Adapting Non-Hierarchical Multilabel Classification Methods for Hierarchical Multilabel Classification

Ricardo Cerri[*], André Carlos P. L. F. de Carvalho

Institute of Mathematical Sciences and Computation

University of São Paulo at São Carlos - Brazil

Alex A. Freitas

School of Computing

University of Kent at Canterbury - UK

Abstract

In most classification problems, a classifier assigns a single class to each instance and the classes form a flat (non-hierarchical) structure, without superclasses or subclasses. In hierarchical multilabel classification problems, the classes are hierarchically structured, with superclasses and subclasses, and instances can be simultaneously assigned to two or more classes at the same hierarchical level. This article proposes two new hierarchical multilabel classification methods based on the well-known local approach for hierarchical classification. The methods are compared with two global methods and one well-known local binary classification method from the literature. The proposed methods presented promising results in experiments performed with bioinformatics datasets.

Keywords: hierarchical classification, multilabel classification, bioinformatics

[*] Ricardo Cerri is with the Department of Computer Sciences, University of São Paulo at São Carlos, Av. Trabalhador São-carlense, 400, Centro, P.O.Box: 668, CEP: 13560-970, São Carlos, SP, Brazil (phone: +55 16 3373-8161; email: cerri@icmc.usp.br).

1 Introduction

In traditional classification problems, each instance from a dataset is associated with just one out of two or more classes. In hierarchical multilabel classification (HMC), the classification task is more complex, since the classes are hierarchically structured and an instance can simultaneously belong to more than one class at the same hierarchical level. These problems are very common in applications like protein and gene function prediction and text classification.

HMC problems can be treated using two major approaches, referred to as local (or top-down) hierarchical classification, and global (or one-shot) hierarchical classification. In the local approach, traditional classification algorithms are trained to produce a hierarchy of classifiers, which typically are applied in a top-down fashion to classify each new (test) instance. In contrast, the global approach induces a single classification model considering the class hierarchy as a whole.

According to [39], there are different training versions for the local approach: a local classifier per node, a local classifier per parent node, and a local classifier per level. These versions use a similar local procedure to predict the class of a new instance. The experiments presented in this work use only the local classifier per parent node version. In this version, for each parent node in the class hierarchy, a classifier is induced considering the classes of its child nodes. When this classifier has to deal with more than two classes, a multiclass classification approach must be used.

For the classification of a new instance, the system initially predicts its most generic class, which is a class node at the first level in the hierarchy. The predicted class is used to reduce the set of possible child classes at the next level, defining its subclasses. Thus, in the test phase, when an instance is assigned to a class that is not a leaf node, it is further classified into one or more subclasses of this class. A disadvantage of this approach is that, as the tree is traversed toward the leaves, classification errors are propagated to the deeper levels of the class hierarchy. However, it has the positive aspect that any traditional (non-hierarchical) classification algorithm can be used at each node of the class hierarchy.

In the global approach, after inducing a single classification model using the whole training set, the classification of a new instance occurs in just one step. Since global methods must consider the peculiarities of hierarchical classification, traditional classification algorithms can-not be used, unless adaptations are made to consider the whole class hierarchy. As a result, global methods have a more complex implementation. However, they avoid the error propagation problem associated with the local approach. In addition, if the global approach is used to generate a set of classification rules, the induced global rule set tends to be less complex (with much fewer rules) than the collection of all local rule sets generated by classifiers following the local approach [2, 47].

This article proposes and evaluates two local methods, named HMC-Label-Powerset and HMC-Cross-Training. These new methods are hierarchical variations of non-hierarchical multilabel methods found in the literature: Label-Powerset [44] and Cross-Training [38]. The HMC-Label-Powerset method uses a label combination strategy to combine sibling classes assigned to an instance into a new class, transforming the original HMC problem into a hierarchical single-label problem. The HMC-Cross-Training method applies a label decomposition strategy, which transforms the original HMC problem into a set of hierarchical single-label problems. In spite of the transformations performed, in the end, both methods produce solutions to the original hierarchical multilabel problem. Thus, the main difference between the proposed methods is:

- HMC-Label-Powerset: based on local label combination, where the set of labels assigned to an instance is combined into a new class;

- HMC-Cross-Training: based on local label decomposition, where the HMC problem is decomposed into a set of single-label problems.

In this work, these two methods are compared with three well-known HMC methods: the well-known local binary-relevance method (HMC-Binary-Relevance) [39], used as baseline in many works, and two global methods, HC4.5 [11] and Clus-HMC [47]. The main aspects of these three methods are:

- HMC-Binary-Relevance: local method based on local binary classification, where a classifier is associated with each class and trained to solve a binary classification task;

- HC4.5: global hierarchical multilabel variation of the C4.5 algorithm [34], where the entropy formula is modified to cope with HMC problems;

- Clus-HMC: global method based on the concept of Predictive Clustering Trees (PCTs) [6], where a decision tree is structured as a cluster hierarchy.

An experimental comparison between the global and local approaches is another contribution of this work, since the analysis of the results can lead to the improvement of existing hierarchical classification methods and to the development of new methods. Besides, there are few empirical comparisons of these two approaches in the literature [14, 47], since most of the works compare a proposed method with a non-hierarchical counterpart.

The five methods investigated are evaluated using 10 datasets related with gene function prediction for the *Saccharomyces cerevisiae* (a specific type of *Yeast*) model organism, regarding different data conformations. Specific metrics developed for the evaluation of HMC classifiers are used. The experimental results show that the proposed methods, specially the HMC-Label-Powerset method, can provide a good alternative to deal with HMC problems.

This article is organized as follows: Section 2 introduces the basic concepts of hierarchical and multilabel classification; Section 3 has a brief review of recent HMC works found in the literature; the proposed methods are explained in details in Section 4; the experimental setup and the analysis of the results are presented in Sections 5 and 6, respectively; finally, Section 7 discusses the main conclusions and future research directions.

2 Hierarchical and Multilabel Classification

This section briefly introduces hierarchical and multilabel classification problems. For such, it starts with hierarchical single-label problems followed by non-hierarchical multilabel problems. These types of problems are then combined in the discussion of hierarchical multilabel classification problems, which are formally defined.

2.1 Hierarchical Single-Label Classification

In most of the classification problems described in the literature, a classifier assigns a single class to each instance x_i and the classes form a non-hierarchical structure, with no consideration of superclasses or subclasses. However, in many real classification problems, one or more classes can be divided into subclasses or grouped into superclasses. In this case, the classes follow a hierarchical structure, usually a tree or a Directed Acyclic Graph (DAG). These problems are known in the literature of Machine Learning (ML) as hierarchical classification problems. In these problems, new instances are classified into the class(es) associated with one or more nodes in a class hierarchy. When each new instance must be assigned to a leaf node of the class hierarchy, the hierarchical classification task is named *mandatory leaf node classification.* When the most specific class assigned to an instance can be an internal (non-leaf) node of the class hierarchy, the task is named *non-mandatory leaf node classification* [22].

Figure 1 illustrates examples of two text classification problems, one whose classes are structured as a tree and the other as a DAG. The main difference between the tree 1(a) and the DAG 1(b) structures is that, in the tree structure, each node (except the root) has exactly one parent node, while, in the DAG structure, each node can have more than one parent node. In tree structures, each class has a unique depth, because there is only one path from the root to any given class node. In DAG structures, however, the depth of a class is no longer unique, since there can be more than one path between the root and a given class node. These characteristics must be taken into account in the design and evaluation of classification models based on these structures.

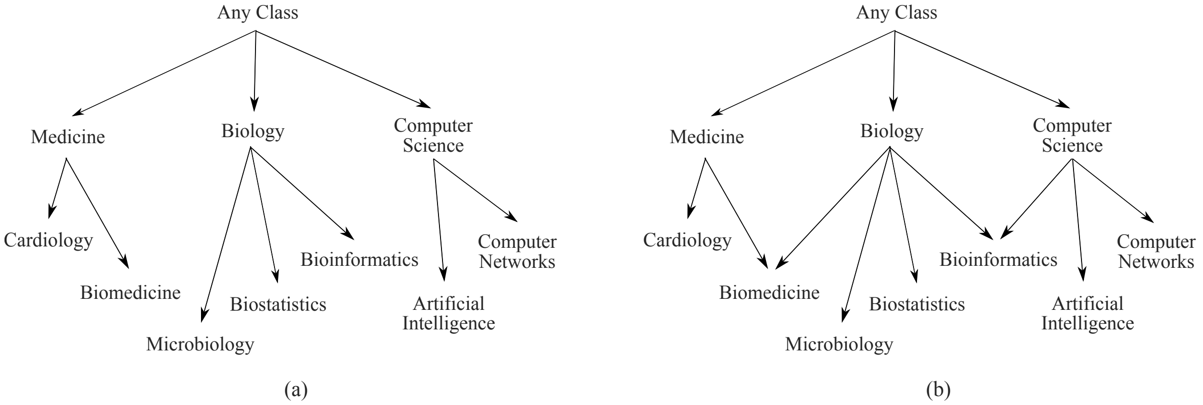


Fig. 1: Classification problems of scientific reports: (a) hierarchy structured as a tree; (b) hierarchy structured as a DAG.

In both structures, classes associated with deeper nodes in the hierarchy usually have lower prediction accuracy. This occurs because these classes are more specific and the classifiers for these nodes are trained with fewer instances than classes associated with shallower nodes.

2.2 Non-Hierarchical Multilabel Classification

In multilabel classification problems, each instance x_i can be associated to two or more classes at the same time. A multilabel classifier can be represented by a function $H : X \rightarrow 2^C$, which maps an instance $x_i \in X$ (space of instances) into a set of classes $C_i \in C$ (space of classes).

Figure 2 illustrates a comparison between a conventional classification problem, where instances can be assigned to only one class, and a multilabel classification problem. Figure 2(a) represents a classification problem in which a document can belong to one of the two classes "Biology" or "Computer Science", but never to both classes at the same time. Figure 2(b) shows a classification problem in which a document can be simultaneously assigned to the classes "Biology" and "Computer Science", referred to as "Bioinformatics" documents.

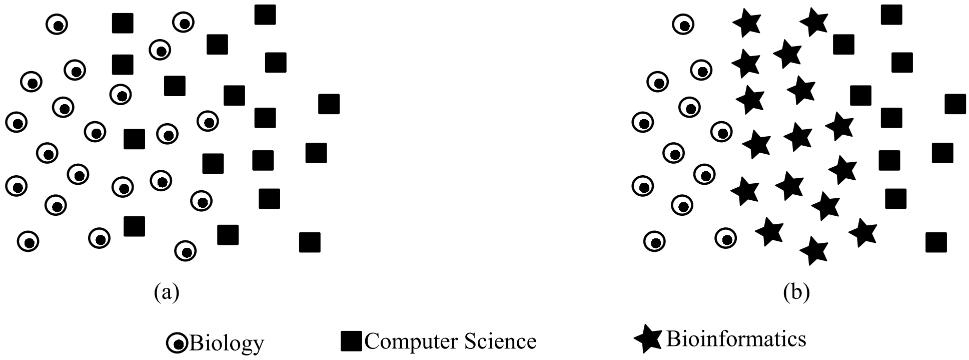


Fig. 2: Examples of classification problems: (a) traditional single-label classification; (b) multilabel classification.

Similar to hierarchical problems, where local and global approaches can be used to solve the classification task, two major approaches have been used in multilabel problems, referred to as algorithm independent and algorithm dependent [9]. The algorithm independent approach transforms the original multilabel problem into a set of single-label problems and, as in the local approach, any traditional classification algorithm can be used. In the algorithm dependent approach, as the name suggests, new algorithms are developed specifically for multilabel problems, or modifications are made in the internal mechanisms of existing traditional algorithms. The global approach for hierarchical problems can be seen as an algorithm dependent approach, as new or modified algorithms are used.

As stated by [7], it is worth noting that multilabel classification is different from fuzzy classification. Fuzzy classification is used to deal with ambiguity between multiple classes for a given instance. It is not used to achieve a multilabel classification. Usually, a defuzzification step is used to derive a crisp classification decision. The multilabel classification, on the other hand, is a problem where an instance can have properties of multiple classes, and these classes can be very distinct. Additionally, the use of membership functions in both problems is different. While, in fuzzy systems, for each instance, the sum of the degrees of memberships in all "fuzzy classes" is 1, this constraint in not applied to multilabel problems, where each instance can be assigned to more than one class, belonging 100% to each class, which would be equivalent to a "total degree of membership" much larger than 1.

2.3 Hierarchical Multilabel Classification

In hierarchical multilabel classification problems, the multilabel and hierarchical characteristics are combined, and an instance can be assigned to two or more subtrees of the class hierarchy. The HMC problem is formally defined by [47] as follows:

Given:

- a space of instances X;
- a class hierarchy $(C, \leq_h)$, where C is a set of classes and $\leq_h$ is a partial order representing the superclass relationship (for all $c_1, c_2 \in C : c_1 \leq_h c_2$ if and only if c_1 is a superclass of c_2);
- a set T of instances (x_i, C_i) with $x_i \in X$ and $C_i \subseteq C$, such that $c \in C_i \Rightarrow \forall c' \leq_h c : c' \in C_i$;
- a quality criterion q that rewards models with high accuracy and low complexity.

Find:

- a function $f : X \rightarrow 2^C$, where 2^C is the powerset of C, such that $c \in f(x) \Rightarrow \forall c' \leq_h c : c' \in f(x)$ and f optimizes q.

The quality criterion q can be the mean accuracy of the predicted classes or the distances between the predicted and real classes in the class tree. It can also consider that misclassifications in levels close to the root node are worse than misclassifications in deeper levels. Besides, the complexity of the classifiers and the induction time can be taken into account as quality criteria.

