Noname manuscript No.
(will be inserted by the editor)

Evaluation Measures for Hierarchical Classification: a Unified View and Novel Approaches

Aris Kosmopoulos · Ioannis Partalas · Eric Gaussier · Georgios Paliouras · Ion Androutsopoulos

Abstract Hierarchical classification addresses the problem of classifying items into a hierarchy of classes. An important issue in hierarchical classification is the evaluation of different classification algorithms, an issue which is complicated by the hierarchical relations among the classes. Several evaluation measures have been proposed for hierarchical classification using the hierarchy in different ways without however providing a unified view of the problem. This paper studies the problem of evaluation in hierarchical classification by analysing and abstracting the key components of the existing performance measures. It also proposes two alternative generic views of hierarchical evaluation and introduces two corresponding novel measures. The proposed measures, along with the state-of-the-art ones, are empirically tested on three large datasets from the domain of text classification. The empirical results illustrate

Aris Kosmopoulos
National Center for Scientific Research "Demokritos" and
Athens University of Economics and Business,
Athens, Greece
E-mail: akosmo@iit.demokritos.gr

Ioannis Partalas
Laboratoire d'Informatique de Grenoble, Univesité Joseph Fourier,
Grenoble, France
E-mail: ioannis.partalas@imag.fr

Eric Gaussier
Laboratoire d'Informatique de Grenoble, Univesité Joseph Fourier,
Grenoble, France
E-mail: eric.gaussier@imag.fr

Georgios Paliouras
National Center for Scientific Research "Demokritos",
Athens, Greece
E-mail: paliourg@iit.demokritos.gr

Ion Androutsopoulos
Athens University of Economics and Business,
Athens, Greece
E-mail: ion@aueb.gr

the undesirable behaviour of existing approaches and how the proposed methods overcome most of these problems across a range of cases.

1 Introduction

Hierarchical classification addresses the problem of classifying items into a hierarchy of classes. In past years mainstream classification research did not place enough emphasis on the presence of relations between the classes, in our cases hierarchical relations. This is gradually changing and more effort is put into hierarchical classification in particular, partly because many real-world knowledge systems and services use a hierarchical scheme to organize their data (e.g. Yahoo, Wikipedia). Research in hierarchical classification has become important, because flat classification algorithms are ill-equipped to address large scale problems with hundreds of thousands of hierarchically related classes. Promising initial results on large-scale problems show that hierarchical classifiers can be effective in improving information retrieval [Kosmopoulos et al., 2010].

Many research questions in hierarchical classification remain open. An important issue is how to properly evaluate hierarchical classification algorithms. While standard flat classification problems have benefited from established measures such as precision and recall, there are no established evaluation measures for hierarchical classification tasks, where the assessment of an algorithm becomes more complicated due to the relations among the classes. For example, classification errors in the upper levels of the hierarchy (e.g. when wrongly classifying a document of the class `music` into the class `food`) are more severe than those in deeper levels (e.g. when classifying a document from `progressive rock` as `alternative rock`). Several evaluation measures have been proposed for hierarchical classification (HC) [Costa et al., 2007, Sokolova and Guy, 2009] using the hierarchy in different ways. Nevertheless, none of them is widely adopted, making it very difficult to compare the performance of different HC algorithms.

A number of comparative studies of HC performance measures have been published. An early study can be found in [Sun et al., 2003], which is limited to a particular type of graph-distance measures. A review of HC measures is presented in [Costa et al., 2007], focusing on single-label tasks and without providing any empirical results; in multi-label tasks each object can be assigned to more than one classes, e.g. a newspaper article may belong to both `politics` and `economics`. In [Nowak et al., 2010] many multi-label evaluation measures are compared, but the role of the hierarchy is not fully considered. Finally, Brucker et al. [2011] provide a comprehensive empirical analysis of performance measures, but they focus on the evaluation of clustering methods rather than classification ones. While these studies provide interesting insights, they all miss important aspects of the problem of evaluating HC algorithms. In particular, they do not abstract the problem in order to describe existing evaluation measures within a common framework.

The work presented here addresses these issues by analysing and abstracting the key components of existing HC performance measures. More specifically:

1. It groups existing HC evaluation measures under two main types and provides a generic framework for each type, based on flow networks and set theory.
2. It provides a critical overview of the existing HC performance measures using the proposed framework.
3. It introduces two new HC evaluation measures that address important deficiencies of state-of-the-art measures.
4. It provides comparative empirical results on large HC datasets from text classification with a variety of HC algorithms.

The remainder of this paper is organized as follows. Section 2 introduces the problem of HC, presents general requirements for HC measures and the proposed frameworks. Furthermore, it presents existing HC evaluation measures using the proposed frameworks and introduces two new measures that address problems the state-of-the-art measures have. Section 3 presents a case study comparison and analysis of the proposed measures and the existing ones. Section 4 describes the empirical setting, along with the data of the empirical analysis of the measures and then presents and discusses the empirical results. Finally, Section 5 concludes and summarizes remaining open issues.

2 A Framework for Hierarchical Classification Performance Measures

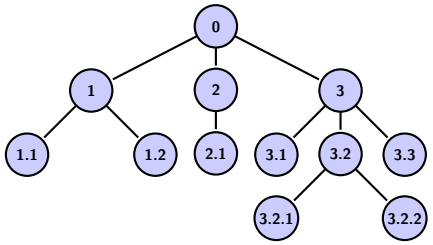
This section presents a new framework within which HC performance measures can be described and characterized. Firstly, the main dimensions of the problem are defined and then the general requirements for the evaluation are presented and discussed, based on standard problems that appear in hierarchical classification. We then proceed with the presentation of the proposed framework, which is used in further sections to describe and analyse the measures.

2.1 The main dimensions of the Hierarchical Classification Problem

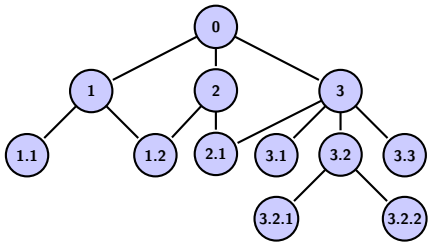
In classification tasks the training set is typically denoted as $S = \left\{(\mathbf{x}^i, \mathbf{y}^i)\right\}_{i=1}^{n}$, where $\mathbf{x}^i \in \mathcal{X}$ is the feature vector of instance i in the input space $\mathcal{X}$ and $\mathbf{y}^i \subseteq \mathcal{Y}$ is the set of classes to which the instance belongs, where $\mathcal{Y} = \{y_1, \ldots, y_K\}$ is the set of the target classes.

We define as the *first dimension* (**D1**) of the hierarchical classification problem, whether it is single-label or multi-label. In the single-label case, which is the simplest, each instance i belongs in only one category (the cardinality of the set of labels $\mathbf{y}^i$ is 1), while in the multi-label case, an instance i might

belong in more than one category (the cardinality of the set of labels $\mathbf{y}^i$ is ≥ 1).



(a) Tree



(b) DAG

Fig. 1 A tree and a DAG class hierarchy.

In contrast to flat classification, where the classes are considered unrelated, in HC the classes are organized in taxonomies. The type of the taxonomy is the *second dimension* (**D2**) of hierarchical classification. It can be either a tree, in which case each node (classes) has a single parent, or directed acyclic graphs (DAGs), in which case nodes can have multiple parents; see Figures 1(a) and 1(b) respectively. In some cases, hierarchies may also be cyclic graphs. In all cases the hierarchy imposes a parent-child relation among the classes, which implies that an instance belonging to a specific class, *also belongs to all its ancestor classes*. A taxonomy is thus usually defined as a pair $(\mathcal{V}, \prec)$, where $\mathcal{V}$ is the set of all classes and $\prec$ is the `subclass-of` relationship with the following properties:[1]

- Asymmetry: if $v_i \prec v_j$ then $v_j \not\prec v_i$ for every v_i, $v_j \in \mathcal{V}$.
- Anti-reflexivity: $v_i \not\prec v_i$ for every $v_i \in \mathcal{V}$.
- Transitivity: if $v_i \prec v_j$ and $v_j \prec v_k$, then $v_i \prec v_k$ for every v_i, v_j, $v_k \in \mathcal{V}$.

In graphs with cycles, only the transitivity property holds. In this article we consider only hierarchies without cycles and we denote the descendants and ancestors of a class $v \in \mathcal{V}$ as $De(v)$ and $An(v)$, respectively. The parents of a class v are denoted as $Pa(v)$.

[1] Without loss of generality, we assume a `subclass-of` relationship among the classes, but in some cases a different relationship may hold, for example `part-of`. We assume, however, that the three properties always hold for the relationship.

Symbol	Description
$\mathbf{x}$	a feature vector
y	a class label
$De(v), An(v), Pa(v)$	descendants, ancestors and parents of class v
k_{ij}	cost of predicting class $\hat{y}_i$ instead of y_j
$\hat{Y}, Y$	sets of predicted and true classes
$\hat{Y}_{aug}, Y_{aug}$	augmented sets of predicted and true classes
$[b_u; c_u]$	capacity interval of an edge u of a flow network
$\alpha_p, \alpha_t, \beta_p, \beta_t$	capacity variables of the flow network
G	directed graph of flow network
E	edge set of flow network
E_H	edge set of hierarchy H
LCA	lowest common ancestor

Table 1 Description of symbols used throughout the article.

The *third dimension* (**D3**) concerns whether an instance can be classified only to a leaf of the hierarchy or to any class of the hierarchy. In the first case the problem is called as *mandatory leaf node prediction* [Silla and Freitas, 2011].

It is important to note that in most cases an instance is always classified in at least one class at the hierarchy. This is the situation that we consider in this paper. Table 1 describes the most frequent symbols used throughout this article.

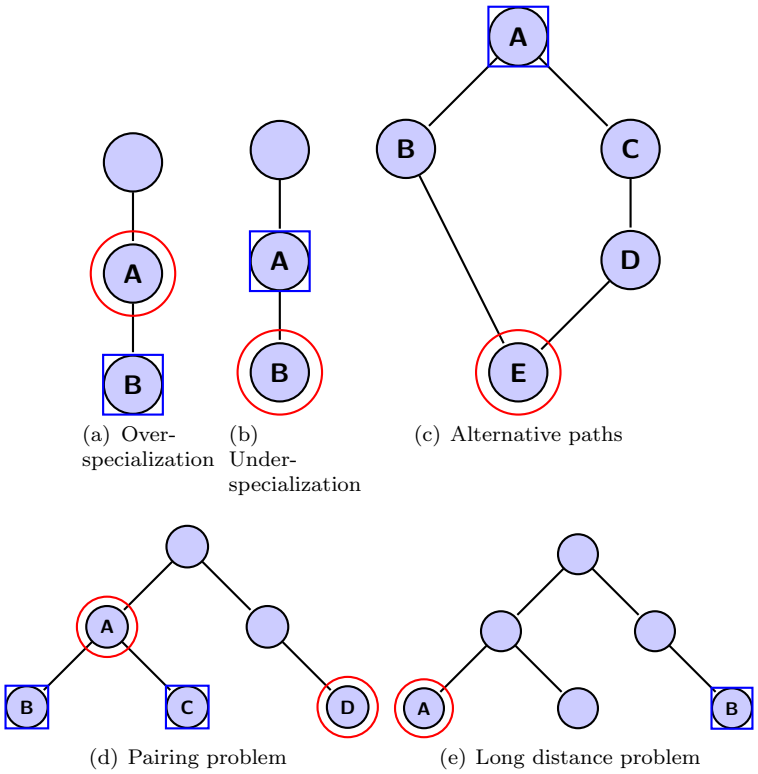
2.2 General Problems in Hierarchical Classification Evaluation

Although in many influential hierarchical classification papers [Koller and Sahami, 1997, McCallum et al., 1998] accuracy, precision, recall, F-measure, etc. are used for evaluation, these measures are not appropriate for HC, due to the relations that exist among the classes. A hierarchical performance measure should use the class hierarchy in order to evaluate properly HC algorithms. In particular, one must account for several different types of error according to the hierarchy. For example, consider the tree hierarchy in Figure 1(a). Assume that the true class for a test instance is `3.1` and that two different classification systems output `3` and `1` as the predicted classes. Using flat evaluation measures, both systems are punished equally, but the error of the second system is more severe as it makes a prediction in a different and unrelated sub-tree.

In order to measure the severity of an error in hierarchical classification, there are several interesting issues that need to be addressed. Figure 2 presents five cases that require special handling. In all cases, the nodes surrounded by circles are the true classes, while the nodes surrounded by rectangles are the predicted ones. These cases can be sub-grouped in (i) *pairing* problems (Figures 2(d) and 2(e)), where one must decide which pairs of predicted and true classes to take into account for the calculation of the error, and (ii) *distance-measuring* problems (Figures 2(c), 2(a) and 2(b)), which concern the way that the error will be calculated for a pair of predicted and true classes.

The first two *distance-measuring* problems are linked with dimension **D3**. They appear only when classification to inner nodes is allowed. Figure 2(a) presents an *over-specialization* error where the predicted class is a descendant of the true class. Figure 2(b) depicts an *under-specialization* error, where an ancestor of the true class is selected. In both cases the desired behaviour of the measure would be to reduce the penalty of the classification system, according to the distance between the true class and the predicted one.

