance problem in convolutional neural networks, Neural Networks 106 (2018) 249–259.
[50] R. Mohammed, J. Rawashdeh, M. Abdullah, Machine learning with oversampling and undersampling techniques: Overview study and experimental results, in: 2020 11th International Conference on Information and Communication Systems (ICICS), IEEE, 2020, pp. 243–248.