An example of HMC problem is illustrated in Figure 3, in which the class hierarchy is structured as a tree. In this example, a scientific article can be classified in the classes "Biology/Biostatistics", "Biology/Bioinformatics", and "Computer Science/Artificial Intelligence". When a prediction is made in the internal nodes of the tree, it generates a subtree. In the case of hierarchical single-label classification, this subtree is reduced to a path.

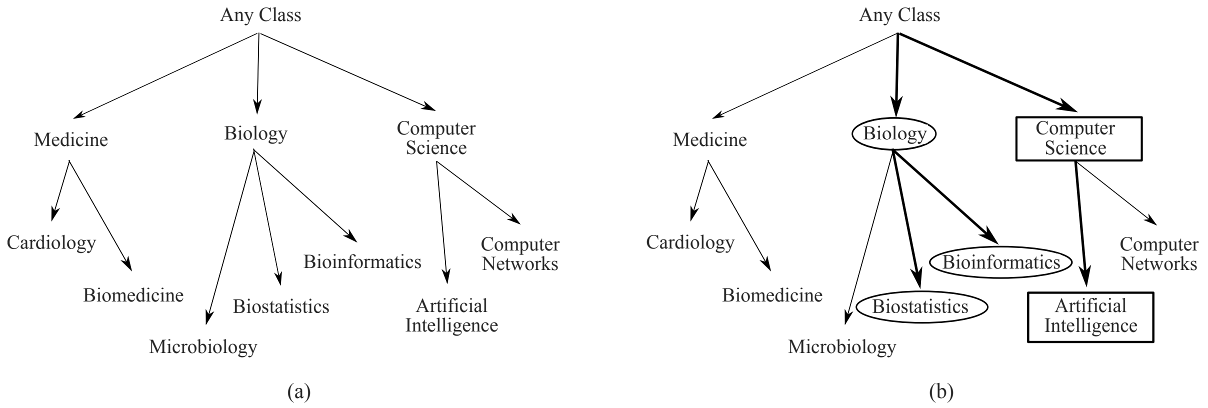


Fig. 3: HMC problem structured as a tree: (a) class hierarchy; (b) predictions generating a subtree.

It is important to notice the difference between hierarchical single-label problems and multilabel problems. A hierarchical single-label problem can be seen as being naturally multilabel in a kind of trivial way, due the fact that a path in the hierarchy has more than one class. When the class "Biology/Bioinformatics" is assigned to an instance, this prediction means that the instance belongs to two classes: "Biology" and "Bioinformatics". However, in this paper, a hierarchical problem is considered multilabel only in the non-trivial case where classes from more than one path in the hierarchy are assigned to an instance.

3 Related work

Many methods have been proposed in the literature to deal with HMC problems. The majority of them are applied to protein and gene function prediction and text classification. This section reviews some of the recent methods, organizing them according to the taxonomy proposed by [39]. In this taxonomy, a hierarchical classification algorithm is described by a 4-tuple $< \Delta, \Xi, \Omega, \Theta >$, where:

- Δ: indicates if the algorithm is hierarchical single-label (SPP - Single Path Prediction) or hierarchical multilabel (MPP - Multiple Path Prediction);

- Ξ: indicates the prediction depth of the algorithm - MLNP (Mandatory Leaf-Node Prediction) or NMLNP (Non-Mandatory leaf-node prediction);

- Ω: indicates the taxonomy structure the algorithm can handle - T (tree structure) or D (DAG structure);

- Θ: indicates the categorization of the algorithm under the proposed taxonomy - LCN (Local Classifier per Node), LCL (Local Classifier per Level), LCPN (Local Classifier per Parent Node) or GC (Global Classifier).

A LCN method was proposed in [4], where a hierarchy of SVM classifiers [46] is used for the classification of gene functions according to the biological process hierarchy of the GO (Gene Ontology) [3]. Classifiers are trained for each class separately. The predictions are combined using a Bayesian network model, with the objective of finding the most probable consistent set of predictions.

Another LCN method, [10], proposed a Bayes-optimal classifier and applied it to two document datasets: the Reuters Corpus Volume 1, RCV1 [29] and a specific subtree of the OHSUMED corpus of medical abstracts [25]. In this method, the relationships between the classes in the hierarchy are seen as a forest. Trees in this forest represent a class taxonomy G. The method starts by putting all nodes of G in a set S. The nodes are then removed from S one by one. Every time an instance is assigned to a class c_i, it is also assigned to the classes in the path from the root of its tree to the class c_i.

Also based on the LCN strategy, an ensemble where each base classifier is associated with a class of the hierarchy was investigated in [45]. It was applied to datasets with genes annotated according to the FunCat scheme developed by MIPS [30]. The base classifiers were trained to become specialized on the classification of one class in the hierarchy. For such, each trained classifier estimates the local probabilities $\hat{p}_i(x)$, that a given instance x belongs to a class c_i. The ensemble phase estimates the "consensus" global probability $p_i(x)$.

A local strategy that initially trains a classifier for the first hierarchical level is proposed in [27]. The dataset used was composed by MedLine articles associated with GO codes. The training process follows a $LCPN$ strategy, and, during the top-down classification process, each classifier outputs a real value, representing the probability that the input instance belongs to a class c_i. Only the classes whose probability is higher than a given threshold are assigned to the instance.

A second LCPN method was proposed by [20]. This method is a hierarchical variation of the AdaBoost [37], named TreeBoost.MH. It was applied to the hierarchical multilabel classification of documents, using a hierarchical version of the Reuters-21578 corpus, generated in

[42], and the Reuters Corpus Volume 1 - version 2 (RCV1-v2) [29]. The TreeBoost.MH is a recursive algorithm that uses the AdaBoost.MH as its base step, and is called recursively over the hierarchical class structure.

There also works proposing GC methods. The work of [36], for example, investigated a kernel based algorithm for the hierarchical multilabel classification. This method was applied to text and biological classification problems (RCV1, WIPO-alpha patent dataset [48] and Enzyme classification dataset [32]). The classification method is based on a variation of a framework named Maximum Margin Markov Network [41, 43], where the hierarchical structure is represented as a Markov tree. For the learning process, the authors defined a joint feature map $\phi(x, y)$ over the input and output spaces. In the HMC context, the output space can be defined as all possible subtrees or subgraphs of the class hierarchy.

Another GC method was proposed in [11]. This global method, named HC4.5, is based on decision tree induction. It was applied to the classification of gene functions of the *Saccharomyces cerevisiae* organism. It uses a variation of the C4.5 algorithm, in which modifications in the use of the class entropy are made. In the original C4.5 algorithm, the entropy is used to decide the best split in the decision tree. The authors' variation of C4.5 employed the sum of the entropies of all classes to choose the best attribute to label an internal node of the tree.

The GC strategy was also used in the work of [8], where two new HMC methods were proposed and applied to the task of document classification, also using the WIPO-alpha patent dataset. The first method is a generalized version of the Perceptron algorithm, and the second is a hierarchical multilabel SVM. For the multilabel version of the SVM, the authors generalized the multiclass formulation [15] to a multilabel formulation similar to [19]. The proposed hierarchical Perceptron algorithm uses the minimum-overlap (Minover) learning rule [28], so that the instance that most violates the desired margin is used to update the separating hyperplane.

The work of [2] also followed the GC strategy to developed an Artificial Immune System (AIS), named Multilabel Hierarchical Classification with an Artificial Immune System (MHC-AIS), for the prediction of protein functions described in the GO. The proposed algorithm is able to find a set of rules that are both hierarchical and multilabel, so that a single classification rule can assign more than one class to a given protein (instance). The algorithm training is divided into two basic procedures, named Sequential Covering (SC) and Rule Evolution (RE).

These procedures produce candidate classification rules, each rule composed of two parts: an antecedent (IF part), represented by a vector of attribute-value conditions, and a consequent (THEN part), represented by a set of predicted classes.

Still based on the GC approach, the work of [47] compared three methods that use decision trees, based on PCTs [6], for HMC problems. The methods are compared using datasets related to functional genomics. The authors compared the performance of the Clus-HMC method that induces a single decision tree making predictions for all the classes of the hierarchy at once, with other two methods that induce a decision tree for each hierarchical class, named Clus-SC and Clus-HSC. Clus-SC defines an independent single-label hierarchical classification task for each class, ignoring the hierarchical relationships between the classes. Clus-HSC explores the hierarchical relationships to induce a decision tree for each class in the hierarchy. The authors also applied the methods to class hierarchies structured as DAGs, discussing the issues that arise when dealing with these kinds of structures and the modifications required to the algorithms to be able to deal with such hierarchies.

Table 1 presents the methods reviewed in this section, organized according to the taxonomy presented by [39].

Table 1: Detailed categorization of the algorithms according to the taxonomy proposed by [39].

$< \Delta, \Xi, \Omega, \Theta >$	List of Works
$< MPP, NMLNP, D, LCN >$	[4]
$< MPP, NMLNP, T, LCN >$	[10, 45]
$< MPP, NMLNP, D, LCPN >$	[27]
$< MPP, MLNP, T, LCPN >$	[20]
$< MPP, NMLNP, T, GC >$	[36, 11]
$< MPP, NMLNP, D, GC >$	[8, 2, 47]

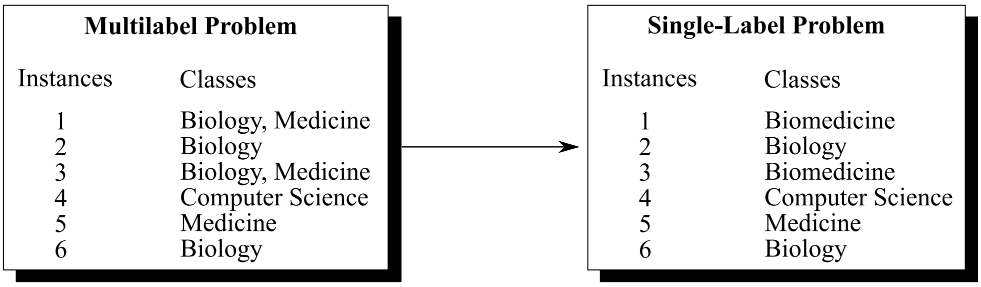
4 Proposed Methods

This section presents the HMC methods proposed in this work. The first method, named Hierarchical Multilabel Classification with Label-Powerset (HMC-LP), performs a combination of labels (classes), where all labels assigned to an instance, at a specific level, are combined into a new and unique label. This creates a hierarchical single-label problem. The second method, named Hierarchical Multilabel Classification with Cross-Training (HMC-CT), carries out a decomposition of labels, creating many hierarchical single-label problems. According to the taxonomy proposed by [39], these new methods can be classified as $< MPP, MLNP, T, LCPN >$.

4.1 Hierarchical Multilabel Classification with Label-Powerset (HMC-LP)

The HMC-LP method uses a label combination process to transform the HMC problem into a hierarchical single-label problem. It is a hierarchical adaptation of the Label-Powerset method used for non-hierarchical multilabel classification in [7] and [44]. Different from the HMC-Binary-Relevance (HMC-BR) method, HMC-LP considers the sibling relationships between classes.

In the Label-Powerset method used for multilabel non-hierarchical classification, all classes assigned to each instance are combined into a new and unique class. Figure 4 illustrates an example of the application of the label combination process. This figure illustrates a combination of the classes in the instances 1 and 3. For each of these two instances, a new class was created, labeled "Biomedicine".



Multilabel Problem

Instances	Classes
1	Biology, Medicine
2	Biology
3	Biology, Medicine
4	Computer Science
5	Medicine
6	Biology

Single-Label Problem

Instances	Classes
1	Biomedicine
2	Biology
3	Biomedicine
4	Computer Science
5	Medicine
6	Biology

Fig. 4: Label combination process of the Label-Powerset method.

In the HMC-LP method, labels are combined at each level of the class hierarchy. This occurs by combining all classes assigned to each instance, at a specific level, creating a new class.

To illustrate this process, consider an instance belonging to the classes $A.D$ and $A.F$, and another instance belonging to the classes $E.G$, $E.H$, $I.J$ and $I.K$, where $A.D$, $A.F$, $E.G$, $E.H$, $I.J$ and $I.K$ are hierarchical structures, such that $A \leq_h D$, $A \leq_h F$, $E \leq_h G$, $E \leq_h H$, $I \leq_h J$ and $I \leq_h K$ with A, E and I belonging to the first level and D, F, G, H, J and K belonging to the second level, as shown in Figure 5. When the HMC-LP method is applied, the resulting combination of classes for the two instances would be a new hierarchical structure with the label paths $C_A.C_{DF}$ and $C_{EI}.C_{GHJK}$, respectively. In the first instance, C_{DF} is a new label formed by the combination of the labels D and F at the second level. In the second instance, C_{GHJK} is a label formed by the combination of the labels G, H, J and K at the second level. Figure 5 illustrates this process of label combination. To the best of our knowledge, this type of adaptation was not yet reported in the literature.

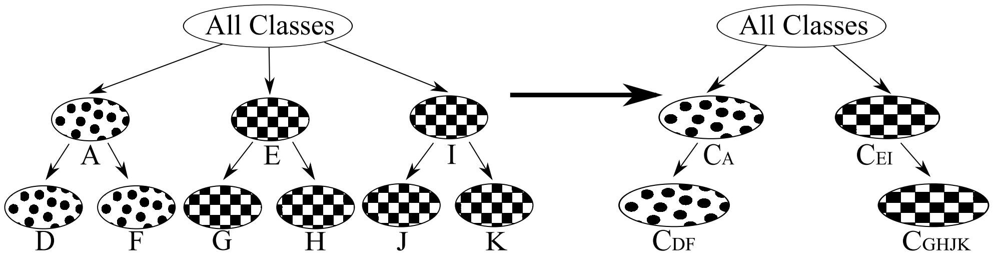


Fig. 5: Label combination process of the HMC-Label-Powerset method.