The third case (Figure 2(c)), called *alternative paths*, presents a scenario where there are two different ways to reach the true class starting from a predicted class. In this case, a measure could use one of the two paths or both in order to evaluate the performance of the classification system. Selecting the path that minimizes the distance between the two classes and using that as a measure of error seems reasonable. In Figure 2(c) the predicted class is an ancestor of the true class, but an alternative paths case may also involve multiple paths from an ancestor to a descendant predicted class. This case appears only in DAG taxonomies and in that way it is linked to the **D2** dimension of the hierarchical classification problem.



(a) Over-specialization (b) Under-specialization (c) Alternative paths

(d) Pairing problem (e) Long distance problem

Fig. 2 Interesting cases when evaluating hierarchical classifiers. Nodes surrounded by circles are the true classes while the nodes surrounded by rectangles are the predicted classes.

Figure 2(d) presents a scenario linked with the **D1** dimension of the hierarchical classification problem: It can only occur in multi-label settings. In this case one must decide, before even measuring the error, which pairs of true and predicted classes should be compared. For example, node `A` (true class) could be compared to `B` (predicted) and `D` to `C`; or node `A` could be compared to both `B` and `C`, and node `D` to none; other pairings are also possible. Depending on the pairings, the score assigned to the classifier will be different. It seems reasonable to use the pairings that minimize the classification error. For example, in Figure 2(d) it could be argued that the predictions of `B` and `C` are based on evidence about `A` and thus both `B` and `C` should be compared to A.

Finally, Figure 2(e) presents a case where the predicted class should probably not be matched to any true class. This is typically the case when the predicted class and the true class are too distant, which is why we call this case the *long distance problem*.

The simplest case scenario is when we have a single-label classification problem, with a tree hierarchy and where instances are only allowed to be classified to leaves. The first four problems do not appear in this scenario and, hence, all the existing hierarchical evaluation measures have a similar behaviour in this case. If we alter any dimension of the problem, one has to deal with some of the above cases and existing measures start behaving differently. Only the final problem (2(e)) appears in every combination of the dimensions of the problem.

We now turn to the two main families of HC evaluation measures, namely pair-based measures and set-based measures.

2.3 Pair-based Measures

Pair-based measures assign costs to pairs of predicted and true classes. For example, in Figure 2(d) class `B` could be paired with `A` and class `C` with `D`, and then the sum of the corresponding costs would give the total misclassification error.

Let $\hat{Y} = \{\hat{y}_i | i = 1 \dots M\}$ and $Y = \{y_j | j = 1 \dots N\}$ be the sets of the predicted and true classes respectively, for a single test instance (the index of the instance is omitted for simplicity). The sets Y and $\hat{Y}$ are augmented with a default predicted and a default true class, denoted respectively $\hat{y}_{M+1}$ and y_{N+1}. The default classes are used when a predicted class cannot or should not be paired to any true class and vice-versa. For example, when the distances between a predicted class $\hat{y}_i$ and all the true classes y_i exceed a predefined threshold (see the long distance problem in Figure 2(e)), the predicted class $\hat{y}_i$ may be paired with the default true class.

Additionally, let κ_{ij} be the cost of predicting class $\hat{y}_i$ instead of the true class y_j. The matrix $\mathbf{K} = [\kappa_{ij}]_{i=1\dots M+1, j=1\dots N+1}$, $\kappa_{ij} \geq 0, \forall i, j$ contains the costs of all possible pairs of predicted and true classes, including the default classes.

Pair-based measures typically calculate the cost κ_{ij} of a pair of a predicted class $\hat{y}_i$ and a true class y_i as the minimum distance of $\hat{y}_i$ and y_i in the hierarchy, e.g. as the number of edges between the classes along the shortest path that connects them. The intuition is that the closer the two classes are in the hierarchy, the more similar they are, and therefore the less severe the error. More elaborate cost measures may assign weights to the hierarchy's edges, and the weights may decrease when moving from the top to the bottom [Blockeel et al., 2002, Holden and Freitas, 2006]. The distance to the default classes is usually set to a fixed large value.

In a spirit of fairness (minimum penalty), the aim of an evaluation measure is to pair the classes returned by a system and the true classes in a way that minimizes the overall classification error. This can be formulated as the following optimization problem:

Problem 1.

$$\begin{cases} \min\limits_{\substack{l_{ij}, \\ 1\leq i\leq M+1, \\ 1\leq j\leq N+1}} \sum\limits_{\substack{i=1\ldots(M+1), \\ j=1\ldots(N+1)}} \kappa_{ij} l_{ij} \\ \textit{subject to:} \\ \quad \textit{(i)}\ \forall i = 1\ldots(M+1), \forall j = 1\ldots(N+1), l_{ij} \in \{0,1\}; \\ \qquad l_{(M+1)(N+1)} = 0 \\ \quad \textit{(ii)}\ \alpha_p \leq \sum_{j=1}^{N+1} l_{ij} \leq \beta_p, \forall i = 1\ldots M \\ \quad \textit{(iii)}\ \alpha_t \leq \sum_{i=1}^{M+1} l_{ij} \leq \beta_t, \forall j = 1\ldots N \end{cases}$$

Constraint (i) states that l_{ij}, which denotes the alignment between classes, is either 0 (classes $\hat{y}_i$ and y_j are not paired) or 1 (classes $\hat{y}_i$ and y_j are paired); it furthermore states that the default predicted and true classes cannot be aligned (these default classes are solely used to "collect" those predicted and true classes with no counterpart). The parameters α_p, $\beta_p \in \mathbb{N}$ (constraint (ii)) are the lower and upper bounds of the allowed number of true classes that a predicted class can be paired with. For example, setting $\alpha_p = \beta_p = 1$ requires each predicted class to be paired with exactly one true class. Similarly, the parameters α_t, $\beta_t \in \mathbb{N}$ (constraint (iii)) limit the number of predicted classes that a true class can be paired with. The above constraints directly imply that $\forall i = 1\ldots M, l_{iN+1} \leq \beta_p$ and $\forall j = 1\ldots N, l_{M+1j} \leq \beta_t$, which in turn implies that the default true class can be aligned to at most $\beta_p M$ predicted classes and the default predicted class to at most $\beta_t N$ true classes.

Problem 1 corresponds to a best pairing problem in a bipartite graph, with the nodes of the two types standing for predicted and true classes, respectively. It is important to note here that the pairing we are looking for is not a 1-1 matching, since the same node of one type can be paired with several nodes of the other type. We opt to approach Problem 1 as a graph pairing one rather than an integer linear programming one, for two reasons: first because there exist simple polynomial solutions to pairing problems in graphs, and second because the graph framework allows one to easily illustrate how the different

cost-based measures proposed so far relate to each other. In particular, we model Problem 1 as a cost flow minimization problem [Ahuja et al., 1993].

2.3.1 A Flow Network Model for Class Pairing

A flow network is a directed graph $G = (V, E)$ with m edges, where each edge $u \in E$ is associated with a lower and an upper capacity denoted b_u and c_u respectively. The flow along an edge u is denoted as ϕ_u and $b_u \leq \phi_u \leq c_u$. The flow of the network is a vector $\phi = (\phi_1, \phi_2, \ldots, \phi_m)^T \in \mathbb{R}^m$. For each vertex $i \in V$, the flow conservation property holds:

$$\sum_{u \in \omega^+(i)} \phi_u = \sum_{u \in \omega^-(i)} \phi_u$$

where $\omega^+(i)$ and $\omega^-(i)$ denote the set of edges entering and leaving vertex i respectively. Each edge u is also associated with a cost κ_u which represents the cost of using this edge. The total cost of a flow ϕ is:

$$\kappa^T \phi = \sum_{u \in E} \kappa_u \phi_u,$$

where $\kappa = (\kappa_1, \kappa_2, \ldots, \kappa_m)^T \in \mathbb{R}^m$. The minimum cost flow is the one that minimizes $\kappa^T \phi$ while satisfying the capacity and flow conservation constraints. The quantity to be minimized in flow networks is the same as the one in Problem 1, the constraints in this latter problem corresponding to capacity constraints, as explained below. Furthermore, the following integrality theorem states that when the bounds of capacity intervals are integers, there exists a minimal cost flow such that the quantity of flow on each edge is also an integer. Furthermore, all standard algorithms for finding minimal cost flows guarantee to find this particular flow [Ahuja et al., 1993].

Integrality Theorem. *If a flow network has capacities which are all integer valued and there exists some feasible flow in the network, then there is a minimum cost feasible flow with an integer valued flow on every arc.*

Pairing problems in bipartite graphs are represented with flow networks by adding two nodes, a source and a sink, and edges from the source to the first set of nodes, from the second set of nodes to the sink, and from the sink to the source. These extra nodes and edges ensure that the flow conservation constraints are satisfied. For pair-based measures, one thus obtains the following flow network framework $G(V, E)$ (see also Figure 3):

- V includes a source, a sink, the predicted classes, the true classes, a default true class and a default predicted class;
- E includes edges from the source to all the predicted classes (including the default predicted class), from every predicted class to every true class (including the default true class), from every true class to the sink and from the sink to the source.

No edges exist between the default predicted and default true class, as required by constraint (i) above.

In our setting, the capacity intervals $[b_u; c_u]$ of the edges u express the possible number of pairs that each predicted or true class can participate in. In the solved flow network the flow values will reflect the specific evaluation measure as these values indicate the solution with the minimum cost. Reflecting constraints of Problem 1, the capacity intervals are defined as follows:

- From each predicted class $\hat{y_j}$ to each true class y_i, excluding the default class, the capacity interval is [0;1]; the integrality theorem here implies that the flow value between predicted and true classes will be either 0 or 1, i.e. a predicted and a true class either be paired (1) or not paired (0). The capacity bounds here correspond to the l_{ij} values of Problem 1 (constraint (i)).
- From the source to a (non-default) predicted class, the capacity interval is $[\alpha_p; \beta_p]$ meaning that a predicted class is aligned with at least α_p and at most β_p true classes.
- Similarly, from a (non-default) true class to the sink, the capacity interval is $[\alpha_t;\beta_t]$ meaning that a true class is aligned with at least α_t and at most β_t predicted classes;
- From each predicted class $\hat{y_j}$ to the default true class the capacity interval is $[0;\beta_p]$ and from the default predicted class to each true class the capacity interval is $[0;\beta_t]$; from the source (resp. sink) to the default predicted (resp. true) class, the capacity interval is $[0; \beta_t N]$ (resp. $[0; \beta_p M]$), reflecting the fact that the default predicted (resp. true) class can be mapped to at most $\beta_t N$ true classes (resp. $\beta_p M$ predicted classes).
- Lastly, from the sink to the source, the capacity interval is $[\alpha_p M;\beta_t N + \beta_p M]$, which corresponds to a loose setting compatible with the intervals given above; this last capacity interval does not impose any constraint, but is necessary to ensure flow conservation.

2.3.2 Existing Pair-based Measures

The majority of the existing pair-based measures deals only with tree hierarchies and single-label problems. Under these conditions the pairing problem becomes simple, because a single path exists between the predicted and the true classes of each classified item. The complexity of the problem increases when the hierarchy is a DAG or when the problem is multi-labeled; current measures cannot handle the majority of the phenomena presented in Section 2.2.

In the simplest case of pair-based measures [Dekel et al., 2004, Holden and Freitas, 2006, Xiao et al., 2011], the measure trivially pairs the single prediction with the single true label ($M = N = 1$), so that $\alpha_p = \beta_p = \alpha_t = \beta_t = 1$. Note that no default classes exist in this measure, or equivalently the corresponding costs are equal to infinity, $\kappa(DP, y) = \kappa(DT, \hat{y}) = +\infty$

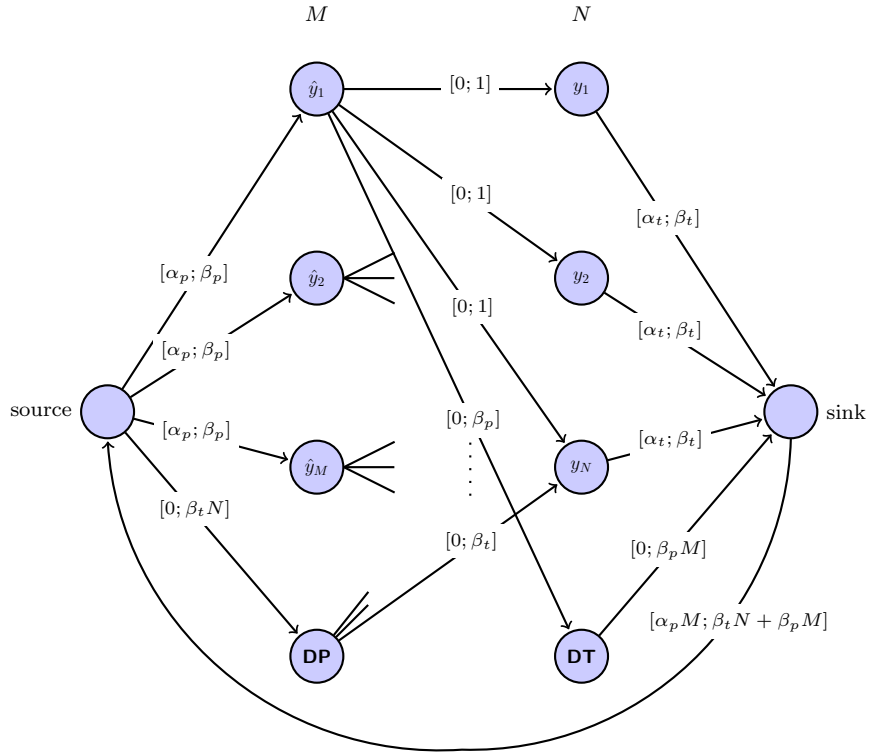


Fig. 3 General form of the proposed flow network.

For a pair $(\hat{y_j}, y_i)$ of a predicted and a true class κ_{ij} is taken to be the distance between y_i and $\hat{y_j}$:

$$\kappa_{ij} = \sum_{e \in E_H(i,j)} w_e, \tag{1}$$

where $E_H(i,j)$ is the set of edges along the path from y_i to $\hat{y_j}$ in the hierarchy and w_e is the weight of edge e. For $w_e = 1$, we get what Dekel et al. [2004] call *tree induced error*.

In [Sun and Lim, 2001] two cost measures are proposed for multi-label problems in tree hierarchies, where all possible pairs of the predicted and true classes are used in the calculation. In this case, $\alpha_p = \beta_p = N$ and $\alpha_t = \beta_t = M$. Again, no default classes are used and so the corresponding costs are: $\kappa(DP, y_i) = \kappa(DT, \hat{y_j}) = +\infty,\ i = 1 \ldots N, j = 1 \ldots M$. Note that this is an extreme case, where all pairs of predicted and true labels are used. The weights w_e are calculated in two alternative ways: a) as the similarity (e.g., cosine similarity between the gravity centres of the example vectors of the two classes) between the classes of the predicted and true ones, and b) using the distances of the hierarchy as in Equation 1.

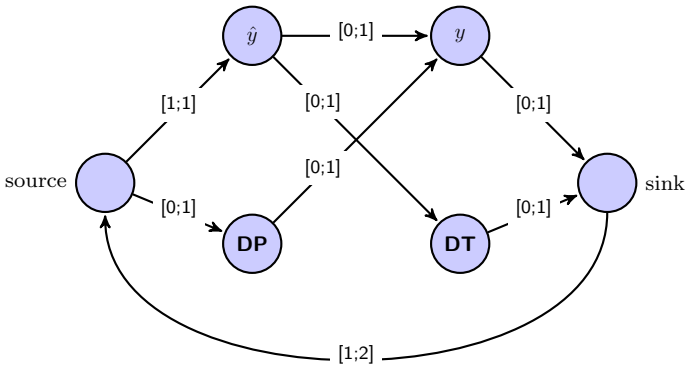


Fig. 4 Tree Induced Error flow network.

A measure dubbed *Graph Induced Error* (GIE) was proposed and used during the second Large Scale Hierarchical Text classification challenge (LSHTC).[2] GIE is based on the best matching pairs of predicted and true classes and can handle multi-label (and single-labeled) classification with both tree and DAG class hierarchies. For a particular instance being classified, each predicted class is paired either with exactly one true class or with the default true class; multiple predicted classes can be paired with the default true class (Figure 5). Similarly, each true class is paired with exactly one predicted class or with the default predicted class, and several true classes can be paired with the default predicted class. Hence, $\alpha_p = \beta_p = \alpha_t = \beta_t = 1$. The cost κ_{ij} is computed as in Equation 1, with $w_e = 1, \forall e$. If the hierarchy is a DAG, multiple hierarchy paths may link each predicted class $\hat{y}_j$ to its paired true class y_i; then $E_H(i, j)$ is taken to be the shortest of these paths. The cost of pairing a class (predicted or true) with a default one is set to a positive value $D_{\max}$. Figure 5 presents the corresponding flow network.

In multi-label classification GIE's concept of "best" matching fails to address the pairing problem of Section 2.2. For example, if a predicted class has two true classes as children, as in Figure 2(d), then only one of them would be paired with its parent. The other one would either be penalized with $D_{\max}$ or would be paired with another distant class.

2.3.3 Multi-label Graph Induced Accuracy

We propose here a straightforward extension of GIE called *Multi-label Graph Induced Accuracy* (MGIA), in which each class is allowed to participate in more than one pair. This extension makes the method more suitable to the pairing problem. Figure 6 presents the MGIA flow network, in which $\alpha_p = \alpha_t = 1$, $\beta_p = N$, $\beta_t = M$, reflecting the fact that each predicted (resp. true) class

[2] http://lshtc.iit.demokritos.gr/

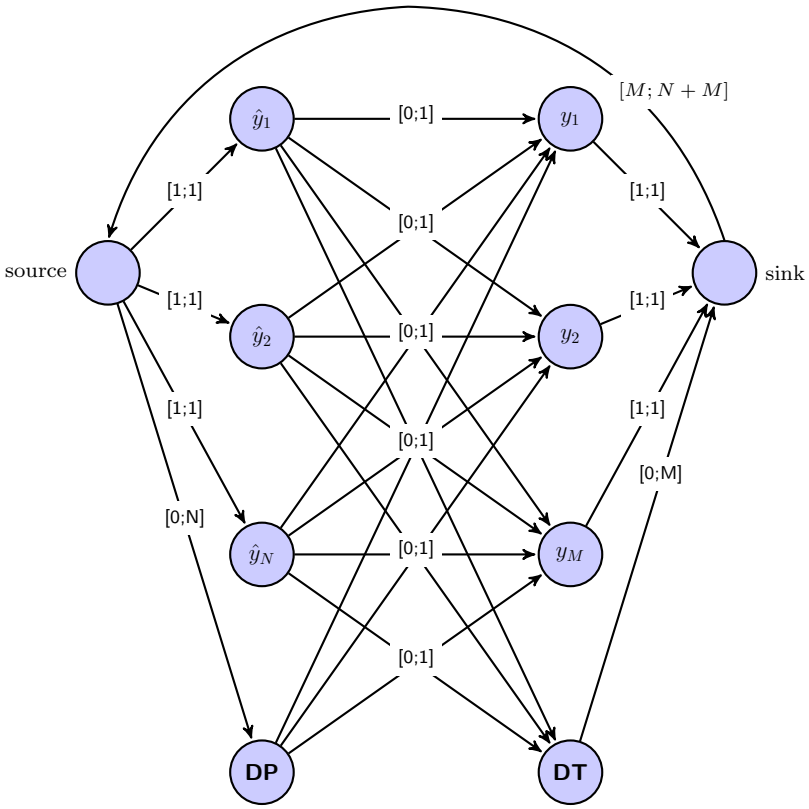


Fig. 5 Graph Induced Error (GIE) flow network.

must be paired with at least one and at most $\beta_p = N$ (resp. $\beta_t = M$) true (resp. predicted) classes. The cost of pairing a class (predicted or true) with a default one is set as in GIE. Solving the flow network optimization problem is easy since the only constraints are that the default predicted class cannot be paired with the default true class and that categories of the same set (predicted or true) cannot be paired to each other. Thus each pairing can be solved separately from the others by pairing a class with either the default class of the other set, or the nearest class of the other set. An alternative extension is to consider that only true (or predicted) classes can be paired to more than one class. We do not consider these alternatives here even though we believe they can be useful in certain settings.

As in the previous pair-based measures, after solving the problem the corresponding minimum cost is the error of the evaluation measure. Instead of using directly this error for evaluation, we define an accuracy based measure as follows:

$$1 - \frac{fnerror}{|(P \cup T) \setminus (P \cap T)| \cdot D_{max}}$$

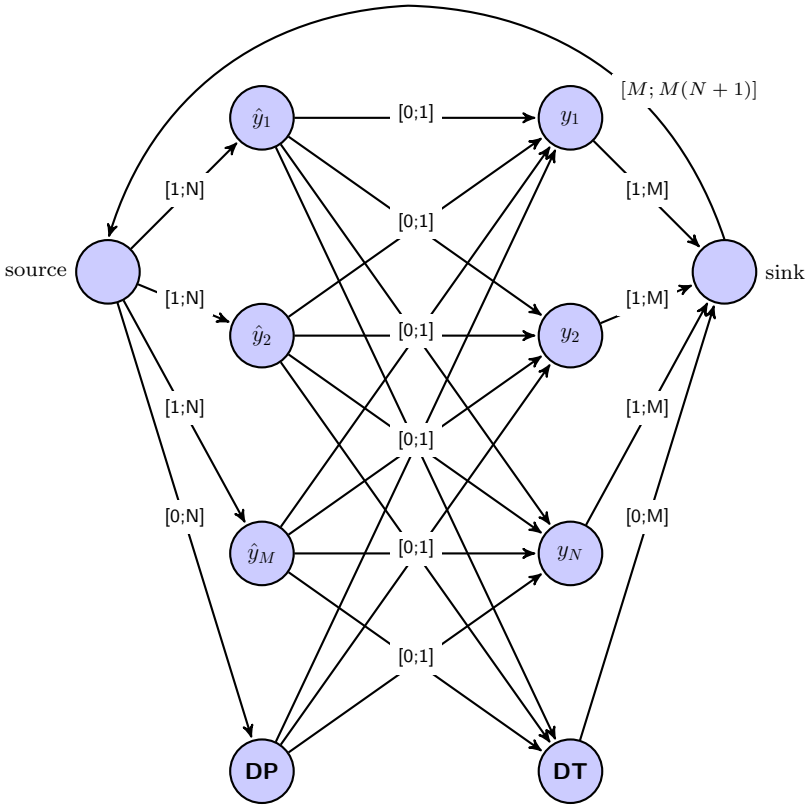


Fig. 6 Multi-label Graph Induced Accuracy flow network.

where $fnerror$ is the value provided by the solved flow network.

The above measure is bounded in [0,1] and the better a system is the closer it will be to 1. Note that in the case where all predicted classes and all true classes are paired with the respective default classes, *fnerror* will reach its maximum value $(|P| + |T|) \cdot D_{max}$ and will be equal to the denominator, as $P \cap T = \emptyset$ resulting in a value of 0. Essentially, the advantage of the proposed measure over other pair-based measures is that it takes into account the correct predictions of the classification system (that is the true positives, $P \cap T$).

2.4 Set-based Measures

The performance measures of this category are based on operations on the entire sets of predicted and true classes, possibly including also their ancestors and descendants, as opposed to pair-based measures, which consider only pairs of predicted and true classes.

Set-based measures have two distinct phases:

1. The augmentation of Y and $\hat{Y}$ with information about the hierarchy.
2. The calculation of a cost measure based on the augmented sets.

The augmentation of Y and $\hat{Y}$ is a crucial step, attempting to capture the hierarchical relations of the classes. For example, the sets may be augmented with the ancestors of the true and predicted classes as follows:

$$Y_{aug} = Y \cup An(y_1) \cup \ldots \cup An(y_N) \tag{2}$$

$$\hat{Y}_{aug} = \hat{Y} \cup An(\hat{y}_1) \cup \ldots \cup An(\hat{y}_M) \tag{3}$$

Using the augmented sets of predicted and true classes, two approaches have mainly been adopted to calculate the misclassification cost: a) symmetric difference loss and b) hierarchical precision and recall.

Symmetric difference loss is calculated as follows, where $|S|$ denotes the cardinality of a set S:

$$l_{\Delta}(Y_{aug}, \hat{Y}_{aug}) = |(\hat{Y}_{aug} \setminus Y_{aug}) \cup (Y_{aug} \setminus \hat{Y}_{aug})|$$

If we use the initial $\hat{Y}$ and Y sets instead of $\hat{Y}_{aug}, Y_{aug}$, the measure becomes the standard symmetric difference for flat multi-label classification. In addition, the two quantities of the symmetric loss difference are the false positives $(\hat{Y}_{aug} \setminus Y_{aug})$ and the false negatives $(Y_{aug} \setminus \hat{Y}_{aug})$ respectively.

Hierarchical precision and recall are defined as follows:

$$P_H = \frac{|\hat{Y}_{aug} \cap Y_{aug}|}{|\hat{Y}_{aug}|}$$

$$R_H = \frac{|\hat{Y}_{aug} \cap Y_{aug}|}{|Y_{aug}|}$$

Set-based measures are not affected by the pairing problem of Figure 2(d) and the long distance problem of Figure 2(e), as they do not rely on pairings of true and predicted classes.

2.4.1 Existing Set-based Measures

Different measures differ mainly in the way the sets of predicted and true classes are augmented. In [Kiritchenko et al., 2005, Struyf et al., 2005, Cai and Hofmann, 2007] the ancestors of the predicted and true classes are added to Y_{aug} and $\hat{Y}_{aug}$, as in Equations 2 and 3 above. Alternatively, in [Ipeirotis et al., 2001] the descendants of the true and predicted classes are added:

$$Y_{aug} = Y \cup De(y_1) \cup \ldots \cup De(y_N)$$

$$\hat{Y}_{aug} = \hat{Y} \cup De(\hat{y}_1) \cup \ldots \cup De(\hat{y}_M)$$

In the latter approach, when the true and predicted classes are in different sub-graphs of the hierarchy (different sub-trees, if the hierarchy is a tree), a larger penalty is given, compared to the case where Eq. 2 and 3 are used.