As can be seen in Figure 5, after the label combination process, the original problem is transformed into a hierarchical single-label problem. During the training and test phases, the local approach is used, with one or more multiclass classifiers at each internal hierarchical level. At the end of the classification process, the predictions referring to the combined classes are transformed into predictions of their original, individual classes. The label combination procedure is presented in Algorithm 1.

Algorithm 1: Label combination procedure of the HMC-LP method.

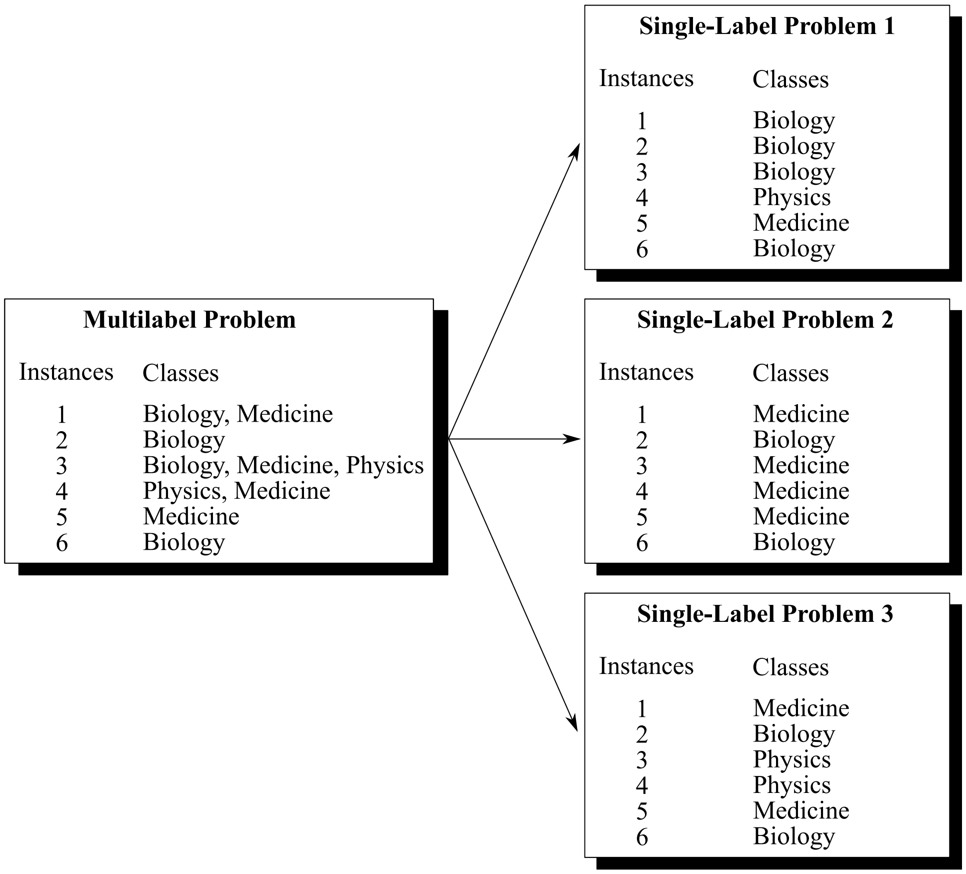
```
Procedure LabelCombination($X, C$)
Input: set of instances $X$, set of classes $C$
Output: $NewClasses$
foreach level $j$ of the class hierarchy do
    foreach subset $C_i$ of the set $C$, assigned to an instance $x_i$ in level $j$ do
        Get a new class $c_{i,j}$ for the instance $x_i$ from $C_i$
        $NewClasses_{i,j} \leftarrow c_{i,j}$
return $NewClasses$
```

One problem with the label combination process is that it can considerably increase the number of classes in the dataset. As a small example, Figure 4 shows a multilabel problem with three classes. After the label combination procedure, the number of classes is increased to four. If there are many possible multilabel combinations in the dataset, the new formed classes may have few positive instances, resulting in sparse training data. Despite this disadvantage, if multiclass classifiers are used at each internal node, instead of binary classifiers, the induction time might decrease considerably when compared with the HMC-Binary-Relevance method.

4.2 Hierarchical Multilabel Classification with Cross-Training (HMC-CT)

The HMC-CT method uses a label decomposition process to modify the original hierarchical multilabel problem, transforming it into a set of hierarchical single-label problems. In the decomposition process, if the maximum number of labels per instance is N, the original problem is decomposed into N single-label problems. For each instance, each possible class is considered the positive class in turn. Thus, multilabel instances participate more than once in the training process. As an example, if a dataset has multilabel instances belonging to the classes c_A, c_B and c_C, when a classifier for the class c_A is trained, each multilabel instances that has the class c_A as one of its classes becomes a single-label instance for the class c_A. The same procedure is adopted for the classes c_B and c_C. The method, named Cross-Training, was originally proposed by [38] for non-hierarchical multilabel classification.

The label decomposition process of the Cross-Training method for non-hierarchical classification is illustrated in Figure 6. It is possible to see in the figure that when a classifier is trained for the class “Biology”, all multilabel instances that belong to the class “Biology” become single-label instances for that class, and the same procedure is adopted for the other classes.



Multilabel Problem

Instances	Classes
1	Biology, Medicine
2	Biology
3	Biology, Medicine, Physics
4	Physics, Medicine
5	Medicine
6	Biology

Single-Label Problem 1

Instances	Classes
1	Biology
2	Biology
3	Biology
4	Physics
5	Medicine
6	Biology

Single-Label Problem 2

Instances	Classes
1	Medicine
2	Biology
3	Medicine
4	Medicine
5	Medicine
6	Biology

Single-Label Problem 3

Instances	Classes
1	Medicine
2	Biology
3	Physics
4	Physics
5	Medicine
6	Biology

Fig. 6: Label decomposition process of the Cross-Training method.

When using multiclass classifiers, the number of classifiers used in the Cross-Training method is equal to the number of classes assigned to, at least, one multilabel instance. When using binary classifiers, however, the number of classifiers is equal to the number of classes in the problem. The method allows the original multilabel problem to be recovered from the single-label problems generated. In the example of Figure 6, three classifiers were used, because the three classes “Biology”, “Medicine” and “Physics” are assigned to a multilabel instance. It is important to notice in the figure that the method does not consider all possible cross-combinations of labels. As explained, the number of classifiers used is equal to the number of classes assigned to, at least, one multilabel instance. This method is named Additive Cross-Training. If all possible cross-combinations are considered, the method is named Multiplicative Cross-Training.

In the new hierarchical variation of the Cross-Training method proposed here, the label decomposition process is applied to all hierarchical levels, and the local approach is used during the test and training phases. Figure 7 illustrates a label decomposition process performed by the HMC-CT method. In this figure, when an instance belongs to more than one class, these classes are separated by a slash (/).

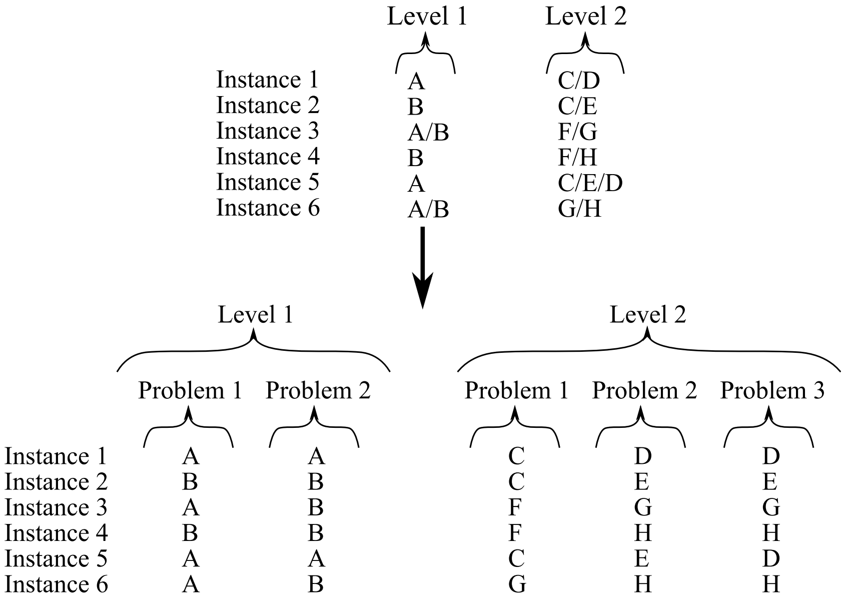


Fig. 7: Label decomposition process of the HMC-Cross-Training method.

It is important to notice the difference between the HMC-CT and HMC-BR methods. In the HMC-CT method, a classifier is not associated with each class. Thus, it does not transform the original problem into a binary problem. Instead, because all classes participate in the training process, it uses multiclass classifiers. Therefore, given a multilabel instance x_i, if x_i belongs to two classes, c_A and c_B, the training process occurs twice, first considering x_i as belonging to class c_A and later considering x_i as belonging to class c_B. Algorithm 2 shows the classification process of the HMC-CT method.

Algorithm 2: Classification process of the HMC-CT method.

```
Procedure Classify($x, Cl$)
Input: instance $x$, set of classifiers $Cl$
Output: $Classes$
$Classes \leftarrow \emptyset$
foreach classifier $cl_i$ from the set of classifiers $Cl$ do
    Predict a class $c_i$ for the instance $x$ using the classifier $cl_i$
    if not the last hierarchical level then
        Get the set $Cl_i$ of children classifiers of the classifier $cl_i$ trained with instances
        from class $c_i$
        $Classes \leftarrow Classes \cup \{c_i\} \cup$Classify($x, Cl_i$)
    else
        $Classes \leftarrow Classes \cup \{c_i\}$
return $Classes$
```

A problem with this method is its computational cost, which is higher than the cost for the

HMC-BR and HMC-LP methods. It is higher because the training process uses each instance several times, since it considers all its classes.

5 Experimental Setup

5.1 Datasets

The datasets used in the experiments are related to gene functions of the *Saccharomyces cerevisiae* (a specific type of *Yeast*), often used in the fermentation of sugar for the production of ethanol. This organism is also used in the fermentation of wheat and barley for the production of alcoholic beverages. It is one of the biology's classic model organisms, and has been the subject of intensive study for many years [47].

The gene functions (classes to be predicted) in these datasets are structured as a tree following the FunCat annotation scheme (http://mips.gsf.de/projects/funcat) developed by MIPS [30]. These datasets are freely available at http://www.cs.kuleuven.be/~dtai/clus/hmcdatasets.html. The FunCat scheme has 28 main categories for the functions, including cellular transport, metabolism and cellular communication. Its tree structure has up to six levels and a total of 1632 functional classes. To reduce computational cost, for each dataset, four classes of the FunCat scheme (01, 02, 10, and 11) were randomly selected at the first level. These classes and all its descendant classes up to the fourth class level were used in the experiments. Tables 2 and 3 show the main characteristics of the reduced datasets.

Table 2: Characteristics of the datasets.

Dataset	Num. Atrib.	Num. Instances		Avg. Num. Instances per Class				Avg. Num. Classes per Instance			
		Total	Multilabel	L1	L2	L3	L4	L1	L2	L3	L4
Expr	551	2444	1451	611.0	111.1	29.7	17.4	1.3	1.6	1.4	0.9
CellCycle	77	2445	1451	611.2	111.1	29.7	17.4	1.3	1.6	1.4	0.9
Church	27	2441	1449	610.2	110.9	29.7	17.4	1.3	1.6	1.4	0.9
Derisi	63	2438	1449	609.5	110.8	29.6	17.3	1.3	1.6	1.4	0.9
Eisen	79	1579	988	394.7	71.7	21.3	13.1	1.3	1.7	1.5	1.0
Gasch1	173	2444	1450	611.0	111.0	29.7	17.4	1.3	1.6	1.4	0.9
Gasch2	52	2454	1456	613.5	111.5	29.8	17.4	1.3	1.6	1.4	0.9
Phenotype	69	1059	634	264.7	48.1	13.6	8.1	1.4	1.7	1.4	0.9
Sequence	478	2480	1477	620.0	112.7	30.2	17.6	1.3	1.6	1.4	0.9
SPO	80	2419	1439	604.7	109.9	29.4	17.2	1.3	1.6	1.4	0.9

Table 3: Number of classes per level for each dataset.

Dataset	Number of Classes			
	Level 1	Level 2	Level 3	Level 4
Expr	4	22	70	84
CellCycle	4	22	70	84
Church	4	22	70	84
Derisi	4	22	70	84
Eisen	4	22	66	78
Gasch1	4	22	70	84
Gasch2	4	22	70	84
Phenotype	4	22	66	76
Sequence	4	22	70	84
SPO	4	22	70	84

5.2 Evaluation of the Classification Methods

For the experimental evaluation, the real and predicted sets of classes are represented as boolean vectors, where each position represents a class in the dataset. If an instance belongs to a class c_i, the i^{th} position of the vector that represents the real set of classes receives the value 1. The same representation is used for the predicted set of classes.

The datasets were divided using k-fold cross-validation, with $k = 5$. Statistical tests were applied to verify the statistical significance of the results with a confidence level of 95%. The tests employed were Friedman [23] and Nemenyi [31], which are recommended for comparisons involving many datasets and several classifiers [16].

The evaluation was carried out level by level in the classification hierarchy. For each hierarchical level, a value resulting from the evaluation of the predictive accuracy at that level is reported. The metrics used were those proposed by [40], named Hierarchical Micro Precision and Recall. A combination of these metrics, the Hierarchical-F_β metric, with $\beta = 1$, was also used. The value $\beta = 1$ was chosen so that the Precision and Recall metrics have the same importance in the calculation of the Hierarchical-F_β metric.

These metrics are calculated by computing, for each class, the contribution of the instances erroneously assigned to the class. For such, it is necessary to define an acceptable distance (Dis_θ) between two classes, which must be higher than zero. Equations (1) and (2) define the contribution of an instance x_j to a class c_i, where $x_j.agd$ and $x_j.lbd$ are, respectively, the predicted and the real classes of x_j, and $Dis(c', c_i)$ is the distance between two classes in the hierarchy, which is given by the number of edges between these two classes.

- If x_j is a False Positive:

$$Con(x_j, c_i) = \sum_{c' \in x_j.lbd} (1.0 - \frac{Dis(c', c_i)}{Dis_\theta}) \tag{1}$$

- If x_j is a False Negative:

$$Con(x_j, c_i) = \sum_{c' \in x_j.agd} (1.0 - \frac{Dis(c', c_i)}{Dis_\theta}) \tag{2}$$

The contribution of an instance x_j is therefore restricted to the interval $[-1, 1]$. This refinement, denoted by $RCon(x_j, c_i)$, is defined by Equation (3).

$$RCon(x_j, c_i) = min(1, max(-1, Con(x_j, c_i))) \tag{3}$$

The total contribution of False Positives (FP) ($FpCon_i$) and False Negatives (FN) ($FnCon_i$), for all instances, is defined by Equations (4) and (5), respectively.

$$FpCon_i = \sum_{x_j \in FP_i} RCon(x_j, c_i) \tag{4}$$

$$FnCon_i = \sum_{x_j \in FN_i} RCon(x_j, c_i) \tag{5}$$

After calculating the contributions of each instance, Equations (6) and (7) define the values of the Hierarchical Precision and Recall for each class, respectively.

$$Pr_i^{CD} = \frac{max(0, |TP_i| + FpCon_i + FnCon_i)}{|TP_i| + |FP_i| + FnCon_i} \tag{6}$$

$$Re_i^{CD} = \frac{max(0, |TP_i| + FpCon_i + FnCon_i)}{|TP_i| + |FN_i| + FpCon_i} \tag{7}$$

The extended values of Hierarchical Precision and Recall (Hierarchical Micro Precision and Recall) are presented in Equations (8) and (9), respectively, where m represents the number of classes.

$$\hat{Pr}^{\mu CD} = \frac{\sum_{i=1}^{m}(max(0, |TP_i| + FpCon_i + FnCon_i))}{\sum_{i=1}^{m}(|TP_i| + |FP_i| + FnCon_i)} \tag{8}$$

$$\hat{Re}^{\mu CD} = \frac{\sum_{i=1}^{m}(max(0, |TP_i| + FpCon_i + FnCon_i))}{\sum_{i=1}^{m}(|TP_i| + |FN_i| + FpCon_i)} \tag{9}$$

According to the value of Dis_θ, the values of $FpCon_i$ and $FnCon_i$ can be negative. Therefore, a max function is applied to the numerators of the Equations (8) and (9) to make their values positive. As $FpCon_i \leq |FP_i|$, if $|TP_i| + |FP_i| + FnCon_i \leq 0$, the numerator $max(0, |TP_i| + FpCon_i + FnCon_i) = 0$. The $\hat{Pr}^{\mu CD}$ value can be considered zero in this case. The same rule is applied to the calculation of $\hat{Re}^{\mu CD}$ [40].

The Hierarchical Micro Precision and Recall metrics can then be combined in the Hierarchical-F_β metric (Equation (10)).

$$Hierarchical - F_\beta = \frac{(\beta^2 + 1) \times hP \times hR}{\beta^2 \times hP + hR} \tag{10}$$

In the experiments, the value $Dis_\theta = 2$ was chosen as the acceptable distance between two nodes for the calculation of the Micro Precision and Recall metrics. Thus, if the number of edges (in the class tree) between a predicted class and a real class for a given instance is equal to 2, it will not be counted as a false positive or false negative. On the other hand, if the number of edges between a predicted class and a real class is larger than 2, this distance is counted as a false positive or false negative. The value $Dis_\theta = 2$ was also used in the experiments reported by [40].

As the objective of the evaluation metric is to consider that closer classes in the hierarchy are more similar to each other, the use of $Dis_\theta = 2$ defines that when the distance between a predicted class and a real class is equal to 2, this error should not contribute negatively to the metric value, because the metric considers that these two classes are similar. When the distance is larger than 2, the error should contribute negatively to the value of the metric.

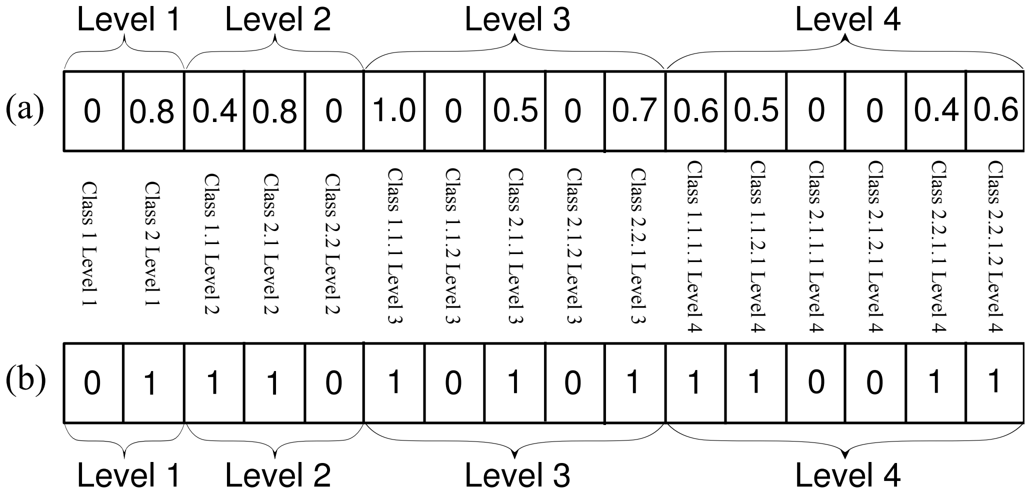
It should be noted that the metrics used in this work and in [40] artificially increase a little the values of Precision and Recall due to the fact that misclassifications involving similar true and predicted classes are not counted as errors. However, this is not a problem for the analysis of the results, since all algorithms being compared in our experiments have been evaluated according to the same metric.

5.3 Comparison with Other Local and Global Approaches

In the experiments carried out, the proposed local hierarchical multilabel methods were compared against the well-known HMC-Binary-Relevance method, based on the local approach, and the HC4.5 [11] and Clus-HMC [47] methods, based on the global approach. Five ML techniques were used as the base classifiers for the local methods: SVM [46], BayesNet [24], Ripper

[12], C4.5 [34] and KNN [1].

To compare the local and global hierarchical classification methods, modifications were needed in the vectors of predicted classes produced by the global methods. In these methods, the membership of an instance x_j to a given class c_i is given by a probability (real) value. Figure 8 shows an example of the vector of predicted classes obtained using the local and global methods. In the Figure 8(a), an instance x_j has a probability of 0.8 of belonging to the class c_2, a probability of 0.4 of belonging to the class c_3, and so on. To assign classes of the vector to an instance, a threshold value can be used. Thus, if a threshold value of 0.6 is chosen, only those classes with a value higher than or equal to 0.6 are assigned to the instance. Regarding Figure 8(a), these classes are c_2, c_4, c_6, c_{10}, c_{11} and c_{16}. Unlike the global methods, the vectors of predicted classes of the local methods contain only the values 0 and 1 (Figure 8(b)), indicating if an instance belongs (1) or does not belong (0) to a class.



	Class 1 Level 1	Class 2 Level 1	Class 1.1 Level 2	Class 2.1 Level 2	Class 2.2 Level 2	Class 1.1.1 Level 3	Class 1.1.2 Level 3	Class 2.1.1 Level 3	Class 2.1.2 Level 3	Class 2.2.1 Level 3	Class 1.1.1.1 Level 4	Class 1.1.2.1 Level 4	Class 2.1.1.1 Level 4	Class 2.1.2.1 Level 4	Class 2.2.1.1 Level 4	Class 2.2.1.2 Level 4
(a)	0	0.8	0.4	0.8	0	1.0	0	0.5	0	0.7	0.6	0.5	0	0	0.4	0.6
(b)	0	1	1	1	0	1	0	1	0	1	1	1	0	0	1	1

Fig. 8: Examples of predicted class vectors: (a) using the global methods; (b) using the local methods.

In this paper, five different threshold values were used in the evaluation of the global methods (0.0, 0.2, 0.4, 0.6 and 0.8), so that different performances could be obtained, i.e., different values of precision and recall. As the threshold value increases, the precision value tends to increase, and the recall value tends to decrease.

It is important to notice that there is no best threshold value. For example, suppose that after building a given decision tree (using the HC4.5 or Clus-HMC method), a given leaf node has 50 instances belonging to a class A and 50 instances belonging to a class B. There is no correct classification decision for this new instance, since it can belong to both classes A and

B, only to class *A*, or only to class *B*. There can also be a leaf node with 97 class *A* instances and 3 class *B* instances. In this case, instances from class *B* can represent noisy data or truly rare and important information. Thus, it is difficult to know if the correct decision is to classify a new instance that reaches this leaf in both classes *A* and *B*, in the class *A* with probability of 0.97 and in the class *B* with probability 0.03, or just in the class *A*. Due to these issues, different threshold values were used.

Since the main goal of this paper is to experimentally compare global and local approaches in a way as controlled as possible, the HC4.5 method used in the experiments was modified to work as described in [14]. This version of the algorithm includes the restriction of *mandatory leaf node prediction*, which is the prediction process used in this work.

5.4 Software Tools

The local methods used in this work were implemented using the R language [35], which has many ML-related packages. The e1071 [17] package was used to generate the SVM classifiers. The RWeka [26] package was used to induce the classifiers BayesNet, Ripper, C4.5 and KNN.

6 Experiments and Discussion

Table 4 presents the ranking of the algorithms, considering their average hierarchical f-measure values across the 10 datasets used. The table shows in each row the ranking obtained at a specific level of the class hierarchy. Bold numbers represent the algorithms that achieved the top three positions in the ranking.

Table 4: Ranking of the algorithms considering their average hierarchical f-measure values across all datasets. The top three positions of the rank are shown in bold face.

	Local															Global									
	HMC-BR					HMC-LP					HMC-CT					HC4.5					Clus-HMC				
	KNN	C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	0.0	0.2	0.4	0.6	0.8	0.0	0.2	0.4	0.6	0.8
Level 1	9	14	10	4	11	15	12	20	8	7	5	**3**	**1**	6	**2**	18	17	21	24	25	16	13	19	22	23
Level 2	10	15	16	**3**	7	5	**1**	24	14	4	12	23	21	19	8	18	13	9	25	22	11	6	**2**	17	20
Level 3	10	12	15	8	7	4	**1**	21	9	5	20	25	24	23	19	18	14	**3**	16	22	11	6	**2**	13	17
Level 4	9	10	15	8	7	4	**1**	18	12	5	22	25	24	23	21	19	14	**3**	16	20	13	6	**2**	11	17

According to Table 4, the HMC-CT method achieved the best overall performance at the first hierarchical level. At the second level, the best overall performance was obtained by the

HMC-LP method, followed by the Clus-HMC and the HMC-BR methods, in this order. At the third and fourth hierarchical levels, the first, second and third best overall performances were achieved by the HMC-LP, Clus-HMC and HC4.5 methods, respectively.

The accuracy results of the experimental comparisons of the local and global approaches, for each dataset, are shown in tables 5, 6, 7 and 8, each one for a different hierarchical level. The best results, for each dataset, are shown in bold face and the standard deviations are shown between parentheses.

Table 5: Comparison of local and global approaches at the first hierarchical level using the hierarchical f-measure metric.