In [Cesa-Bianchi et al., 2006], the approach that adds the ancestors is adopted (Eq. 2 and 3), but the augmented sets are then altered as follows:

$$Y_{aug} \leftarrow Y_{aug} \setminus \left\{ y_k \;\middle|\; y_k \in Y_{aug} \wedge y_k \notin \hat{Y}_{aug} \wedge Pa(y_k) \notin \hat{Y}_{aug} \right\} \tag{4}$$

$$\hat{Y}_{aug} \leftarrow \hat{Y}_{aug} \setminus \left\{ \hat{y}_k \;\middle|\; \hat{y}_k \in \hat{Y}_{aug} \wedge \hat{y}_k \notin Y_{aug} \wedge Pa(\hat{y}_k) \notin Y_{aug} \right\} \tag{5}$$

Equation 5 introduces some tolerance to over-specialization. Consider, for example, Figure 7(a) where we assume that the only true class is A and the only predicted class is C. According to Equation 3, one adds class B and class A to $\hat{Y}_{aug}$. Based on Equation 5, one then removes C from $\hat{Y}_{aug}$ to avoid penalizing the classification method for both B and C. Similarly, with Equation 4, one tolerates under-classification. In Figure 7(b) the only true class is C and the only predicted class is A. According to Equation 2, classes B and A are added to Y_{aug}. Based on Equation 4, one removes C from $\hat{Y}_{aug}$ to avoid penalizing the classification method for both B and C. The drawback of this measure is that it tends to favour systems that limit their predictions to high levels of the hierarchy.

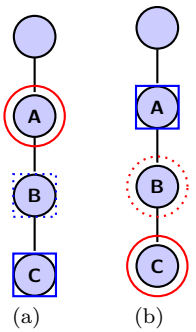
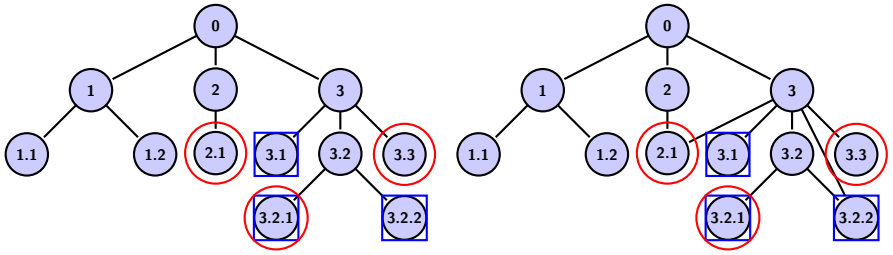


Fig. 7 Tolerating over-specialization and under-specialization.

2.4.2 Lowest Common Ancestor Precision, Recall and F_1 Measures

The set-based measure proposed in this paper is based on the hierarchical versions of precision, recall and F_1, that adds all the ancestors of the predicted and true classes to Y_{aug} and $\hat{Y}_{aug}$. However, adding all the ancestors has the undesirable effect of over-penalizing errors that happen to nodes with many ancestors. To an attempt to address this issue, we propose the Lowest Common Ancestor Precision (P_{LCA}), Recall (R_{LCA}) and F_1 (F_{LCA}) measures. These measures use the concept of the lowest common ancestor (LCA) as defined in graph theory [Aho et al., 1973].

Definition 1. *The lowest common ancestor* $LCA(n_1, n_2)$ *of two nodes* n_1 *and* n_2 *of a tree* T *is defined as the lowest node in* T *(furthest from the root) that is an ancestor of both* n_1 *and* n_2*.*



(a) Tree (b) DAG

Fig. 8 Tree and DAG hierarchies. Nodes surrounded by circles are true classes. Nodes surrounded by rectangles are predicted classes.

For example, in Figure 8(a) $LCA(3.1, 3.2.2) = 3$. In the case of a DAG the definition of LCA changes. $LCA(n_1, n_2)$ is a set of nodes (instead of a single node), since it is possible for two nodes to have more than one LCA. Furthermore, the LCA may not necessarily be the node that is furthest from the root and we need to rely on the shortest path between them.

Definition 2. *Given a set* $p_{all}(n_1, n_2)$ *containing all paths that connect nodes* n_1 *and* n_2*, we define* $path_{min}(n_1, n_2)$ *as the subset of* $p_{all}(n_1, n_2)$ *for which:* $\forall p \in path_{min}(n_1, n_2); \nexists p' \in p_{all}(n_1, n_2) : cost(p') < cost(p)$*.*

where the cost of a path corresponds to its length, when the edges of the hierarchy are unweighted and the cost of a set $path_{min}(n_1, n_2)$ is the cost of any path in the set since all of them are of the same length.

For example in Figure 8(b):

- $path_{min}(2.1, 3.1) = \{\{2.1, 3, 3.1\}\}$
- $path_{min}(2.1, 3.2.2) = \{\{2.1, 3, 3.2.2\}\}$
- $path_{min}(3.2.2, 3.2.1) = \{\{3.2.2, 3.2, 3.2.1\}\}$

It is worth noting that in the general case $path_{min}(n_1, n_2)$ may contain multiple paths, not just one. Using the definition of $path_{min}(n_1, n_2)$, we define the $LCA(n_1, n_2)$ of two nodes n_1 and n_2 of a DAG D as follows:

Definition 3. *The lowest common ancestor* $LCA(n_1, n_2)$ *of two nodes* n_1 *and* n_2 *of a DAG* D *is defined as the set of nodes* S *for which:* $\forall p \in path_{min}(n_1, n_2); \exists n \in S \cap p \wedge \nexists n' \in p : n'$ *is closer to the root than* n*.*

In multi-label classification, it is necessary to extend the definition of LCA to compare a node n (e.g. a true class) against a set of nodes S (e.g. the predicted classes), as follows:

Definition 4. *The $LCA(n, S)$ of a node n and a set of nodes S is the set of all the lowest common ancestors $LCA(n, i)$ for each $i \in S_{best}(n, S) \subseteq S$, where $S_{best}(n, S)$ contains the closest to n nodes of S ($S_{best}(n, S) = \{i \in S : \nexists j \in S, j \neq i \wedge cost(path_{min}(n, i)) > cost(path_{min}(n, j))\}$).*

For example, in Figure 8(b) $S_{best}(3.1, \{2.1, 3.3, 3.2.1\}) = \{2.1, 3.3\}$ and $LCA(3.1, \{2.1, 3.3, 3.2.1\})$ is $\{3\}$.

Given this definitions and assuming again that Y and $\hat{Y}$ are the set of predicted and true classes, respectively, of an instance, we compute the $LCA(y, \hat{Y})$ of each element y of Y. Similarly for each element $\hat{y}$ of $\hat{Y}$, we compute $LCA(\hat{y}, Y)$. Using Figure 8(b), let $Y = \{2.1, 3.2.1, 3.3\}$ and $\hat{Y} = \{3.1, 3.2.1, 3.2.2\}$. Then:

- $LCA(2.1, \hat{Y}) = \{3\}$, connecting 2.1 with either 3.1 using $path_{min}(2.1, 3.1)$ or 3.2.1 using $path_{min}(2.1, 3.2.1)$ or 3.2.2 using $path_{min}(2.1, 3.2.2)$.
- $LCA(3.3, \hat{Y}) = \{3\}$, connecting 3.3 with either 3.1 using $path_{min}(3.3, 3.1)$ or 3.2.1 using $path_{min}(3.3, 3.2.1)$ or 3.2.2 using $path_{min}(3.3, 3.2.2)$.
- $LCA(3.2.1, \hat{Y}) = \{3.2.1\}$, connecting 3.2.1 with itself.
- $LCA(3.2.1, Y) = \{3.2.1\}$, connecting 3.2.1 with itself.
- $LCA(3.1, Y) = \{3\}$, connecting 3.1 with either 2.1 using $path_{min}(3.1, 2.1)$ or 3.3 using $path_{min}(3.1, 3.3)$.
- $LCA(3.2.2, Y) = \{3.2, 3\}$, the first connecting 3.2.2 with 3.2.1 using $path_{min}(3.2.2, 3.2.1)$ and the second connecting 3.2.2 with either 2.1 using $path_{min}(3.2.2, 2.1)$ or 3.3 using $path_{min}(3.2.2, 3.3)$.

Additionally, we are interested in the union of all $\mathrm{LCA}(y, \hat{Y})$ for all $y \in Y$ and the union of all $\mathrm{LCA}(\hat{y}, Y)$ for all $\hat{y} \in \hat{Y}$.

Definition 5. *Given a set of true classes (nodes) Y and a set of predicted classes (nodes) $\hat{Y}$, we define $LCA_{all}(Y, \hat{Y})$ as the union of all $LCA(y, \hat{Y})$ for all $y \in Y$. Similarly we define $LCA_{all}(\hat{Y}, Y)$ as the union of all $LCA(\hat{y}, Y)$ for all $\hat{y} \in \hat{Y}$.*

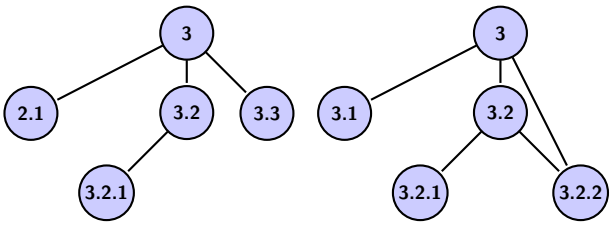
In the above example, $LCA_{all}(Y, \hat{Y})$ =$\{3,\ 3.2.1\}$, $LCA_{all}(\hat{Y}, Y)$ =$\{3,\ 3.2,\ 3.2.1\}$. From the above sets, one can then define the subgraph relating to sets of edges:

Definition 6. *Given a set of true classes (nodes) Y, a set of predicted classes (nodes) $\hat{Y}$ and a set of LCA_{all} ($LCA_{all}(Y, \hat{Y}) \cup LCA_{all}(\hat{Y}, Y)$), we define $G_t^{ex}(Y, \hat{Y})$ as the graph constructed by:*

- *all $path_{min}(y, a)$: $y \in Y \wedge a \in LCA(y, \hat{Y})$*
- *all $paths(y, a)$ subpaths of each path of $paths_{min}(\hat{y}, y)$: $\hat{y} \in \hat{Y} \wedge y \in S_{best}(\hat{y}, Y) \wedge a \in LCA(\hat{y}, Y)$*

Similarly $G_p^{ex}(Y, \hat{Y})$ is the graph that contains:

- *all $path_{min}(\hat{y}, b)$: $\hat{y} \in \hat{Y} \wedge b \in LCA(\hat{y}, Y)$*
- *all $paths(\hat{y}, b)$ subpaths of each path of $paths_{min}(y, \hat{y})$: $y \in Y \wedge \hat{y} \in S_{best}(y, \hat{Y}) \wedge b \in LCA(y, \hat{Y})$*



(a) $G_t^{ex}(Y, \hat{Y})$ (b) $G_p^{ex}(Y, \hat{Y})$

Fig. 9 The nodes of $G_t^{ex}(Y, \hat{Y})$ and $G_p^{ex}(Y, \hat{Y})$ constitute the augmented sets of the true and the predicted classes with $LCAs$.

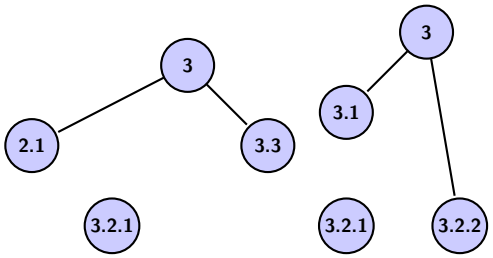
For example, for the Y and $\hat{Y}$ of figure 8(b) we get the $G_t^{ex}(Y, \hat{Y})$ and $G_p^{ex}(Y, \hat{Y})$ graphs of figure 9(a) and 9(b), respectively.

Based on these graphs, the true and predicted sets of classes are augmented, in order for the set-based measures to be calculated. In the case of Figure 9, $Y_{aug} = \{3, 2.1, 3.2, 3.3, 3.2.1\}$ and $\hat{Y}_{aug} = \{3, 3.1, 3.2, 3.2.1, 3.2.2\}$. The next step is to calculate cost measures based on these two sets which in our case are the following:

$$P_{LCA} = \frac{|\hat{Y}_{aug} \cap Y_{aug}|}{|\hat{Y}_{aug}|}, \; R_{LCA} = \frac{|\hat{Y}_{aug} \cap Y_{aug}|}{|Y_{aug}|}, \; F_{LCA} = \frac{2P_{LCA}R_{LCA}}{P_{LCA} + R_{LCA}}$$

In the example of Figure 9, all three measures, P_{LCA} (precision), R_{LCA} (recall) and F_{LCA} (F-measure), between sets Y_{aug} and $\hat{Y}_{aug}$, are 0.6. We prefer this approach over the symmetric difference loss (see Section 2.4), which computes the union of $(\hat{Y}_{aug} \setminus Y_{aug})$ and $(Y_{aug} \setminus \hat{Y}_{aug})$. By computing $(\hat{Y}_{aug} \setminus Y_{aug})$ and $(Y_{aug} \setminus \hat{Y}_{aug})$, one effectively takes into account only the mistakes that the evaluated system made, similar to counting the false possitives (FP) and false negatives (FN), while ignoring the true positives (TP). The inclusion of TP in the calculation of recall and precision measures leads to a special treatment of the possitive class, as compared to the negative one (true negatives are ignored), which is typically also much larger. Therefore, by ignoring TP, an evaluation measure tends to favor systems that do well in the larger class (typically the negative one). In multi-label classification, these systems are the most "conservative" ones that predict fewer classes. This is an undesirable effect.

The two graphs $G_t^{ex}(Y, \hat{Y})$ and $G_p^{ex}(Y, \hat{Y})$ are created using all nodes of $LCA_{all}(Y, \hat{Y})$ and $LCA_{all}(\hat{Y}, Y)$ and all corresponding paths. However, one can select subgraphs of the two graphs $G_t(Y, \hat{Y}) \subseteq G_t^{ex}(Y, \hat{Y})$ and $G_p(Y, \hat{Y}) \subseteq G_p^{ex}(Y, \hat{Y})$ connecting each node of $Y \cup \hat{Y}$ with an LCA. For example, in Figure 8(b) node 3.2.2 has two LCAs, node 3.2 and 3. Node 3.2 can be removed from $G_t^{ex}(Y, \hat{Y})$ and $G_p^{ex}(Y, \hat{Y})$ We would then get graphs $G_t(Y, \hat{Y})$ and $G_p(Y, \hat{Y})$ of Figure 10. P_{LCA}, R_{LCA} and F_{LCA}, between the reduced sets Y_{aug} and $\hat{Y}_{aug}$ of Figure 10, are 0.5 instead of 0.6 (Figure 9). In other words graphs $G_t(Y, \hat{Y})$



(a) $G_t(Y,\hat{Y})$ (b) $G_p(Y,\hat{Y})$

Fig. 10 $G_t(Y,\hat{Y})$ and $G_p(Y,\hat{Y})$ are the minimal graphs required in order to create the augmented sets of the true and the predicted classes with *LCAs*.

and $G_p(Y,\hat{Y})$ should comprise the nodes necessary for connecting the two sets, through their LCAs. As redundant nodes can lead to fluctuations in P_{LCA}, R_{LCA} and F_{LCA}, they should be removed. In order to obtain the minimal LCA graphs, one has to solve the following maximization problem:

Problem 2. *Minimal LCA graph extension.*

$$
\begin{cases}
\underset{\substack{G_t(Y,\hat{Y})\subseteq G_t^{ex}(Y,\hat{Y}),\\ G_p(Y,\hat{Y})\subseteq G_p^{ex}(Y,\hat{Y})}}{\text{argmax}} \quad F_{LCA}(G_t(Y,\hat{Y}),G_p(Y,\hat{Y})) \\
\textit{subject to:} \\
\quad \textit{(i)}\ \forall y\in Y; y\in G_t(Y,\hat{Y})\ \textit{and}\ \forall \hat{y}\in\hat{Y}; \hat{y}\in G_p(Y,\hat{Y}) \\
\quad \textit{(ii)}\ \forall y\in Y; \exists a: a\in G_t(Y,\hat{Y})\wedge a\in G_p(Y,\hat{Y})\wedge a\in LCA(y,\hat{Y}) \\
\qquad \forall \hat{y}\in\hat{Y}; \exists b: b\in G_t(Y,\hat{Y})\wedge b\in G_p(Y,\hat{Y})\wedge b\in LCA(\hat{y},Y) \\
\quad \textit{(iii)}\ \nexists G'_t(Y,\hat{Y})\ \textit{satisfying (i) and (ii) and such that}: \\
\qquad\qquad |\{a: a\in G_t(Y,\hat{Y})\wedge a\in LCA_{all}(Y,\hat{Y})\}| > \\
\qquad\qquad |\{a': a'\in G'_t(Y,\hat{Y})\wedge a'\in LCA_{all}(\hat{Y},Y)\}| \\
\qquad \nexists G'_p(Y,\hat{Y})\ \textit{satisfying (i) and (ii) and such that}: \\
\qquad\qquad |\{b: b\in G_p(Y,\hat{Y})\wedge b\in LCA_{all}(Y,\hat{Y})\}| > \\
\qquad\qquad |\{b': b'\in G'_p(Y,\hat{Y})\wedge b'\in LCA_{all}(\hat{Y},Y)\}| \\
\quad \textit{(iv)}\ \forall y\in Y; \exists p_y\in G_t(Y,\hat{Y}): p_y\in paths_{min}(y,a)\wedge a\in LCA_{all}(Y,\hat{Y}) \\
\qquad \forall \hat{y}\in\hat{Y}; \exists p_{\hat{y}}\in G_p(Y,\hat{Y}): p_{\hat{y}}\in paths_{min}(\hat{y},b)\wedge b\in LCA_{all}(\hat{Y},Y) \\
\quad \textit{(v)}\ \forall a\in G_t(Y,\hat{Y})\cap LCA_{all}(\hat{Y},Y); \exists p_y\in G_t(Y,\hat{Y}): \\
\qquad\qquad p_y\in paths_{min}(y,a)\wedge y\in Y \\
\qquad \forall b\in G_p(Y,\hat{Y})\cap LCA_{all}(Y,\hat{Y}); \exists p_{\hat{y}}\in G_p(Y,\hat{Y}): \\
\qquad\qquad p_{\hat{y}}\in path_{min}(\hat{y},b)\wedge \hat{y}\in\hat{Y}
\end{cases}
$$

The maximization of $F_{LCA}(G_t(Y,\hat{Y}),G_p(Y,\hat{Y}))$ is subject to a set of constraints: Constraint (i) requires all class nodes of an initial set (Y or $\hat{Y}$) to be included in the final subgraphs. Constraint (ii) enforces the existence of at least one LCA for each node of $Y\cup\hat{Y}$, in the subgraphs. Constraint (iii) limits the total number of LCAs used to the minimum required in order to be able

to satisfy constraints (i) and (ii). Constraint (iv) implies the existence of at least one path connecting each class node of each subgraph to one of its LCAs, while constraint (v) implies the inverse, i.e. that each LCA of the subgraphs is connected with at least one class node of each subgraph.

One way to solve this maximization problem would be to create all possible $G_t(Y, \hat{Y})$ and $G_p(Y, \hat{Y})$ graphs, respecting the above constraints and then choose the ones leading to the highest F_{LCA}. This procedure is computationally expensive and can be approximated through the process presented in algorithm 1.

Algorithm 1 Approximation algorithm for computing $G_t(Y, \hat{Y})$ and $G_p(Y, \hat{Y})$, subgraphs of $G_t^{ex}(Y, \hat{Y})$ and $G_p^{ex}(Y, \hat{Y})$

procedure GetSubgraphs($G_t^{ex}(Y, \hat{Y})$, $G_p^{ex}(Y, \hat{Y})$)
  $LCA_{all} \leftarrow getLCAsFrom(G_t^{ex}(Y, \hat{Y}), G_p^{ex}(Y, \hat{Y}))$
  $bestLCAs \leftarrow$ GetBestLCAs($LCA_{all}, G_t^{ex}(Y, \hat{Y}), G_p^{ex}(Y, \hat{Y})$)
  $finalPaths \leftarrow$ GetBestPaths($bestLCAs, G_t^{ex}(Y, \hat{Y}), G_p^{ex}(Y, \hat{Y})$)
  $G_t(Y, \hat{Y}) \leftarrow$ all paths from $finalPaths$ containing a node $\in Y$
  $G_p(Y, \hat{Y}) \leftarrow$ all paths from $finalPaths$ containing a node $\in \hat{Y}$
end procedure
procedure GetBestLCAs($LCAs$, G_t, G_p)
  $sortedLCAs \leftarrow sort(LCAs)$ ▷ sort LCAs, by the number of Y and $\hat{Y}$ that they connect, in descending order
  $bestLCAs \leftarrow \{\}$, $i \leftarrow 1$
  repeat
    $bestLCAs \leftarrow bestLCAs \cup sortedLCAs_i$
    $i \leftarrow i + 1$
  until Satisfied($bestLCAs$, $G_t(Y, \hat{Y})$, $G_p(Y, \hat{Y})$) ▷ procedure SATISFIED checks whether constraints (i), (ii) and (iii) of the optimization problem are satisfied
  for $i \leftarrow 1$ to $sizeof(bestLCAs)$ **do** ▷ top-down redundancy removal
    if Satisfied($bestLCAs \setminus bestLCAs_i$, $G_t(Y, \hat{Y})$, $G_p(Y, \hat{Y})$) **then**
      $bestLCAs \leftarrow bestLCAs \setminus bestLCAs_i$
    end if
  end for
  for $i \leftarrow sizeof(bestLCAs)$ to 1 **do** ▷ bottom-up redundancy removal
    if Satisfied($bestLCAs \setminus bestLCAs_i$, $G_t(Y, \hat{Y})$, $G_p(Y, \hat{Y})$) **then**
      $bestLCAs \leftarrow bestLCAs \setminus bestLCAs_i$
    end if
  end for
  return $bestLCAs$
end procedure
procedure GetBestPaths($bestLCAs$, $G_t(Y, \hat{Y})$, $G_p(Y, \hat{Y})$)
  $bestY \leftarrow \{\}$, $best\hat{Y} \leftarrow \{\}$
  for $i \leftarrow 1$ to $sizeof(bestLCAs)$ **do**
    for $j \leftarrow 1, sizeof(Y)$ **do** ▷ find nodes Y_j, for which $bestLCA_i$ is an LCA
      if $bestLCAs_i \in LCAs(Y_j, \hat{Y})$ **then**
        $bestY \leftarrow bestY \cap$ BestPath($bestLCAs_i, Y_j$) ▷ BestPath selects the path of $path_{min}(bestLCA_i, Y_j)$ that shares most common nodes with all other selected paths
      end if
    end for
    for $j \leftarrow 1$ to $sizeof(\hat{Y})$ **do**
      if $bestLCAs_i \in LCAs(\hat{Y}_j, Y)$ **then**
        $best\hat{Y} \leftarrow best\hat{Y} \cap$ BestPath($bestLCAs_i, \hat{Y}_j$)
      end if
    end for
  end for
  return $bestY \cup best\hat{Y}$
end procedure

Procedure GetBestLCAs returns an approximation of the minimum amount of $LCAs$ needed in order to satisfy constraints (i), (ii) and (iii) of the maximization problem. This is achieved by initially sorting, in descending order, all $LCAs$ by the number of Y and $\hat{Y}$ that they connect. On this list, we perform two passes, top-down and bottom-up, removing all redundant $LCAs$, i.e $LCAs$ of nodes for which other $LCAs$ are already included in the list. In the final step of the algorithm, GetBestPaths selects the minimum paths that satisfy constraints (iv) and (v). In case two or more paths exist that connect the same node with an LCA, we choose the one which leads to the smallest possible subgraphs.

An interesting issue arises when a class and one of its ancestors co-exist in the predicted or the true class sets. Assume for example that a system A predicts that an instance belongs to node X, while another system B assigns it also to one of the ancestors of X. Each extra ancestor of X would lead to higher F_1 score since it would increase the size of the $Y_{aug} \cap \hat{Y}_{aug}$ set. This happens because all the ancestors of an $LCA(n_1, x_2)$ are also ancestors of nodes n_1 and n_2. We address this issue by removing from set Y any node y for which $\exists y' \in Y : y'$ is a descendant of y. We then do the same for set $\hat{Y}$ by removing each node $\hat{y}$ for which $\exists \hat{y'} \in \hat{Y} : \hat{y'}$ is a descendant of $\hat{y}$.

LCA precision, recall and F_1 are not purely set-based measures, but are actually a bridge between pair-based and set-based measures. They are based on augmented sets of predicted and true nodes and calulated scores, based on the relation of those two sets. However the use of the LCA(n, S), is actually a pairing between n and its nearest node of set S. So these measures could also be characterized as hybrid, combining the advantages of both types of measures.

2.5 Summary of Approaches

Reference	D1	D2	D3
[Dekel et al., 2004]	SL	T	non-MLNP
[Holden and Freitas, 2006]	SL	T	non-MLNP
[Sun and Lim, 2001]	ML	T	non-MLNP
GIE	ML	DAG	non-MLNP
MGIA	ML	DAG	non-MLNP
[Kiritchenko et al., 2005]	ML	DAG	non-MLNP
[Struyf et al., 2005]	ML	DAG	non-MLNP
[Cai and Hofmann, 2007]	ML	DAG	non-MLNP
[Cesa-Bianchi et al., 2006]	ML	DAG	non-MLNP
$P_{LCA}, R_{LCA}, F_{LCA}$	ML	DAG	non-MLNP

Table 2 Summary of the different evaluation measures according to the three dimensions: a) single or multi-label problem, b) structure of the hierarchy and c) mandatory leaf node prediction problem or not.

Table 2 summarizes the approaches described in the previous section along the three dimensions:

- Dimension 1: the problem can be either a single-labelling one (SL), where each instance is assigned to only one class label, or multi-labelling (ML) where each instance can be assigned multiple class labels.
- Dimension 2: the graph structure can be either a tree (T) or a directed acyclic graph (DAG).
- Dimension 3: each instance is classified only to leaf class labels of the hierarchy (mandatory leaf node prediction (MLNP)) or to any class in the hierarchy (non-MNLP).