Datasets	Classifier	Local			Threshold	Global	
		HMC-BR	HMC-LP	HMC-CT		C4.5H	Clus-HMC
Expr	KNN	53.45 (0.7)	52.37 (2.2)	57.01 (0.5)	0.0	50.05 (1.5)	51.73 (1.6)
	C4.5	53.86 (1.6)	53.00 (2.2)	59.94 (1.1)	0.2	50.13 (1.4)	51.72 (1.5)
	Ripper	56.21 (1.9)	49.81 (1.7)	61.56 (1.0)	0.4	48.80 (1.6)	50.70 (1.4)
	BayesNet	**61.92 (1.9)**	60.44 (2.3)	59.48 (2.9)	0.6	45.34 (2.1)	49.30 (1.3)
	SVM	49.66 (2.2)	51.28 (1.7)	56.50 (1.0)	0.8	41.82 (2.3)	48.41 (1.3)
CellCycle	KNN	54.99 (2.0)	53.54 (1.9)	58.00 (1.8)	0.0	47.51 (1.7)	47.13 (1.6)
	C4.5	51.64 (2.2)	50.00 (1.7)	58.32 (1.2)	0.2	47.45 (1.3)	47.12 (1.5)
	Ripper	51.64 (0.6)	47.65 (2.2)	59.08 (2.1)	0.4	45.31 (1.1)	46.45 (1.6)
	BayesNet	56.60 (2.6)	53.64 (2.0)	58.03 (1.4)	0.6	41.57 (1.6)	43.11 (0.9)
	SVM	57.45 (2.1)	58.50 (2.7)	**61.85 (1.6)**	0.8	38.42 (1.3)	42.00 (0.6)
Church	KNN	44.76 (1.9)	43.94 (0.6)	53.80 (0.9)	0.0	47.63 (1.2)	47.27 (1.2)
	C4.5	44.40 (2.0)	45.06 (0.4)	54.69 (1.5)	0.2	51.43 (1.1)	50.27 (0.9)
	Ripper	46.14 (1.6)	40.77 (0.7)	54.74 (2.0)	0.4	47.34 (1.8)	47.95 (1.9)
	BayesNet	55.32 (0.5)	44.48 (1.3)	49.16 (3.0)	0.6	28.35 (2.1)	30.55 (3.1)
	SVM	44.36 (1.8)	45.56 (1.2)	**55.85 (1.6)**	0.8	20.50 (1.6)	26.51 (3.3)
Derisi	KNN	47.89 (1.7)	46.23 (1.2)	52.37 (1.4)	0.0	45.51 (1.0)	46.23 (2.3)
	C4.5	40.59 (6.7)	46.11 (1.1)	55.33 (0.9)	0.2	45.26 (0.9)	46.22 (2.3)
	Ripper	47.15 (1.5)	42.76 (0.5)	**56.76 (1.6)**	0.4	42.54 (1.7)	44.72 (2.6)
	BayesNet	54.35 (1.5)	45.30 (3.5)	49.56 (2.0)	0.6	39.37 (1.5)	41.65 (2.8)
	SVM	47.48 (1.4)	46.77 (1.5)	54.76 (1.9)	0.8	34.44 (1.8)	39.79 (2.6)
Eisen	KNN	55.90 (2.1)	54.14 (1.3)	59.83 (2.2)	0.0	48.78 (1.7)	50.66 (1.5)
	C4.5	52.53 (1.5)	52.13 (1.2)	58.35 (2.5)	0.2	48.79 (1.7)	50.66 (1.4)
	Ripper	54.13 (2.1)	50.20 (1.9)	61.59 (1.9)	0.4	47.21 (2.3)	50.17 (1.5)
	BayesNet	60.40 (1.9)	54.18 (2.5)	59.85 (1.7)	0.6	43.26 (2.4)	46.48 (2.3)
	SVM	58.17 (2.4)	59.85 (1.2)	**62.02 (2.9)**	0.8	38.05 (3.1)	44.80 (2.1)
Gasch1	KNN	55.80 (2.7)	54.63 (1.5)	59.03 (1.5)	0.0	48.90 (1.9)	49.53 (1.6)
	C4.5	55.39 (1.7)	52.85 (1.5)	59.44 (1.6)	0.2	48.92 (1.8)	49.53 (1.5)
	Ripper	53.66 (2.4)	46.43 (1.0)	61.40 (1.7)	0.4	46.98 (2.0)	49.50 (1.1)
	BayesNet	60.30 (1.1)	54.99 (1.2)	53.19 (1.7)	0.6	43.52 (1.3)	46.81 (1.1)
	SVM	59.96 (2.1)	60.60 (1.7)	**63.44 (2.1)**	0.8	40.58 (1.2)	45.76 (1.0)
Gasch2	KNN	52.97 (1.4)	49.88 (1.2)	56.77 (2.2)	0.0	46.82 (1.6)	47.42 (1.3)
	C4.5	49.20 (2.2)	49.80 (1.8)	57.21 (0.9)	0.2	46.80 (1.1)	47.42 (1.2)
	Ripper	49.92 (3.3)	45.61 (1.8)	**57.99 (1.2)**	0.4	45.03 (1.4)	46.59 (1.9)
	BayesNet	55.65 (1.4)	48.01 (1.1)	55.31 (1.6)	0.6	41.12 (0.9)	44.00 (1.7)
	SVM	55.31 (1.4)	55.80 (1.0)	57.59 (0.7)	0.8	37.54 (1.1)	42.99 (1.6)
Phenotype	KNN	40.29 (3.9)	44.08 (1.3)	50.76 (2.6)	0.0	48.90 (0.9)	47.43 (1.7)
	C4.5	40.69 (4.3)	46.16 (2.3)	53.70 (1.6)	0.2	49.32 (1.0)	47.03 (1.4)
	Ripper	44.08 (3.4)	44.55 (1.0)	54.11 (1.1)	0.4	45.82 (1.6)	44.48 (1.7)
	BayesNet	46.69 (2.6)	44.21 (2.0)	47.95 (2.5)	0.6	25.04 (1.9)	25.45 (5.1)
	SVM	46.06 (3.0)	45.69 (1.9)	**54.23 (1.6)**	0.8	15.32 (2.1)	15.08 (3.0)
Sequence	KNN	50.73 (2.6)	49.58 (1.7)	53.67 (2.1)	0.0	48.31 (1.7)	49.11 (2.7)
	C4.5	53.24 (0.9)	51.62 (2.4)	59.22 (1.3)	0.2	48.17 (1.8)	49.10 (2.6)
	Ripper	53.07 (1.9)	45.52 (1.4)	59.73 (0.8)	0.4	46.53 (1.7)	48.47 (2.9)
	BayesNet	**61.02 (0.8)**	54.21 (1.8)	57.04 (1.8)	0.6	43.19 (1.6)	46.63 (2.0)
	SVM	45.37 (2.7)	52.69 (2.4)	59.05 (0.9)	0.8	40.01 (1.9)	46.32 (1.7)
SPO	KNN	46.05 (1.9)	44.94 (1.2)	52.43 (2.0)	0.0	45.27 (1.0)	45.83 (1.9)
	C4.5	47.14 (6.0)	47.78 (1.5)	55.02 (1.3)	0.2	44.98 (1.3)	45.83 (1.8)
	Ripper	47.34 (2.9)	45.72 (1.8)	**58.78 (1.2)**	0.4	42.27 (1.8)	45.21 (2.0)
	BayesNet	52.69 (1.7)	47.87 (1.8)	48.71 (1.8)	0.6	38.31 (1.6)	42.32 (1.8)
	SVM	50.12 (1.6)	49.68 (1.9)	54.66 (1.5)	0.8	33.86 (1.4)	41.13 (1.4)

Table 6: Comparison of local and global approaches at the second hierarchical level using the hierarchical f-measure metric.

Datasets	Classifier		Local		Threshold	Global	
		HMC-BR	HMC-LP	HMC-CT		C4.5H	Clus-HMC
Expr	KNN	31.03 (0.5)	32.05 (1.2)	30.90 (0.5)	0.0	30.32 (1.1)	32.12 (0.6)
	C4.5	30.93 (0.5)	33.20 (0.7)	24.54 (0.9)	0.2	30.74 (0.7)	32.11 (0.6)
	Ripper	32.07 (1.2)	26.81 (1.7)	27.00 (1.6)	0.4	31.57 (0.4)	32.05 (0.7)
	BayesNet	34.21 (1.2)	**35.30 (1.0)**	30.94 (1.2)	0.6	29.76 (0.9)	31.40 (0.7)
	SVM	27.70 (1.3)	26.29 (0.7)	25.89 (0.8)	0.8	26.93 (1.0)	30.50 (0.6)
CellCycle	KNN	31.76 (1.0)	32.46 (0.7)	30.88 (0.9)	0.0	29.16 (0.9)	30.53 (0.7)
	C4.5	30.71 (0.8)	32.52 (1.1)	23.74 (0.3)	0.2	29.76 (0.9)	30.52 (0.6)
	Ripper	29.16 (0.4)	24.96 (1.1)	27.17 (1.1)	0.4	30.19 (0.6)	31.01 (0.7)
	BayesNet	31.75 (1.4)	30.34 (1.2)	28.89 (0.6)	0.6	27.72 (1.0)	28.62 (0.7)
	SVM	33.63 (0.8)	**35.17 (1.1)**	32.17 (0.6)	0.8	24.81 (0.9)	27.45 (0.4)
Church	KNN	25.63 (1.2)	27.89 (0.4)	28.83 (0.6)	0.0	16.32 (1.1)	18.82 (1.7)
	C4.5	24.79 (0.9)	29.25 (0.6)	26.83 (1.0)	0.2	30.76 (0.5)	30.86 (1.0)
	Ripper	25.50 (1.1)	19.74 (0.4)	25.18 (1.4)	0.4	**33.53 (1.1)**	33.33 (1.0)
	BayesNet	28.94 (0.5)	25.46 (0.9)	24.63 (1.2)	0.6	16.96 (1.5)	18.41 (2.1)
	SVM	24.91 (0.6)	28.10 (0.7)	28.71 (1.2)	0.8	11.31 (1.0)	15.28 (2.1)
Derisi	KNN	28.75 (0.9)	29.93 (0.8)	28.98 (0.5)	0.0	27.86 (0.3)	29.57 (1.2)
	C4.5	22.97 (5.1)	**30.52 (0.3)**	23.46 (0.6)	0.2	28.19 (0.5)	29.57 (1.1)
	Ripper	26.72 (0.8)	21.31 (0.3)	26.59 (0.5)	0.4	28.84 (1.0)	29.43 (1.0)
	BayesNet	29.39 (0.9)	25.53 (1.5)	25.61 (1.1)	0.6	26.66 (0.7)	27.82 (1.4)
	SVM	28.42 (0.8)	30.49 (0.9)	29.83 (0.8)	0.8	22.04 (1.3)	25.94 (1.3)
Eisen	KNN	32.03 (1.2)	32.98 (0.8)	31.12 (0.8)	0.0	30.09 (1.0)	31.71 (0.5)
	C4.5	31.33 (1.0)	33.11 (0.4)	24.21 (1.5)	0.2	30.52 (0.9)	31.70 (0.5)
	Ripper	30.20 (1.0)	27.87 (1.6)	27.01 (1.3)	0.4	31.11 (0.8)	32.22 (0.7)
	BayesNet	33.07 (1.5)	31.56 (1.1)	29.52 (0.8)	0.6	28.36 (1.1)	29.98 (1.4)
	SVM	33.66 (0.9)	**35.64 (0.8)**	31.88 (0.8)	0.8	24.01 (1.9)	28.39 (1.2)
Gasch1	KNN	32.71 (1.2)	33.27 (0.9)	31.13 (0.9)	0.0	28.91 (0.8)	31.24 (0.8)
	C4.5	32.38 (0.8)	33.59 (1.1)	24.11 (0.7)	0.2	29.85 (0.8)	31.23 (0.8)
	Ripper	30.47 (1.4)	24.54 (0.4)	27.01 (1.9)	0.4	30.55 (1.1)	31.78 (0.5)
	BayesNet	31.75 (0.4)	32.35 (0.6)	27.44 (0.8)	0.6	28.79 (1.0)	30.12 (0.6)
	SVM	34.47 (1.0)	**35.84 (0.9)**	32.87 (0.8)	0.8	26.03 (0.9)	29.19 (0.8)
Gasch2	KNN	31.27 (0.6)	31.11 (0.7)	31.30 (1.2)	0.0	28.76 (1.1)	30.10 (0.3)
	C4.5	28.35 (1.7)	32.26 (0.9)	23.74 (0.5)	0.2	29.26 (0.7)	30.10 (0.2)
	Ripper	28.00 (1.8)	23.29 (0.9)	26.64 (0.5)	0.4	30.13 (0.6)	30.36 (1.0)
	BayesNet	30.24 (0.4)	27.09 (0.7)	28.21 (0.9)	0.6	27.67 (0.6)	28.94 (1.0)
	SVM	33.06 (0.4)	**34.57 (0.5)**	31.04 (0.5)	0.8	24.30 (0.5)	27.80 (0.7)
Phenotype	KNN	22.81 (2.6)	26.09 (1.2)	26.11 (1.1)	0.0	15.64 (0.4)	20.47 (3.2)
	C4.5	22.42 (2.8)	27.60 (1.5)	26.75 (0.7)	0.2	**33.61 (0.8)**	29.15 (2.6)
	Ripper	24.67 (2.6)	21.41 (0.6)	22.52 (0.6)	0.4	30.10 (1.4)	29.32 (2.1)
	BayesNet	28.76 (1.8)	25.38 (1.5)	23.16 (0.7)	0.6	14.15 (1.7)	14.36 (3.7)
	SVM	26.30 (2.2)	27.61 (1.4)	28.03 (1.4)	0.8	08.44 (2.2)	08.01 (1.5)
Sequence	KNN	29.30 (1.5)	30.08 (0.9)	27.92 (0.8)	0.0	30.06 (1.1)	30.95 (1.2)
	C4.5	31.21 (1.2)	32.48 (1.3)	24.01 (0.9)	0.2	30.23 (1.1)	30.94 (1.1)
	Ripper	29.81 (1.7)	24.09 (0.7)	25.93 (0.7)	0.4	30.76 (1.0)	31.35 (1.5)
	BayesNet	**34.70 (0.4)**	32.83 (1.5)	28.59 (0.3)	0.6	28.47 (0.7)	30.02 (1.2)
	SVM	27.05 (2.4)	28.06 (1.2)	27.98 (0.6)	0.8	25.67 (0.9)	29.30 (0.9)
SPO	KNN	27.58 (0.9)	29.05 (0.8)	29.23 (0.6)	0.0	27.80 (0.5)	29.68 (0.9)
	C4.5	27.91 (4.5)	31.34 (0.8)	22.79 (0.6)	0.2	28.19 (0.8)	29.68 (0.9)
	Ripper	26.30 (1.7)	23.19 (1.0)	27.65 (0.8)	0.4	28.77 (1.1)	29.92 (1.2)
	BayesNet	29.37 (1.2)	27.90 (0.7)	25.72 (0.7)	0.6	25.75 (0.8)	27.82 (1.1)
	SVM	30.70 (0.8)	**32.04 (0.9)**	29.87 (0.6)	0.8	21.54 (0.8)	26.61 (0.9)

Table 7: Comparison of local and global approaches at the third hierarchical level using the hierarchical f-measure metric.