3 Case Studies

In this section we apply various measures to selected cases in order to demonstrate their pros and cons. As a representative of the previously proposed pair-based measures we chose the Graph Induced Error (GIE), while for set-based measures we selected the hierarchical versions of precision (P_H), recall (R_H), F_1-measure ($F_H = \frac{2 \cdot P_H \cdot R_H}{P_H + R_H}$) and Symmetric Difference Loss ($l_\Delta(Y_{aug}, \hat{Y}_{aug})$), using all the ancestors of the predicted ($\hat{Y}$) and true (Y) labels in order to augment the sets of classes. We also use our proposed pair-based measure MGIA and the set-based LCA versions of precision (P_{LCA}), recall (R_{LCA}), F_1-measure (F_{LCA}) in order to illustrate their advantages and limitations, as well as the differences between the two types of measure. The LCAs used in this section are minimum LCAs. Regarding MGIA we also provide in brackets the $fnerror$, before the transformation that we proposed in Section 2.3.3, in order to have an easier comparison with GIE. All the above measures are implemented in a fast and easy to use tool written in C++ that is open source and available for download.[3]

Like all pair-based methods, GIE and MGIA require a maximum distance threshold, above which nodes are paired with a default one. In the cases studied here, this threshold is set to 5.

Based on the three dimensions of the hierarchical classification problem (Sections 2.1 and 2.5) and the subproblems presented in Section 2.2, which highlight important challenges in hierarchical evaluation, we present cases of specific problem settings. The list of cases here is not exhaustive, but it is sufficient to motivate the use of the proposed measures. Additionally, in Section 4, we present results on real datasets with real classification systems.

3.1 Simplest problem setting: single-label classification in a tree hierarchy

The simplest problem setting in hierarchical classification is the one where each instance belongs in only one category (single-label classification), the hierarchy

[3] The tool is available from http://nlp.cs.aueb.gr/software_and_datasets/HEMKit.zip

is a tree and classification is allowed only to leaves. Figure 11(a) presents such a case, where the true category of an instance is *Pop*, while the predicted one is *Rock*. Figure 11(b) presents a similar case, where the predicted class is *Theater*, which is a more serious mistake. Table 3 presents the results of each measure for each of these two cases.

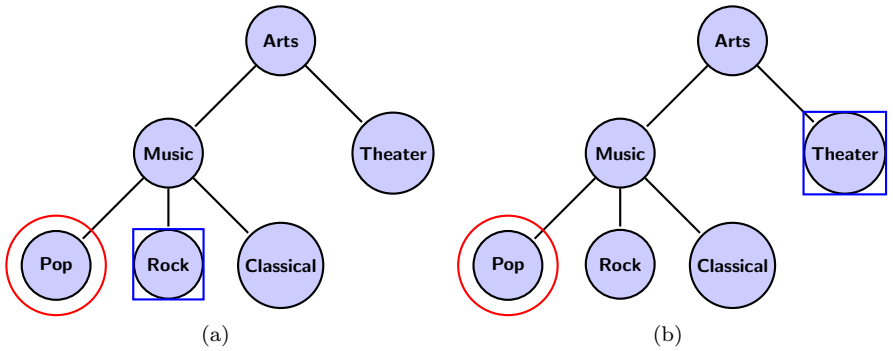


Fig. 11 Simplest case of misclassification. Nodes surrounded by circles are the true classes while the nodes surrounded by rectangles are the predicted classes.

	pair-based		set-based measures						
	GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
a	2	0.8(2)	0.66	0.66	0.66	2	0.5	0.5	0.5
b	3	0.7(3)	0.5	0.33	0.4	3	0.5	0.33	0.4

Table 3 Results per measure for Figure 11. The MGIA scores before the transformation of Section 2.3.3 are shown in brackets.

The first observation is that GIE behaves in the same desirable way as MGIA in such simple settings, since they both penalize more the second case (error increases and accuracy decreases). This is also true for all the set-based measures (F_H, $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ and F_{LCA}), since they all penalize the second case more than the first. The only interesting observation here is between the existing hierarchical measures of precision, recall, F_1 and the proposed LCA versions of them. In the first case, the LCA versions use in their calculations only the node *Music*, which is the LCA of *Pop* and *Rock*, while the existing hierarchical versions also take into account node *Arts*. In that way the LCA versions are stricter in the cases (a) than the existing set-based measures. We believe that such a behavior is desirable, because for every extra node above *Arts* the LCA versions would provide the same results, while the existing set-based measures would underestimate the error, giving higher results.

3.2 Switching the **D1** dimension and Handling the Pairing Problem

The cases presented here alter the **D1** dimension of the hierarchical classification problem, by allowing an instance to belong in more than one category. The problems in these settings arise when the number of true and predicted labels (classes) differ, which was defined in Section 2.2 as the Pairing Problem. In this elementary case, where the true and predicted classes are at the same level of the hierarchy, Figures 12(a) and 12(b) are two symmetric variants leading to different, but symmetric hierarchical precision (P_H) and recall (R_H) scores, as shown in Table 4. The same is true for our proposed LCA versions of precision and recall (P_{LCA} and R_{LCA}). The results between the hierarchical versions and the LCA versions differ, because the LCA versions ignore the graph above the node *Music*, which is the lowest common ancestor of nodes *Pop*, *Rock* and *Classical*. The hierarchical versions on the other hand also take into account node *Arts* and in that way they give higher results. This behavior is undesirable, since each node above *Music* would increase the results of the hierarchical measures, but would not affect the LCA versions.

All the other measures give the same result in both cases. $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ always computes the symmetric difference between the two augmented sets. The symmetric difference takes into account only the sum of false positives and false negatives, which are the same in both cases. Among the pair-based measures, GIE seems inappropriate for this problem. It matches the true node (*Pop*) with one of the two predicted nodes (*Rock* and *Classical*) in Fig 12(a) and penalizes the unmatched predicted class with the maximum cost, ignoring the fact that the misclassification is in the proximity of the correct category. Similarly, in Figure 12(b) MGIA provides a more suitable evaluation, as it allows multiple categories to match to the nearest one.

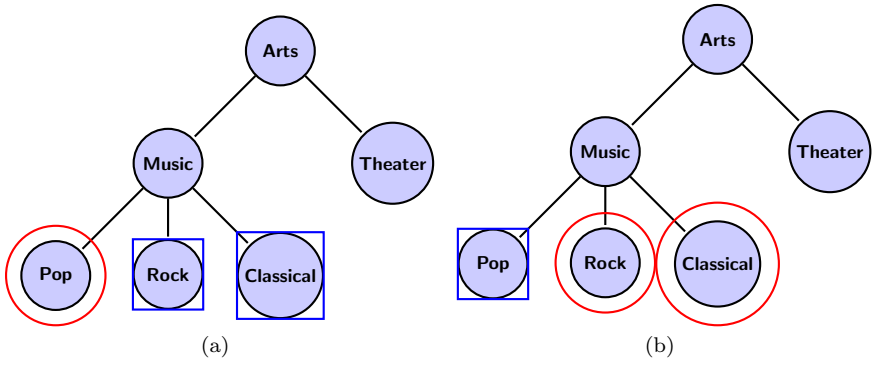


Fig. 12 Different numbers of true and predicted labels. Nodes surrounded by circles are the true classes while the nodes surrounded by rectangles are the predicted classes.

	pair-based		set-based measures						
	GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
a	7	0.73(4)	0.5	0.66	0.57	3	0.33	0.5	0.4
b	7	0.73(4)	0.66	0.5	0.57	3	0.5	0.33	0.4

Table 4 Results per measure for Figure 12.

The more complex case of Figure 13 is an example showing that taking into account all the ancestors is undesirable compared to our proposed LCA approach for set based measures. The hierarchy is still a tree and the classification is multi-label. *Europop* is predicted correctly, while an extra false category is predicted. Although the mistake in Figure 13(b) is worse than that of Figure 13(a), since it is further from the true class *Europop*, all set-based measures except our proposed LCA measures continue to give the same results. This is because $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ and the hierarchical versions of precision, recall and F_1 take into account all the ancestors of the predicted and true labels, while the LCA versions use the augmented graphs $G_t(Y, \hat{Y})$ and $G_p(Y, \hat{Y})$, which were created using only the least common ancestors (LCAs). LCA measures pair each node with the closest node of the other set and in that way take into account the distance between predicted and true nodes, which the other set-based measures ignore.

	pair-based		set-based measures						
	GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
a	2	0.8(2)	0.8	1	0.89	1	0.66	1	0.8
b	3	0.7(3)	0.8	1	0.89	1	0.66	0.66	0.66

Table 5 Results per measure for Figure 13.

Thus in Figure 13(a) the augmented sets of the LCA are {*Europop*, *Pop*} and {*Europop*, *Pop*, *Beat Music*}, while for all the other set-based measures they are {*Europop*, *Pop*, *Music*, *Arts*} and {*Europop*, *Pop*, *Beat Music*, *Music*, *Arts*}. In Figure 13(b) the augmented sets of LCA become {*Europop*, *Pop*, *Music*} and {*Europop*, *Rock*, *Music*}, while for all the other set-based measures they remain the same. The differentiation between such cases is an advantage of the LCA versions of precision, recall and F_1 over existing set-based measures.

3.3 Single label classification in a DAG Hierarchy

In Figure 14(a) we assume a single-label classification task, where the hierarchy is a *DAG*. In this *DAG* there are two paths from the root (node *Arts*) to node *Opera*. It is worth noting that this is the simplest case, where both paths {*Arts*, *Music*, *Opera*} and {*Arts*, *Theater*, *Opera*} have the same length.

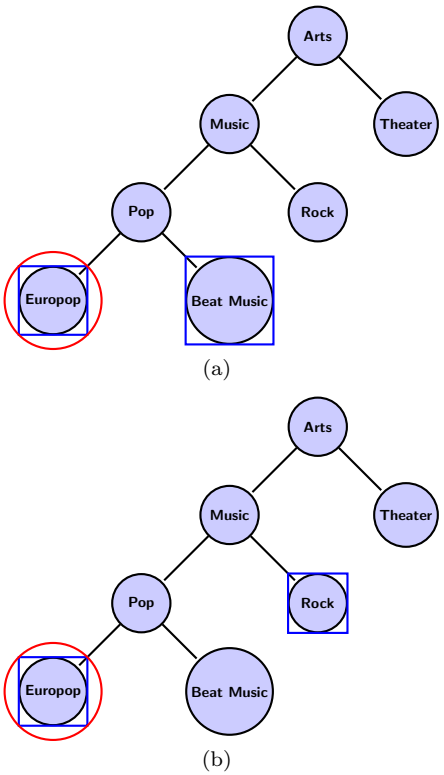


Fig. 13 Different numbers of true and predicted labels. The distance to the closest true class differs per predicted class. Nodes surrounded by circles are the true classes, while the nodes surrounded by rectangles are the predicted classes.

In this case, pair-based measures (Table 6) remain unaffected compared to Figure 14(b) which is a tree, a desirable behavior that is due to the use of the shortest path from *Pop* to *Opera*, while existing set-based measures are affected. In particular, they calculate a misclassification error that takes into account both of the two alternative paths from *Pop* to *Opera*. On the other hand LCA measures behave like the pair-based measures, due to the use of the lowest common ancestor, having again an advantage over existing set-based measures.

	pair-based		set-based measures						
	GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
a	2	0.8(2)	0.5	0.66	0.57	3	0.5	0.5	0.5
b	2	0.8(2)	0.66	0.66	0.66	2	0.5	0.5	0.5

Table 6 Results per measure for Figure 14.

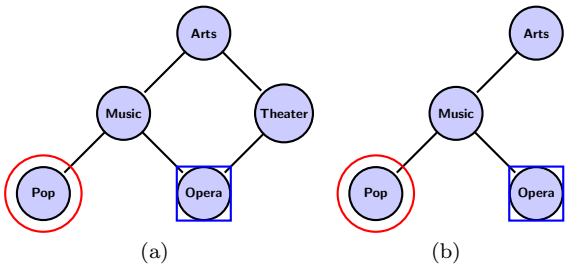


Fig. 14 Many paths of the same length leading from *Arts* to *Opera*. Nodes surrounded by circles are the true classes, while the nodes surrounded by rectangles are the predicted classes.

3.4 Multi-label Classification in a DAG

In this subsection we study cases where the multi-label classification combined with the DAG hierarchy (dimensions **D1** and **D2**) lead to different behavior of the set-based evaluation measures. Figure 15 presents such a case, where although GIE is affected by the matching problem discussed in Section 3.2, the set-based methods give similar results to each other according to Table 7. The multiple paths phenomenon does not affect the hierarchical versions of precision, recall and F_1, because all nodes above the lowest common ancestors *Theater* and *Music* of *Drama*, *Opera* and *Rock* are also shared by all classes (true and predicted). With this example we wish to show that there are certain cases in which existing set-based methods give the same results as the LCA ones, but we have not identified cases in which the behavior of an existing set-based measure would be more desirable than that of the LCA versions.

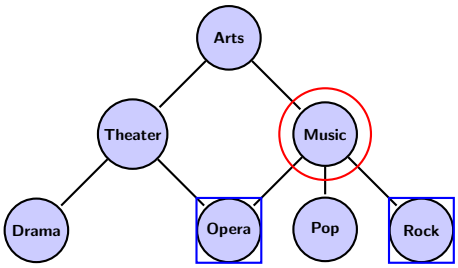


Fig. 15 Combining the pairing problem with alternative paths in a single example, where none of the set-based methods is affected. Nodes surrounded by circles are the true classes while the nodes surrounded by rectangles are the predicted classes.

The case shown in Figure 16 differs from the previous case, since the right sub-graph now connects to the left one only through node *Arts* and also *Beat Music* connects to *Music* through two different nodes *Electro* and *Pop*. This

pair-based		set-based measures						
GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
7	0.6(6)	0.4	0.66	0.5	4	0.4	0.66	0.5

Table 7 Results per measure for Figure 15.

is the reason why now hierarchical versions of precision, recall and F_1 differ from the LCA versions, as shown in Table 8. The LCA versions, due to their nearest common ancestor approach, ignore *Electro*, while all other set-based measures count both *Electro* and *Pop* and thus over-penalize the error of *Beat Music*.

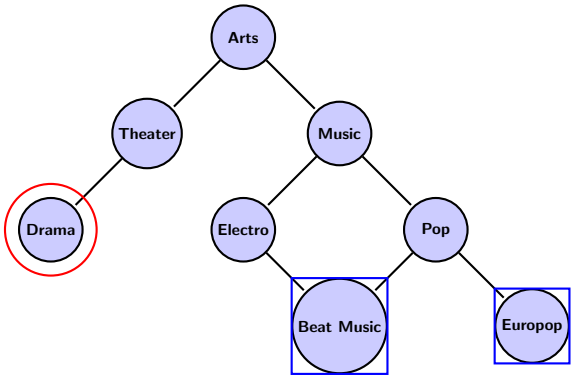


Fig. 16 Combining the pairing problem with alternative paths in a single example where all previous set-based measures behave undesirably, while our proposed LCA measures do not. Nodes surrounded by circles are the true classes while the nodes surrounded by rectangles are the predicted classes.

pair-based		set-based measures						
GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
10	0(10)	0.166	0.33	0.22	7	0.2	0.33	0.25

Table 8 Results per measure for Figure 16.

3.5 Over and under-specialization

In all the previous cases, the predicted and the true categories were leaves of the hierarchies, but as we discussed in Section 2.1 (dimension **D3**) this is not always the case. Figure 17 presents simple examples of an inner node being

either a true (Figure 17(a)) or a predicted (Figure 17(b)) category. As shown in Table 9 the two cases receive symmetrical scores for precision and recall.

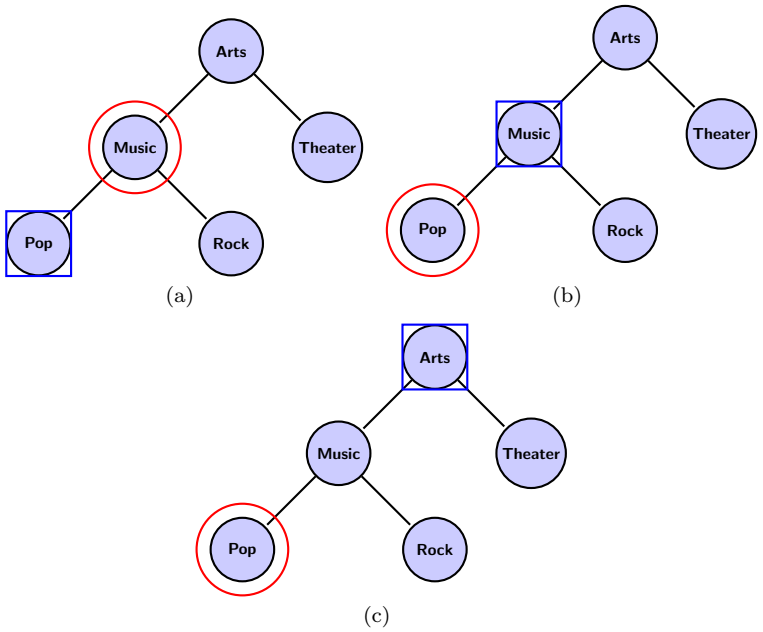


Fig. 17 Simplest over and under-specialization cases. Nodes surrounded by circles are the true classes while the nodes surrounded by rectangles are the predicted classes.

	pair-based		set-based measures						
	GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
a	1	0.9(1)	0.66	1	0.8	1	0.5	1	0.66
b	1	0.9(1)	1	0.66	0.8	1	1	0.5	0.66
c	2	0.8(2)	1	0.33	0.49	2	1	0.33	0.49

Table 9 Results per measure for Figure 17.

Figure 17(a) shows a case of over-specialization. As described in Section 2.2, different evaluation measures treat this type of error differently. One could even argue that since *Pop* is predicted, *Music* is also predicted as a direct ancestor of it. But the true category is *Music* not *Pop* and as shown in Table 9 all measures treat *Music* as a misclassification error.

Regarding under-specialization, the simplest example is shown in Figure 17(b). It is also considered an error that is more severe the further the true category is from the predicted one. For example in Figure 17(c) the predicted

node is an ancestor of the predicted node of Figure 17(b). All measures lead to a higher error estimate in this case than in 17(b). A similar example for over-specialization would lead to the same observations.

Our proposed measures do not offer any advantage along this dimension of the hierarchical classification problem, compared to the existing ones. We have not observed any negative behaviors of the existing measures, that we would like to correct. We just wish show that our new measures do not have any peculiar behavior regarding this dimension of the problem.

3.6 Long distance predictions

The aim of this case (Figure 18) is to show how each of the two types of measure (pair-based and set-based) handle very large distances between predicted and true labels. As we discussed in Section 2.2, the long distance problem is not affected by the dimensions of the hierarchical classification problem. Pair-based measures compute the distance between each pair of predicted and true nodes and if this distance is above a certain threshold, a standard maximum distance is assigned. Set-based measures can use a threshold on the number of ancestors of the predicted and true nodes that will be used in the augmented sets. Using this threshold, we impose an artificial common ancestor to be used in order to connect at least one predicted with one true node, at a distance equal to the threshold.

For example, in the case shown in Figure 18(a), if a maximum distance of 4 is used for pair-based measures, then for set-based measures it would be reasonable to request a lowest common ancestor at distance 2. By adding the distance of the lowest common ancestor to both the true and the predicted labels, a distance of 4 between them is reached.

This operation on the hierarchy of Figure 18(a) leads to the hierarchy of Figure 18(b), where an artificial node 0 was used in order to directly connect *Theater* and *Pop*. If multiple distant predictions exist, we add only one artificial node 0, which is used in order to connect the true and predicted nodes of each of these distant predictions. The results of each measure are presented in Table 10. In this example we see that all the measures can be run with a maximum distance threshold that might be necessary for computational reasons or due to the long distance problem discussed in Section 2.2. Pair-based measures are affected more by the threshold than set-based ones, as shown in the example. $l_{\Delta}(Y_{aug}, \hat{Y}_{aug})$ decreased by 2 points, while MGIA decreased by 4. This is due to multiple counting of paths, which most of the times is undesirable as we will discuss in the next section.

3.7 Multiple path counting

Due to their use of pairs of true and predicted classes, pair-based methods will often count the same path more than once. This multiple counting increases

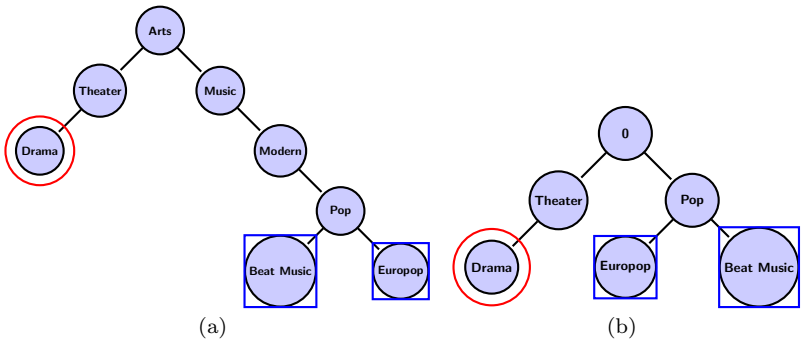


Fig. 18 Distant prediction problem. Nodes surrounded by circles are the true classes while the nodes surrounded by rectangles are the predicted classes.

	pair-based		set-based measures						
	GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
a	6 + Max	0.2(12)	0.16	0.33	0.22	7	0.16	0.33	0.22
b	4 + Max	0.466(8)	0.25	0.33	0.28	5	0.25	0.33	0.28

Table 10 Results per measure for Figure 18.

the error estimated by these methods. Figure 19 illustrates such a case. In Figure 19(a) the edge between *Drama* and *Theater* will be counted for the length of both {*Drama*, *Theater*, *Comedy*} and {*Drama*, *Theater*, *Arts*, *Music*, *Pop*, *Beat Music*} paths. As a result, pair-based measures in this case tend to overestimate the errors, in comparison to set-based ones. For each extra predicted node that we add as a descendant of *Theater*, pair-based measures increase the error by at least 2, while the size of Y_{aug} of set-based measures increases by 1, which seems more reasonable for such a change in the hierarchy.

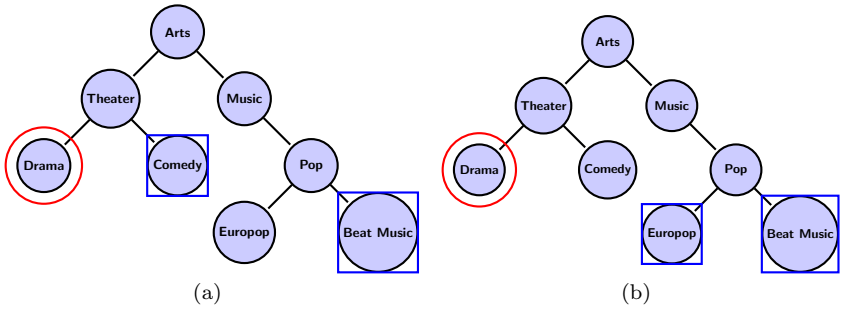


Fig. 19 Comparison between pair-based and set-based measures, counting paths more than once. Nodes surrounded by circles are the true classes while the nodes surrounded by rectangles are the predicted classes.

	pair-based		set-based measures						
	GIE	MGIA	P_H	R_H	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	P_{LCA}	R_{LCA}	F_{LCA}
a	7	0.533(7)	0.33	0.66	0.44	5	0.33	0.66	0.44
b	10	0.33(10)	0.2	0.33	0.25	6	0.2	0.33	0.25

Table 11 Results per measure for Figure 19.

Figure 19(b) presents a similar example. According to Table 11, the error of MGIA before the proposed transformation of Section 2.3.3 is increased from 7 to 10, while $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ increases from 5 to 6 and F_{LCA} decreases from 0.44 to 0.25. This is because the whole path form *Drama* to *Pop* is counted twice by the pair-based method. However double counting seems desirable in this case, as the errors in 19(b) are more severe than that in 19(a). While both measures penalize the errors in 19(b) more than in 19(a), the extra penalization of MGIA is roughly proportional to the distance between the true and the predicted category nodes, while that of the set based measures is not. All set based measures would give the same result even if *Europop* were a child of *Music*, which is a less severe error than when it is a child of *Pop*. Therefore, counting more than once the common paths, may be an advantage of the pair-based measures in some cases.

3.8 Summary

Table 12 presents the advantages and disadvantages of each measure from the scope of the cases presented in Section 2.2. The pair-based measures can handle alternative paths, while, from the set-based measures, only the proposed LCA measures are able to deal with them efficiently. All measures can handle over-specialization and under-specialization in some way. All set-based measures can deal with the pairing problem since they produce augmented sets, while GIE cannot handle it efficiently and this is why MGIA was proposed. Considering the *long distance problem*, only pair-based measures can handle it by definition, while set-based measures cannot handle it without using the threshold modification that we proposed in Section 3.6. Finally, multi-path counting is a special feature of the pair-based measures, which although usually undesirable, could make them behave better than the set-based measures in certain cases.

Table 13 presents the best measures that researchers could use in any given problem setting. The first dimension defines whether the problem is multi-label or single-label, while the second is about the type of the hierarchy. We have not added the third dimension here (classification to any node or only to leaves) since we have not found any advantages of the existing or proposed measures along that dimension.

As a general conclusion the proposed measures (MGIA and LCA-based) always behave better or at least as well as the existing measures of their type (pair-based and set-based). Therefore, if one wishes to use a pair-based or a set-

	pair-based		set-based measures		
	GIE	MGIA	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	F_{LCA}
Alternative Paths	+	+	-	-	+
Over-specialization	+	+	+	+	+
Under-specialization	+	+	+	+	+
Pairing Problem	-	+	+	+	+
Long Distance Problem	+	+	*	*	*
Multiple Path Count	-	-	+	+	+

Table 12 Summary table regarding evaluation measures over certain situations. * means that the measure cannot handle the situation, but can be modified to do so.

	Single-label	Multi-label
Tree hierarchy	Any pair-based or LCA	LCA or MGIA
DAG hierarchy	LCA or MGIA	LCA or MGIA

Table 13 Summary table regarding evaluation measures over certain problem settings.

based measure, we suggest using the ones proposed in this paper, instead of the existing ones. Furthermore, in most cases one should choose the LCA measures over MGIA, due to the multiple counting of paths discussed in Section 3.7. Multiple counting of paths is most of the times undesirable, since it leads to over-penalization. Additionally, these cases could also serve as benchmarks, in order to observe the behaviour of newly proposed hierarchical evaluation measures. In this way, we conclude the discussion regarding the behavior of the measures in benchmark cases. In the following section, we study them using real data and systems.