Datasets	Classifier	Local			Threshold	Global	
		HMC-BR	HMC-LP	HMC-CT		C4.5H	Clus-HMC
Expr	KNN	22.84 (0.4)	24.71 (1.0)	17.78 (0.5)	0.0	21.30 (0.6)	24.13 (0.5)
	C4.5	23.29 (0.5)	**26.78 (0.4)**	11.27 (0.6)	0.2	22.40 (0.5)	24.13 (0.4)
	Ripper	23.51 (0.9)	19.06 (1.6)	12.98 (1.0)	0.4	25.39 (0.4)	25.15 (0.5)
	BayesNet	24.79 (1.2)	26.76 (0.8)	18.03 (0.5)	0.6	23.16 (0.6)	24.23 (0.4)
	SVM	19.32 (0.9)	17.96 (0.4)	14.97 (0.4)	0.8	19.98 (0.7)	23.14 (0.4)
CellCycle	KNN	23.33 (0.5)	24.86 (0.6)	18.16 (0.6)	0.0	19.81 (0.8)	23.06 (0.3)
	C4.5	23.29 (0.5)	26.45 (1.0)	10.51 (0.3)	0.2	21.53 (0.9)	23.06 (0.3)
	Ripper	21.21 (0.3)	17.41 (0.8)	13.57 (0.7)	0.4	24.37 (0.7)	24.69 (0.6)
	BayesNet	23.19 (1.1)	22.07 (0.9)	16.42 (0.5)	0.6	21.25 (1.0)	22.23 (0.6)
	SVM	25.31 (0.7)	**27.51 (0.5)**	19.12 (0.5)	0.8	17.99 (0.7)	20.87 (0.4)
Church	KNN	18.42 (0.9)	20.97 (0.6)	16.22 (0.5)	0.0	07.24 (0.6)	09.13 (1.4)
	C4.5	17.56 (0.7)	22.55 (0.9)	13.70 (0.4)	0.2	24.91 (0.6)	24.80 (0.9)
	Ripper	17.80 (1.0)	13.17 (0.3)	13.99 (1.1)	0.4	**26.91 (0.9)**	26.76 (1.0)
	BayesNet	19.92 (0.6)	17.69 (0.6)	13.41 (0.9)	0.6	11.80 (1.1)	13.20 (1.6)
	SVM	17.61 (0.4)	21.17 (0.8)	15.97 (0.7)	0.8	07.59 (0.7)	10.64 (1.5)
Derisi	KNN	21.20 (0.7)	23.54 (0.7)	16.60 (0.4)	0.0	19.21 (0.5)	22.30 (0.8)
	C4.5	16.40 (4.0)	25.01 (0.6)	10.71 (0.4)	0.2	20.04 (0.6)	22.29 (0.8)
	Ripper	19.11 (0.6)	14.46 (0.3)	13.82 (0.1)	0.4	23.45 (0.8)	23.86 (0.7)
	BayesNet	20.41 (0.4)	17.83 (1.1)	13.63 (0.6)	0.6	20.40 (0.5)	21.48 (1.1)
	SVM	21.54 (0.7)	**24.74 (0.3)**	16.60 (0.5)	0.8	15.86 (0.9)	19.47 (1.0)
Eisen	KNN	23.44 (0.9)	25.06 (0.6)	17.70 (0.7)	0.0	20.53 (0.9)	23.64 (0.5)
	C4.5	23.63 (0.7)	26.33 (0.5)	10.95 (0.8)	0.2	21.89 (0.8)	23.63 (0.5)
	Ripper	21.88 (0.8)	19.87 (1.3)	12.95 (0.5)	0.4	24.62 (0.4)	25.13 (0.6)
	BayesNet	23.81 (0.7)	23.14 (0.9)	16.81 (0.7)	0.6	21.61 (0.6)	22.93 (1.4)
	SVM	25.00 (0.6)	**27.39 (0.6)**	18.64 (0.4)	0.8	17.40 (1.4)	21.32 (1.0)
Gasch1	KNN	24.27 (1.1)	25.55 (0.8)	18.01 (0.8)	0.0	19.39 (0.7)	23.54 (0.7)
	C4.5	24.57 (0.6)	**27.37 (0.8)**	11.11 (0.3)	0.2	21.39 (0.7)	23.54 (0.6)
	Ripper	22.14 (0.9)	17.24 (0.4)	13.14 (1.4)	0.4	24.56 (1.0)	25.42 (0.7)
	BayesNet	21.67 (0.6)	23.73 (0.4)	15.09 (0.5)	0.6	21.96 (0.8)	23.46 (0.4)
	SVM	25.50 (0.8)	27.02 (0.8)	19.98 (0.8)	0.8	19.00 (0.6)	22.23 (0.5)
Gasch2	KNN	23.31 (0.8)	24.24 (0.5)	17.93 (0.9)	0.0	19.66 (0.8)	22.48 (0.3)
	C4.5	20.63 (1.5)	26.24 (0.8)	10.72 (0.4)	0.2	20.84 (0.6)	22.48 (0.2)
	Ripper	20.14 (1.3)	16.07 (0.7)	13.38 (0.2)	0.4	24.56 (0.6)	23.93 (0.9)
	BayesNet	21.49 (0.3)	19.63 (0.6)	15.70 (0.4)	0.6	21.37 (0.5)	22.55 (0.7)
	SVM	25.43 (0.6)	**27.63 (0.4)**	17.94 (0.2)	0.8	17.72 (0.3)	21.07 (0.4)
Phenotype	KNN	16.21 (1.9)	18.96 (1.1)	15.00 (0.9)	0.0	07.01 (0.2)	12.31 (3.1)
	C4.5	15.85 (1.9)	19.91 (0.9)	14.24 (0.7)	0.2	**26.68 (1.0)**	21.06 (2.8)
	Ripper	17.46 (1.7)	14.25 (0.4)	11.53 (0.5)	0.4	23.25 (1.6)	22.18 (2.4)
	BayesNet	20.64 (1.2)	18.61 (1.0)	13.91 (1.0)	0.6	09.89 (1.0)	10.07 (2.6)
	SVM	18.98 (1.8)	20.26 (1.1)	15.37 (1.1)	0.8	05.99 (2.0)	05.38 (1.0)
Sequence	KNN	21.15 (1.1)	22.52 (0.7)	14.59 (0.5)	0.0	21.70 (0.6)	23.35 (0.9)
	C4.5	23.56 (0.9)	**26.16 (1.0)**	10.56 (0.4)	0.2	22.32 (0.7)	23.34 (0.8)
	Ripper	21.60 (1.3)	16.96 (0.5)	12.15 (0.5)	0.4	24.80 (0.6)	25.04 (1.3)
	BayesNet	24.77 (0.3)	25.66 (1.4)	16.25 (0.4)	0.6	21.88 (0.5)	23.35 (1.1)
	SVM	19.57 (1.8)	19.22 (0.9)	16.56 (0.4)	0.8	18.89 (0.7)	22.34 (0.6)
SPO	KNN	20.30 (0.6)	22.63 (0.8)	16.43 (0.4)	0.0	18.90 (0.3)	22.21 (0.6)
	C4.5	20.42 (3.8)	25.62 (0.7)	10.26 (0.4)	0.2	19.95 (0.6)	22.20 (0.5)
	Ripper	18.83 (1.4)	15.86 (0.7)	14.33 (0.6)	0.4	23.36 (1.3)	23.73 (0.8)
	BayesNet	20.54 (0.7)	20.54 (0.5)	14.88 (0.6)	0.6	19.59 (0.5)	21.51 (0.9)
	SVM	23.85 (0.8)	**25.77 (0.7)**	17.09 (0.5)	0.8	15.51 (0.7)	20.19 (0.7)

Table 8: Comparison of local and global approaches at the fourth hierarchical level using the hierarchical f-measure metric.

Datasets	Classifier		Local		Threshold	Global	
		HMC-BR	HMC-LP	HMC-CT		C4.5H	Clus-HMC
Expr	KNN	19.28 (0.3)	21.05 (0.7)	12.28 (0.4)	0.0	16.78 (0.4)	19.98 (0.6)
	C4.5	19.84 (0.4)	**23.33 (0.5)**	07.24 (0.4)	0.2	18.08 (0.6)	19.98 (0.6)
	Ripper	19.71 (0.6)	15.48 (1.4)	08.12 (0.7)	0.4	22.12 (0.4)	21.55 (0.4)
	BayesNet	20.65 (0.8)	22.88 (0.9)	12.87 (0.4)	0.6	19.60 (0.6)	20.69 (0.3)
	SVM	16.04 (0.7)	14.41 (0.3)	09.93 (0.6)	0.8	16.51 (0.7)	19.58 (0.3)
CellCycle	KNN	19.46 (0.4)	21.16 (0.7)	12.37 (0.5)	0.0	15.39 (0.6)	19.07 (0.4)
	C4.5	19.69 (0.5)	23.12 (1.0)	06.64 (0.3)	0.2	17.45 (0.8)	19.06 (0.3)
	Ripper	17.70 (0.2)	14.07 (0.7)	08.31 (0.5)	0.4	21.26 (0.7)	21.27 (0.5)
	BayesNet	19.29 (0.9)	18.24 (0.8)	11.15 (0.4)	0.6	17.78 (1.1)	18.92 (0.5)
	SVM	21.49 (0.7)	**23.63 (0.4)**	13.49 (0.5)	0.8	14.57 (0.7)	17.60 (0.4)
Church	KNN	15.21 (0.7)	17.75 (0.7)	11.11 (0.5)	0.0	04.76 (0.4)	06.18 (1.1)
	C4.5	14.46 (0.7)	18.80 (0.9)	09.03 (0.3)	0.2	21.32 (0.6)	20.92 (1.2)
	Ripper	14.48 (0.7)	10.36 (0.2)	08.77 (0.9)	0.4	**22.96 (0.9)**	22.83 (1.3)
	BayesNet	16.22 (0.5)	14.10 (0.4)	09.23 (0.7)	0.6	09.43 (0.9)	10.70 (1.3)
	SVM	14.42 (0.3)	17.40 (0.7)	10.81 (0.6)	0.8	05.98 (0.6)	08.49 (1.2)
Derisi	KNN	17.89 (0.6)	20.06 (0.7)	11.00 (0.3)	0.0	15.05 (0.4)	18.50 (0.8)
	C4.5	13.54 (3.5)	**21.92 (0.6)**	06.80 (0.3)	0.2	15.94 (0.6)	18.49 (0.8)
	Ripper	15.79 (0.6)	11.49 (0.3)	08.54 (0.3)	0.4	20.44 (0.5)	20.57 (0.9)
	BayesNet	16.66 (0.3)	14.35 (0.9)	09.29 (0.5)	0.6	16.96 (0.3)	18.10 (0.9)
	SVM	18.36 (0.6)	21.56 (0.2)	11.28 (0.4)	0.8	12.85 (0.8)	16.18 (0.8)
Eisen	KNN	18.95 (0.7)	21.12 (0.5)	11.80 (0.5)	0.0	15.93 (0.7)	19.39 (0.5)
	C4.5	19.87 (0.6)	22.74 (0.3)	06.94 (0.5)	0.2	17.44 (0.6)	19.38 (0.5)
	Ripper	18.14 (0.9)	16.35 (1.2)	08.09 (0.3)	0.4	20.96 (0.5)	21.18 (0.7)
	BayesNet	19.48 (0.5)	19.19 (0.9)	11.07 (0.5)	0.6	18.04 (0.5)	19.32 (1.5)
	SVM	20.98 (0.5)	**23.54 (0.3)**	12.95 (0.3)	0.8	14.09 (1.1)	17.84 (0.9)
Gasch1	KNN	20.58 (1.1)	21.92 (0.6)	12.44 (0.6)	0.0	14.77 (0.8)	19.55 (0.6)
	C4.5	20.78 (0.5)	**23.77 (0.9)**	07.19 (0.2)	0.2	17.15 (0.8)	19.55 (0.6)
	Ripper	18.51 (0.8)	13.89 (0.4)	08.16 (0.9)	0.4	21.25 (0.8)	21.95 (0.8)
	BayesNet	17.73 (0.7)	19.69 (0.4)	10.81 (0.5)	0.6	18.35 (0.8)	20.00 (0.4)
	SVM	21.61 (0.7)	22.59 (0.8)	14.11 (0.6)	0.8	15.51 (0.5)	18.72 (0.4)
Gasch2	KNN	19.82 (0.8)	20.67 (0.4)	12.12 (0.9)	0.0	15.19 (0.9)	18.64 (0.4)
	C4.5	17.20 (1.2)	23.06 (0.7)	06.86 (0.3)	0.2	16.48 (0.5)	18.63 (0.3)
	Ripper	16.82 (1.0)	12.94 (0.6)	08.41 (0.2)	0.4	21.35 (0.6)	20.55 (0.8)
	BayesNet	17.79 (0.2)	16.19 (0.5)	10.79 (0.6)	0.6	17.95 (0.4)	19.19 (0.7)
	SVM	21.76 (0.7)	**24.02 (0.3)**	12.42 (0.2)	0.8	14.42 (0.2)	17.70 (0.5)
Phenotype	KNN	13.48 (1.8)	15.62 (1.0)	10.01 (0.6)	0.0	04.71 (0.1)	09.05 (2.6)
	C4.5	13.17 (1.6)	16.12 (0.6)	09.09 (0.4)	0.2	**22.82 (0.9)**	17.01 (2.9)
	Ripper	14.50 (1.4)	11.23 (0.3)	07.05 (0.5)	0.4	19.86 (1.8)	18.48 (2.3)
	BayesNet	17.47 (1.1)	14.90 (0.8)	10.33 (1.0)	0.6	07.95 (0.9)	08.27 (2.1)
	SVM	15.93 (1.5)	16.58 (0.8)	10.13 (0.5)	0.8	04.84 (1.5)	04.27 (0.8)
Sequence	KNN	17.61 (0.9)	18.89 (0.6)	09.57 (0.3)	0.0	17.25 (0.7)	19.16 (0.8)
	C4.5	19.98 (0.7)	**22.90 (0.9)**	06.64 (0.2)	0.2	18.12 (0.6)	19.16 (0.7)
	Ripper	18.11 (1.2)	13.77 (0.4)	07.69 (0.4)	0.4	21.48 (0.5)	21.27 (1.2)
	BayesNet	21.27 (0.4)	22.15 (1.2)	11.32 (0.5)	0.6	18.40 (0.2)	19.84 (1.0)
	SVM	17.04 (1.0)	15.37 (0.7)	11.38 (0.2)	0.8	15.56 (0.6)	18.77 (0.6)
SPO	KNN	17.04 (0.5)	19.29 (0.8)	10.71 (0.3)	0.0	14.51 (0.5)	18.40 (0.4)
	C4.5	16.83 (3.1)	**22.34 (0.7)**	06.56 (0.2)	0.2	15.72 (0.5)	18.40 (0.3)
	Ripper	15.61 (1.2)	12.65 (0.5)	08.91 (0.6)	0.4	20.25 (1.2)	20.29 (0.5)
	BayesNet	16.70 (0.7)	16.98 (0.5)	10.83 (0.5)	0.6	16.35 (0.5)	18.20 (0.7)
	SVM	20.33 (0.6)	22.33 (0.5)	11.74 (0.2)	0.8	12.59 (0.6)	16.93 (0.7)

It can be observed in these tables that, in most of the results, the performance of the methods decreases as the class level becomes deeper. This is expected, because the deeper a hierarchical level, the larger the number of classes involved and the smaller the number of instances per class, making the classification process more difficult. In addition, the error propagation problem is associated with the methods based on the local approach, i.e., misclassifications at shallower hierarchical levels are propagated to the deeper levels, contributing to decrease the classification performance at these levels.

The HMC-CT, Clus-HMC and HC4.5 methods made more errors in all class levels, probably due to the large number of class predictions made for each instance. These methods achieved high Micro Hierarchical Recall values and low Micro Hierarchical Precision values. High Recall values and low Precision values can indicate that the methods make more errors because of the increasing number of false positives and the decreasing number of true positives.

Despite the higher number of errors, Table 5 shows that the best performance at the first level was obtained by the HMC-CT method. Although less accurate at the first level, making more errors, the HMC-CT method achieved a better coverage on the instances of the dataset, obtaining a higher Micro Hierarchical Recall value. Thus, this method achieved a better balance between the Micro Hierarchical Precision and Recall metrics, resulting in a better performance for the Hierarchical-F_β metric.

The smaller number of classes at the first level also seemed to favor the HMC-CT method. When using the data more than once during the training process, in a reduced number of classes (four classes at the first level), the data may have become less sparse, which improved the classification performance.

At the second, third and fourth levels (Tables 6, 7 and 8), the best results were obtained by the HMC-LP method. The evaluation metrics used consider the distances between the classes of the hierarchy, which may have contributed to this best performance.

An analysis of the errors committed by the methods showed that the HMC-LP method committed more errors in the subtrees rooted at the classes "01" and "10". In the datasets, these subtrees have most of the classes of the hierarchical structure and also have many classes that are leaf nodes at the third level (the class hierarchy has four levels). The subtrees rooted by the classes "02"and "11" have fewer classes. Besides, the subtree rooted at the class "02" does not have classes at the fourth level of the hierarchy.

When an instance x is a false positive for a class c, the false positive and false negative contributions are calculated through the sum of the distances between all real classes of x and the predicted class c assigned to it. When the instance is classified as a false negative for the class c, the contributions are calculated through the sum of the distances of all classes assigned to the instances and the class c. Therefore, it may be the case that the distances between the predicted and real classes, in the final classification produced by the HMC-LP method, were smaller than in the other methods. As the metric considers that closer classes are more similar, the misclassifications were less penalized, resulting in a better classification performance.

The methods based on the global approach had the worst performance because, depending on the threshold value used, they predicted a larger number of classes than the methods based on the local approach. The Hierarchical Precision and Recall metrics reflect the variation in the classification performance of the methods according to the threshold values selected. In many cases, the global methods obtained values for these metrics higher than values obtained by the methods based on the local approach. The larger number of predicted classes is also a feature of the HMC-CT method. In this method, the training instances are used several times due to the class decomposition process, increasing the number of classifiers, and, as a result, increasing the number of predictions.

Despite the worst predictive performance overall, it is possible to observe that, in some cases, the methods based on the global approach achieved better performance than the methods based on the local approach in the last hierarchical levels, especially in comparison with the HMC-CT method. A possible reason is that in the local approach, misclassifications in the shallower levels of the class hierarchy are propagated to the deeper levels. This error propagation problem is not present in the methods based on the global approach.

The variation of the threshold values used in the methods based on the global approach is another characteristic that influences their predictive performance. It is possible to see that, in some cases, the use of different thresholds values significantly influenced the accuracy obtained by the HC4.5 and Clus-HMC methods, as can be seen in the Church and Phenotype datasets. Therefore, the value selected for the threshold may be an important parameter in such methods, and it can be adjusted according to the desirable behavior. High threshold values lead to more precise classifiers, while low threshold values lead to classifiers with a higher coverage (recall) of the instances.

It is important to consider that, although the worst values of predictive performance were obtained by the methods based on the global approach, these methods produce a less complex (smaller) classifier, which may result in classification models easier to be interpreted by users. The HC4.5 and Clus-HMC methods both produce decision trees that can be translated into a set of rules, making the model more interpretable for specialists in the problem domain. Such aspects were not taken into consideration in this work, but a multi-objective evaluation could be used to consider the interpretability of the generated classifiers.

Although this paper addressed a problem related to functional genomics, there are other application domains which could be investigated, like, for example, human diseases. In fact, examples of HMC problems can be found in many domains, like protein function prediction [36, 2, 47, 33], text classification [21, 5], and image classification [18]. The multilabel hierarchical classification methods proposed in this work are generic enough to be applied to all these and other types of problems, in the same sense that a single-label flat classification algorithm can be applied to any application domain.

The comparisons of the algorithms' predictive accuracies using statistical significance tests are shown in Tables 10, 11, 12 and 13. To facilitate the understanding of the tables, the symbols presented in Table 9 are used.

Table 9: Legend for the results of statistical tests.

Symbol	Meaning
▲	Indicates that the algorithm located at the row of the table obtained statistically significant better results than the algorithm located at the column of the table
△	Indicates that the algorithm located at the row of the table obtained better results than the algorithm located at the column of the table, but with no statistically significant difference
▽	Indicates that the algorithm located at the column of the table obtained better results than the algorithm located at the row of the table, but with no statistically significant difference
▼	Indicates that the algorithm located at the column of the table obtained statistically significant better results than the algorithm located at the row of the table

Table 10: Results of statistical tests at the first hierarchical level.

		HMC-BR				HMC-LP					HMC-CT					HC4.5					Clus-HMC				
		C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	0.0	0.2	0.4	0.6	0.8	0.0	0.2	0.4	0.6	0.8
HMC-BR	KNN	△	△	▽	△	△	△	△	▽	▽	▽	▽	▽	▽	▽	△	△	△	▲	▲	△	△	△	△	△
	C4.5		▽	▽	▽	△	▽	△	▽	▽	▽	▽	▽	▽	▽	△	△	△	△	△	△	▽	△	△	△
	Rip			▽	△	△	△	△	▽	▽	▽	▽	▽	▽	▽	△	△	△	▲	▲	△	△	△	△	△
	BN				△	△	△	▲	△	△	△	▽	▽	△	▽	△	△	▲	▲	▲	△	△	▲	▲	▲
	SVM					△	△	△	▽	▽	▽	▽	▽	▽	▽	△	△	△	▲	▲	△	△	△	△	△
HMC-LP	KNN						▽	△	▽	▽	▽	▽	▼	▽	▽	△	△	△	△	△	△	▽	△	△	△
	C4.5							△	▽	▽	▽	▽	▽	▽	▽	△	△	△	△	△	△	△	△	▲	△
	Rip								▽	▽	▼	▼	▼	▽	▼	▽	▽	△	△	△	▽	▽	▽	△	△
	BN									▽	▽	▽	▽	▽	▽	△	△	△	▲	▲	△	△	△	△	△
	SVM										▽	▽	▽	▽	▽	△	△	△	▲	▲	△	△	△	▲	▲
HMC-CT	KNN											▽	▽	▽	▽	△	△	▲	▲	▲	△	△	△	▲	▲
	C4.5												▽	△	▽	▲	△	△	▲	▲	△	△	▲	▲	▲
	Rip													△	△	△	▲	▲	▲	▲	▲	△	▲	▲	▲
	BN														▽	△	△	▲	▲	▲	△	△	△	▲	▲
	SVM															▲	▲	▲	▲	▲	△	△	▲	▲	▲
HC4.5	0.0																▽	△	△	△	▽	▽	△	△	△
	0.2																	△	△	△	▽	▽	△	△	△
	0.4																		△	△	▽	▽	▽	△	△
	0.6																			△	▽	▽	▽	▽	▽
	0.8																				▽	▽	▽	▽	▽
Clus-HMC	0.0																					▽	▽	△	△
	0.2																						▽	△	△
	0.4																							△	△
	0.6																								△

Table 11: Results of statistical tests at the second hierarchical level.

		HMC-BR				HMC-LP					HMC-CT					HC4.5					Clus-HMC				
		C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	0.0	0.2	0.4	0.6	0.8	0.0	0.2	0.4	0.6	0.8
HMC-BR	KNN	△	△	▽	▽	▽	▽	△	△	▽	△	△	△	△	▽	△	△	▽	△	▲	△	▽	▽	△	△
	C4.5		△	▽	▽	▽	▽	△	▽	▽	▽	△	△	△	▽	△	▽	▽	△	△	▽	▽	▽	△	△
	Rip			▽	▽	▽	▽	△	▽	▽	▽	△	△	△	▽	△	▽	▽	△	△	▽	▽	▽	△	△
	BN				△	△	▽	▲	△	△	△	▲	▲	△	△	△	△	△	▲	▲	△	△	▽	△	▲
	SVM					▽	▽	▲	△	▽	△	▲	△	△	△	△	△	△	△	▲	△	▽	▽	△	△
HMC-LP	KNN						▽	▲	△	▽	△	▲	▲	△	△	△	△	△	▲	▲	△	△	▽	△	▲
	C4.5							▲	△	△	△	▲	▲	▲	△	▲	△	△	▲	▲	△	△	△	▲	▲
	Rip								▽	▼	▽	▽	▽	▽	▼	▽	▽	▼	▽	△	▽	▼	▼	▽	▽
	BN									▽	▽	△	△	△	▽	△	▽	▽	△	△	▽	▽	▽	△	△
	SVM										△	▲	▲	△	△	△	△	△	▲	▲	△	△	▽	△	▲
HMC-CT	KNN											△	△	△	▽	△	△	▽	△	▲	▽	▽	▽	△	△
	C4.5												▽	▽	▼	▽	▽	▼	▽	△	▽	▼	▼	▽	▽
	Rip													▽	▽	▽	▽	▽	△	△	▽	▼	▼	▽	▽
	BN														▽	▽	▽	▽	△	△	▽	▽	▽	▽	△
	SVM															△	△	△	△	▲	△	▽	▽	△	△
HC4.5	0.0																▽	▽	△	△	▽	▽	▽	▽	△
	0.2																	▽	△	▲	▽	▽	▽	△	△
	0.4																		△	▲	△	▽	▽	△	△
	0.6																			△	▽	▼	▼	▽	▽
	0.8																				▼	▼	▼	▽	▽
Clus-HMC	0.0																					▽	▽	△	△
	0.2																						▽	△	▲
	0.4																							△	▲
	0.6																								△

Table 12: Results of statistical tests at the third hierarchical level.

		HMC-BR				HMC-LP					HMC-CT					HC4.5					Clus-HMC				
		C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	0.0	0.2	0.4	0.6	0.8	0.0	0.2	0.4	0.6	0.8
HMC-BR	KNN	△	△	▽	▽	▽	▽	△	▽	▽	△	▲	△	△	△	△	△	▽	△	△	△	▽	▽	△	△
	C4.5		△	▽	▽	▽	▽	△	▽	▽	△	△	△	△	△	△	△	▽	△	△	▽	▽	▽	△	△
	Rip			▽	▽	▽	▽	△	▽	▽	△	△	△	△	△	△	▽	▽	△	△	▽	▽	▽	▽	△
	BN				▽	▽	▽	△	△	▽	△	▲	▲	▲	△	△	△	▽	△	▲	△	▽	▽	△	△
	SVM					▽	▽	▲	△	▽	△	▲	▲	▲	△	△	△	▽	△	▲	△	▽	▽	△	△
HMC-LP	KNN						▽	▲	△	△	▲	▲	▲	▲	▲	▲	△	▽	△	▲	△	△	▽	△	△
	C4.5							▲	△	△	▲	▲	▲	▲	▲	▲	△	△	▲	▲	△	△	△	△	▲
	Rip								▽	▼	▽	△	△	△	▽	▽	▽	▼	▽	△	▽	▼	▼	▽	▽
	BN									▽	△	▲	▲	△	△	△	△	▽	△	▽	▽	△	△	▽	▽
	SVM										▲	▲	▲	▲	▲	▲	△	▽	△	▲	△	△	▽	△	△
HMC-CT	KNN											△	△	△	▽	▽	▽	▼	▽	△	▽	▽	▼	▽	▽
	C4.5												▽	▽	▽	▽	▽	▼	▽	▽	▼	▼	▼	▽	▽
	Rip													▽	▽	▽	▽	▼	▽	▽	▽	▼	▼	▽	▽
	BN														▽	▽	▽	▼	▽	▽	▽	▼	▼	▽	▽
	SVM															▽	▽	▼	▽	△	▽	▽	▼	▽	▽
HC4.5	0.0																▽	▼	▽	△	▽	▽	▼	▽	▽
	0.2																	▽	△	△	▽	▽	▽	▽	△
	0.4																		△	△	▲	△	▽	△	△
	0.6																			△	▽	▽	▽	▽	△
	0.8																				▽	▼	▼	▽	▽
Clus-HMC	0.0																					▽	▽	△	△
	0.2																						▽	△	△
	0.4																							△	▲
	0.6																								△

Table 13: Results of statistical tests at the fourth hierarchical level.

		HMC-BR				HMC-LP					HMC-CT					HC4.5					Clus-HMC				
		C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	KNN	C4.5	Rip	BN	SVM	0.0	0.2	0.4	0.6	0.8	0.0	0.2	0.4	0.6	0.8
HMC-BR	KNN	△	△	▽	▽	▽	▽	△	△	▽	△	▲	▲	△	△	△	△	▽	△	△	△	▽	▽	△	△
	C4.5		△	▽	▽	▽	▽	△	△	▽	△	▲	▲	△	△	△	△	▽	△	△	△	▽	▽	△	△
	Rip			▽	▽	▽	▽	△	▽	▽	△	△	△	△	△	△	▽	▽	△	△	▽	▽	▽	▽	△
	BN				▽	▽	▽	▲	△	▽	△	▲	▲	△	△	△	△	▽	△	△	△	▽	▽	△	△
	SVM					▽	▽	△	△	▽	▲	▲	▲	▲	△	△	△	▽	△	△	△	▽	▽	△	△
HMC-LP	KNN						▽	▲	△	△	▲	▲	▲	▲	▲	▲	△	▽	△	▲	△	△	▽	△	△
	C4.5							▲	△	△	▲	▲	▲	▲	▲	▲	△	△	▲	▲	△	△	△	△	▲
	Rip								▽	▼	△	△	△	△	△	△	▽	▼	▽	△	▽	▽	▼	▽	▽
	BN									▽	△	▲	▲	△	△	△	△	▽	△	△	△	▽	▽	▽	△
	SVM										▲	▲	▲	▲	▲	▲	△	▽	△	▲	△	△	▽	△	△
HMC-CT	KNN											△	△	△	▽	▽	▽	▼	▽	▽	▽	▼	▼	▽	▽
	C4.5												▽	▽	▽	▽	▽	▼	▽	▽	▽	▼	▼	▼	▽
	Rip													▽	▽	▽	▽	▼	▽	▽	▽	▼	▼	▼	▽
	BN														▽	▽	▽	▼	▽	▽	▽	▼	▼	▽	▽
	SVM															▽	▽	▼	▽	▽	▽	▽	▼	▽	▽
HC4.5	0.0																▽	▼	▽	△	▽	▼	▽	▽	▽
	0.2																	▽	△	△	▽	▽	▽	▽	△
	0.4																		△	▲	△	△	▽	△	△
	0.6																			△	▽	▽	▽	▽	△
	0.8																				▽	▽	▼	▽	▽
Clus-HMC	0.0																					▽	▽	▽	△
	0.2																						▽	△	△
	0.4																							△	△
	0.6																								△

7 Conclusions and Future Work

This paper investigated the Hierarchical Multilabel Classification (HMC) problem. In this problem, classes are structured in a hierarchy, where classes can be superclasses or subclasses of other classes, and each example can simultaneously belong to more than one class. These two aspects increase the difficulty of the classification task.

The hierarchical classification methods investigated in this paper are divided into two approaches: local and global. The local approach performs the classification one class level at a time, discriminating among sibling classes of the hierarchy at each step, using a "Divide and Conquer" strategy. The local approach allows the development of methods using conventional ML algorithms as base classifiers. In this work, three methods based on this approach were implemented. The HMC-BR method, previously proposed in the literature; and two new methods proposed in this work, named HMC-LP and HMC-CT. These methods were evaluated using five ML algorithms as base classifiers: KNN, C4.5, Ripper, BayesNet and SVM. Methods following the global approach, unlike the local approach, induce classifiers considering all classes simultaneously. As a result, a single classifier is used for the classification process, instead of conventional ML algorithms. In this work, two existing methods based on this approach were evaluated: HC4.5 and Clus-HMC.

Experiments were performed in order to compare the local and global approaches. As HMC problems are common in bioinformatics, the experiments used datasets related with functional genomics. In particular, 10 datasets of the *Saccharomyces cerevisiae* organism, a specific type of *Yeast*, were used. These datasets are structured according to the schema of the FunCat catalog (http://mips.gsf.de/projects/funcat), developed by MIPS, and describe different aspects of the genes in the *Yeast* genome, like sequence statistics, phenotype and expression.

The experimental results were evaluated with the use of specific metrics for HMC problems, based on the distances between the real and predicted classes in the hierarchy. All results were reported separately for each level of the class hierarchies, and statistical tests were used to analyze the statistical significance of the differences in the predictive accuracy performances of the different HMC methods.

The results show that the proposed local HMC methods can obtain predictive accuracy better than, or similar to, the global methods investigated. These results were observed mainly for the HMC-Label-Powerset method proposed in this work.

As future work, HMC methods for *non-mandatory leaf node classification* will be investigated. In these problems, the deepest classification level associated with each instance is automatically defined by the classifier, without the requirement for the instances to be always assigned to classes represented by leaf nodes.

In the local approach, an important improvement that can be incorporated is a mechanism for the correction of the error propagation problem. This mechanism could detect, at each step of the local strategy, the instances that were previously incorrectly classified and perform their reclassification later. The incorporation of such mechanism could improve the predictive performance of the local approach.

Techniques for combining classifiers can also be employed to improve the predictive accuracy of methods based on the local approach. Ensemble strategies have been already adopted in the development of hierarchical single-label local classification methods, as the method proposed in [13], improving their predictive performance.

Different local approaches can be combined to improve the classification performance. According to the experiments reported in this work, the best performance at the first level was obtained by the HMC-CT method. Therefore, this method could be combined with the HMC-LP method, through the use of HMC-CT at the first level and the HMC-LP method at the other levels.

The development of methods based on the global approach is also a promising research direction. Although more complex to develop, these methods usually produce a simpler and more interpretable classification model than the models induced by methods based on the local approach, especially if the generated model is a decision tree or a set of classification rules, such as the HC4.5 and Clus-HMC methods used in this work.

The consideration of other types of hierarchical structures, such as hierarchies structured as Directed Acyclic Graphs (DAG), is also a topic that deserves future research. In DAG structures, a node can have more than one parent in the hierarchy, which makes the classification process more difficult. To consider class hierarchies structured as DAGs, modifications are necessary in the classification methods.

The evaluation of hierarchical multilabel classifiers also presents good opportunities for future studies. Although several metrics have been proposed in the literature, many considerations can still be made with respect to their performance. Moreover, new evaluation metrics can be developed, and modifications can be incorporated into the existing metrics.

Finally, in addition to the analysis of different evaluation metrics, one can analyse how different classification methods are influenced by different hierarchical and multilabel characteristics of the datasets. This study can support the improvement of existing methods and the development of new methods.

Acknowledgments

The authors would like to thank the Brazilian research councils CAPES, CNPq and FAPESP for their financial support, the Katholieke Universiteit of Leuven's Machine Learning Research Group for the Clus-HMC program and the datasets used, Amanda Clare for the HC4.5 program, and Thiago Covões for the codes of the statistical tests.

References

[1] D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991.

[2] R. Alves, M. Delgado, and A. Freitas. Multi-label hierarchical classification of protein functions with artificial immune systems. In *III Brazilian Symposium on Bioinformatics*, volume 5167 of *Lecture Notes in Bioinformatics*, pages 1–12, Berlin, Heidelberg, 2008. Springer-Verlag.

[3] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet*, 25(1):25–29, May 2000.

[4] Z. Barutcuoglu, R. E. Schapire, and O. G. Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, 2006.

[5] P. N. Bennett and N. Nguyen. Refined experts: improving classification in large taxonomies. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 11–18, New York, NY, USA, 2009. ACM.

[6] H. Blockeel, L. De Raedt, and J. Ramon. Top-down induction of clustering trees. In *Proceedings of the 15th International Conference on Machine Learning*, pages 55–63. Morgan Kaufmann, 1998.

[7] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

[8] L. Cai and T. Hofmann. Exploiting known taxonomies in learning overlapping concepts. In *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*, pages 714–719, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[9] A. Carvalho and A. Freitas. *A tutorial on multi-label classification techniques*, volume 5 of *Studies in Computational Intelligence 205*, pages 177–195. Springer, September 2009.

[10] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Incremental algorithms for hierarchical classification. *Machine Learning*, 7:31–54, 2006.

[11] A. Clare and R. D. King. Predicting gene function in saccharomyces cerevisiae. *Bioinformatics*, 19:42–49, 2003.

[12] W. W. Cohen. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123, 1995.

[13] E. P. Costa, A. C. Lorena, A. C. Carvalho, and A. A. Freitas. Top-down hierarchical ensembles of classifiers for predicting g-protein-coupled-receptor functions. In *III Brazilian Symposium on Bioinformatics*, volume 5167 of *Lecture Notes in Bioinformatics*, pages 35–46, Berlin, Heidelberg, 2008. Springer-Verlag.

[14] E. P. Costa, A. C. Lorena, A. C. Carvalho, A. A. Freitas, and N. Holden. Comparing several approaches for hierarchical classification of proteins with decision trees. In *II Brazilian Symposium on Bioinformatics*, volume 4643 of *Lecture Notes in Bioinformatics*, pages 126–137, Berlin, Heidelberg, 2007. Springer-Verlag.

[15] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Machine Learning*, 2:265–292, 2002.

[16] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[17] E. Dimitriadou, K. Hornik, F. Leisch, D. Meyer, and A. Weingessel. r-cran-e1071, 2008.

[18] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Dzeroski. Hierarchical annotation of medical images. In *11th International Multiconference Information Society*, volume A, pages 174–177, 2008.

[19] A. Elisseeff and J. Weston. Kernel methods for multi-labelled classification and categorical regression problems. In *In Advances in Neural Information Processing Systems 14*, pages 681–687. MIT Press, 2001.

[20] A. Esuli, T. Fagni, and F. Sebastiani. TreeBoost.mh: A boosting algorithm for multi-label hierarchical text categorization. In *In Proceedings of the 13th International Symposium on String Processing and Information Retrieval (SPIRE06)*, pages 13–24, 2006.

[21] A. Esuli, T. Fagni, and F. Sebastiani. Boosting multi-label hierarchical text categorization. *Inf. Retr.*, 11(4):287–313, 2008.

[22] A. A. Freitas and A. C. Carvalho. *A Tutorial on Hierarchical Classification with Applications in Bioinformatics.*, volume 1, chapter VII, pages 175–208. Idea Group, January 2007. Research and Trends in Data Mining Technologies and Applications.

[23] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[24] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Mach. Learn.*, 29(2-3):131–163, 1997.

[25] W. Hersh, C. Buckley, T. J. Leone, and D. Hickam. Ohsumed: an interactive retrieval evaluation and new large test collection for research. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–201, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[26] K. Hornik, C. Buchta, and A. Zeileis. Open-source machine learning: R meets Weka. *Computational Statistics*, 24(2):225–232, 2009.

[27] S. Kiritchenko, S. Matwin, and A. 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*, pages 30–34, Pisa, Italy, 2004.

[28] W. Krauth and M. Mezard. Learning algorithms with optimal stability in neural networks. *Journal of Physics A: Mathematical and General*, 20(11):L745, 1987.

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

[30] H. W. Mewes, D. Frishman, U. Güldener, G. Mannhaupt, K. Mayer, M. Mokrejs, B. Morgenstern, M. Münsterkötter, S. Rudd, and B. Weil. Mips: a database for genomes and protein sequences. *Nucleic Acids Res*, 30(1):31–34, 2002.

[31] P. B. Nemenyi. *Distribution-free multiple comparisons.* PhD thesis, Princeton University, 1963.

[32] I. U. of Biochemistry, M. Biology., and E. C. Webb. *Enzyme nomenclature 1992 : recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology on the nomenclature and classification of enzymes / prepared for NC-IUBMB by Edwin C. Webb.* Published for the International Union of Biochemistry and Molecular Biology by Academic Press, San Diego, 1992.

[33] F. Otero, A. Freitas, and C. Johnson. A hierarchical classification ant colony algorithm for predicting gene ontology terms. In C. Pizzuti, M. Ritchie, and M. Giacobini, editors, *Proc. 7th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics (EvoBio-2009)*, volume Lecture Notes in Computer Science 5483, pages 68–79. Springer, March 2009.

[34] J. R. Quinlan. *C4.5: programs for machine learning.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[35] R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2008.

[36] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research*, 7:1601–1626, 2006.

[37] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *Machine Learning*, volume 37, pages 297–336, Hingham, MA, USA, 1999. Kluwer Academic Publishers.

[38] X. Shen, M. Boutell, J. Luo, and C. Brown. Multi-label machine learning and its application to semantic scene classification. volume 5307, pages 188–199, 2004.

[39] C. Silla Jr and A. Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, (published online on 7 April, 42 pages), 2010. http://dx.doi.org/10.1007/s10618-010-0175-9.

[40] A. Sun and E.-P. Lim. Hierarchical text classification and evaluation. In *Fourth IEEE International Conference on Data Mining*, pages 521–528, 2001.

[41] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 25–32. MIT Press, 2003.

[42] K. Toutanova, F. Chen, K. Popat, and T. Hofmann. Text classification in a hierarchical mixture model for small training sets. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 105–113, New York, NY, USA, 2001. ACM.

[43] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

[44] G. Tsoumakas and I. Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *Proceedings of the 18th European Conference on Machine Learning (ECML 2007)*, pages 406–417, Warsaw, Poland, 2007.

[45] G. Valentini. True path rule hierarchical ensembles. In *MCS '09: Proceedings of the 8th International Workshop on Multiple Classifier Systems*, volume 5519 of *Lecture Notes in Bioinformatics*, pages 232–241, Berlin, Heidelberg, 2009. Springer-Verlag.

[46] V. N. Vapnik. *The Nature of Statistical Learning Theory (Information Science and Statistics)*. Springer-Verlag New York, Inc., 1999.

[47] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2):185–214, 2008.

[48] WIPO. World Intellectual Property Organization, 2001. http://www.wipo.int/portal/index.html.en.