4 Empirical Study

In this section we apply various evaluation measures to the predictions of the systems that participated in the Large Scale Hierarchical Text Classification Pascal Challenges of 2011 (LSHTC2) and 2012 (LSHTC3). We also present ranking correlations between measures from the first task of the 2013 BioASQ challenge. The goal of this section, is to study using real data and systems, the extent to which the performance ranking of systems is affected by the choice between flat and hierarchical evaluation measures and also by the type of hierarchical measure used. In section 3 we demonstrated that, in certain cases, some measures behave more desirably than others. In this section we show that the differences among the methods also affect the rankings of real systems in practice. In the first subsection we present the datasets that we used, in the second subsection we discuss the evaluation measures included in the comparison and in the final subsection we discuss the results of the study.

4.1 Datasets

In LSHTC2 three different datasets were provided as three separate tasks. Each participant could participate in any or all of them with the same or with a different system. The first dataset (DMOZ) was based on pages crawled from the Open Directory Project (ODP), a human-edited hierarchical directory of the Web.[4] The hierarchy of this dataset was transformed into a tree and all instances deeper than level five of the hierarchy were transferred to the fifth level, thus leading to a hierarchy with a maximum depth of 5. This dataset was the smallest of the three, regarding the number of categories and instances.

The other two datasets of LSHTC2, also used in LSHTC3, are based on DBpedia.[5] They are called DBpedia Large and DBPedia Small, respectively. The largest of the two datasets, DBPedia Large, contains almost all abstracts of DBpedia, as instances to be used for training and classification, with the exception of some non-English abstracts. Therefore this dataset comprises many more categories than DMOZ and goes to a larger depth. DBpedia Small is a subset of DBpedia Large, selected in a way that led to a dataset of similar size than DMOZ, while maximizing the ratio of instances per node. This process has resulted in a much easier classification task. The hierarchy of the DBpedia Small dataset has been transformed into a DAG, by removing cycles, while cycles still appear in DBpedia Large.

All three datasets were pre-processed in the same way. All the words of the abstracts were stemmed and each stem was mapped to a feature id. The categories (classes) were also mapped to category ids. Each instance was represented in sparse vector format as a collection of category ids and a collection of feature ids accompanied by their frequencies in the instance. The mapping between ids, categories and stems was different for each dataset. Only leaves of each hierarchy were used as valid classification nodes for LSHTC2, while in LSHTC3 participants were also allowed to classify instances in inner nodes. For each inner node of the hierarchy that was assigned instances however, a dummy leaf was created for evaluation purposes as a direct child and all the instances were transferred to the child.

Table 14 presents basic statistics of the three datasets. The first two datasets are almost of the same size, but DBpedia Small is more multi-labeled and has a deeper, less ballanced hierarchy than DMOZ. However the ratio of training instances to categories is comparable in the two datasets (14.16 for DMOZ and 12.5 for DBpedia Small). DBpedia Large is very different in this respect, having a ratio of training instances to categories equal to 7.2. The ratio of multi-labeled training instances per category is almost the same across the two DBpedia datasets (23.27 for Small and 23.73 for Large) and much smaller for DMOZ (14.5). DBpedia Large is also much larger than the other two datasets in terms of training and test instances.

[4] http://www.dmoz.org/
[5] http://dbpedia.org/About

	DMOZ	DBpedia Small	DBpedia Large
categories	27,875	36,504	325,056
training instances	394,756	456,886	2,365,436
test instances	104,263	81,262	452,167
multi-label factor	1.0239	1.8596	3.2614
training inst. per cat.	14.16	12.5	7.2
multi-label train. inst. per cat.	14.5	23.27	23.73
max depth	5	10	14

Table 14 Basic statistics of datasets showing the number of categories, the number of training and test instances, the average number of true categories per instance (multi-label factor), the ratio of training instances to categories, the ratio of multi-labeled training instances to categories and the maximum depth of the hierarchy.

We also present system rankings from the results of the first task of the 2013 BioASQ Challenge.[6] The training data of this task consists of 800,000 *PubMed* biomedical journal abstracts belonging in 25,000 classes of the *MeSH* hierarchy.[7] The participants were asked to classify new *PubMed* documents, as they became available online, before they were manually annotated with *MeSH* headings by *PubMed* curators. Our proposed F_{LCA} was the hierarchical measure used in this task in order to decide the winning systems.

4.2 Evaluation Measures and Statistical tests

The evaluation measures that were used in this study were the ones presented in Section 3. Accuracy and GIE are reproduced here as reported during the challenges. Using these evaluation measures, different rankings of the participating systems are created. In order to measure the correlation between these rankings, we used Kendall's rank correlation [Kendall, 1938].

In the LSHTC2 challenge, statistical significance tests were only used for the flat evaluation measures. To the best of our knowledge, the literature does not provide special statistical significance tests for hierarchical measures. In this paper, as well as in LSHTC3, we performed a micro sign test (s-test) similar to that used by Yang and Liu [1999]. Each of the hierarchical measures provides a score for each instance and this score is always averaged over the number of instances. Assuming that:

- a_i is the performance of system a for instance i, according to an evaluation measure,
- b_i is the performance of system b for instance i, according to the same evaluation measure,
- n is the number of times that a_i and b_i differ over all i,
- k is the number of times that a_i performs better than b_i over all i,

[6] http://www.bioasq.org/

[7] http://www.ncbi.nlm.nih.gov/pubmed and http://www.ncbi.nlm.nih.gov/mesh

the null hypothesis (H_0) is that k has a bionomial distribution Bin(n, p), where $p = 0.5$. H_1 is that $p > 0.5$, meaning that system a is better than system b. According to Yang and Liu [1999], if n is greater than 12, which is always the case in these large scale problems, then the p-value can be approximately computed using the standard normal distribution for:

$$Z = \frac{k - 0.5n}{0.5\sqrt{n}} \tag{6}$$

It is worth stressing that the s-test only takes into account which system performs better at each instance, ignoring how much better it performs. Alternatively, the Wilcoxon signed-rank test [Wilcoxon, 1945] could take into account the difference in performance at each instance. For reasons of simplicity, however, in this paper we used the s-test.

4.3 Results

In this subsection we present the results for each dataset and discuss the behavior of each measure. Table 15 presents the results on the DMOZ dataset for all the systems that participated in the LSHTC2 challenge. Recall that DMOZ has a tree hierarchy and is the least multi-labeled dataset of the three. Systems are evaluated by each measure and are ranked in descending order. The number in brackets indicates the system's rank using the corresponding measure. If two systems have the same rank, it means that there is no statistically significant difference between their results according to our statistical significance test. Table 16 presents the Kendall rank correlation between each pair of rankings. The Kendall rank correlation ranges from -1 to 1.

The first observation is that the ranking of the flat accuracy is different from that of the other hierarchical measures. This shows that flat and hierarchical measures treat the problem differently. Another interesting observation is that the rankings also differ between hierarchical measures.

The handling of multiple labels per instance is an important aspect of the classification methods. Table 17 presents the average number of predictions per instance for each system. Since most instances of the dataset are single-labeled, most of the participants treated the task as a single-label one. As discussed in previous sections, the treatment of multi-labeling by different measures greatly affects their behavior, but since multi-labeling is rare in this dataset, this decision did not affect much the hierarchical measures. However, there are some examples of systems, such as M and J, which assign multiple labels and also perform better according to hierarchical measures than according to accuracy. D and E, on the other hand, perform worse using some hierarchical measures than with accuracy. The more multi-labeled the decisions, the greater the opportunity for a hierarchical measure to reward or penalize the systems for their decisions.

As discussed in previous sections, hierarchical measures vary in the way they handle multi-labeling and *DAG* hierarchies. Having a tree hierarchy

and almost being single-labeled, DMOZ does not reveal a lot of these differences. The tree hierarchy is the main reason why, according to Table 16, $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ and F_H are very highly correlated with F_{LCA}, since their main difference is in the way they treat multiple ancestors, something possible only in DAGs and not in trees. F_{LCA} and F_H are more correlated with each other than with $l_\Delta(Y_{aug}, \hat{Y}_{aug})$, as expected, since they differ in the calculations they perform on the augmented sets. We also observe a correlation between GIE and MGIA, although the main reasons here are not the hierarchy, but the limited multi-labeling that provides fewer opportunities to deal with the pairing problem and the proposed tranformation of the error that our MGIA performs (Section 2.3.3).

System	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
A	0.388 (1)	2.829 (2)	0.653 (3)	3.962 (2)	0.541 (2)	0.557 (2)
B	0.387 (2)	2.823 (1)	0.660 (1)	3.910 (1)	0.550 (1)	0.559 (1)
C	0.386 (3)	2.831 (3)	0.654 (2)	3.987 (3)	0.542 (1)	0.555 (3)
D	0.380 (4)	3.322 (6)	0.642 (5)	4.257 (4)	0.515 (4)	0.544 (5)
E	0.378 (5)	**3.832 (10)**	0.640 (6)	4.458 (7)	0.501 (5)	0.538 (6)
F	0.371 (6)	2.891 (4)	0.652 (4)	3.996 (4)	**0.538 (3)**	**0.547 (4)**
G	0.347 (7)	3.027 (5)	0.622 (7)	4.335 (6)	0.497 (6)	0.522 (7)
H	0.284 (8)	3.456 (7)	**0.497 (11)**	**5.878 (11)**	0.364 (8)	0.440 (9)
I	0.269 (9)	3.503 (9)	0.571 (8)	4.987 (9)	0.421 (7)	0.460 (8)
J	0.262 (10)	3.476 (8)	0.570 (8)	4.966 (8)	0.428 (7)	0.458 (8)
K	0.172 (11)	3.898 (11)	0.469 (12)	6.165 (12)	0.318 (10)	0.373 (11)
L	0.155 (12)	4.010 (12)	0.446 (13)	6.430 (13)	0.282 (11)	0.353 (12)
M	0.153 (13)	4.024 (13)	0.497 (10)	5.803 (10)	0.333 (9)	0.374 (10)
N	0.107 (14)	4.289 (14)	0.384 (14)	7.080 (14)	0.202 (12)	0.306 (13)
O	0.087 (15)	4.419 (15)	0.340 (15)	7.744 (15)	0.175 (13)	0.280 (14)

Table 15 DMOZ results in LSHTC2. Prominent rank changes are marked with bold.

	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
Acc	1					
GIE	0.829	1				
F_H	0.842	0.785	1			
$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	0.790	0.810	0.919	1		
MGIA	0.829	0.810	0.976	0.924	1	
F_{LCA}	0.867	0.810	0.976	0.924	0.962	1

Table 16 Kendall's rank correlation on the evaluation measure rankings of DMOZ in LSHTC2.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	1	1	1.11	1.22	1	1	1	1.02	1.01	1	1	1.02	1	1

Table 17 Average number of predictions per instance of DMOZ systems in LSHTC2.

Tables 18 and 19 present the LSHTC2 results on DBpedia Small, which is a more multi-labeled dataset with a DAG hierarchy. As expected, these characteristics greatly affect the behavior of the measures. The most important observation is that the two previously proposed hierarchical measures (GIE and $l_\Delta(Y_{aug}, \hat{Y}_{aug})$) that measure only the error, without applying a transformation to it, have very low correlation with Acc and the other hierarchical measures. Furthermore, taking into consideration the average predictions per instance of each system (Table 20), we observe a relation between the rankings of these two measures. By computing only the error these two measures take into account only false positives (FP) and false negatives (FN), without counting the true positives (TP). For this reason they tend to penalize systems with higher average predictions per instance, since they are more likely to make more mistakes. These measures also penalize less the systems that have a higher average number of predictions per instance, if they manage to have some extra TPs by doing so.

System	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
A2	0.374 (1)	**4.171 (5)**	0.647 (1)	**12.114 (6)**	0.356 (1)	0.481 (1)
B2	0.362 (2)	**4.364 (6)**	0.641 (3)	**12.000 (4)**	0.337 (2)	0.470 (2)
C2	0.354 (3)	4.076 (4)	0.646 (2)	11.651(3)	0.323 (4)	0.463 (3)
D2	0.351 (4)	3.858 (2)	0.629 (4)	11.332 (2)	0.329 (3)	0.462 (4)
E2	0.279 (5)	3.726 (1)	0.600 (5)	11.326 (1)	0.286 (5)	0.414 (5)
F2	0.252 (6)	3.859 (3)	0.579 (6)	11.996 (5)	0.280 (6)	0.399 (6)
G2	0.249 (7)	5.701 (7)	0.561 (7)	16.915 (7)	0.245 (7)	0.381 (7)

Table 18 DBpedia Small results in LSHTC2. Prominent differences of rankings are highlighted in bold.

	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
Acc	1					
GIE	-0.143	1				
F_H	0.905	-0.048	1			
$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	-0.143	0.810	-0.048	1		
MGIA	0.810	-0.143	0.714	-0.143	1	
F_{LCA}	0.905	-0.238	0.810	-0.238	0.905	1

Table 19 Kendall's rank correlation on the evaluation measure rankings of DBpedia Small in LSHTC2.

A2	B2	C2	D2	E2	F2	G2
2.04	1.94	1.82	1.51	1.11	1.14	2.84

Table 20 Average predictions per instance of DBpedia Small systems in LSHTC2.

Tables 21 and 22 present the results on the same dataset (DBpedia Small), but with the systems of LSHTC3. The number of systems participating in LSHTC3 is much larger than LSHTC2 (17 instead of 7). This is not only important for statistical reasons (more experiments lead to safer conclusions), but also because according to Table 23 we now have more systems with a higher average number of predictions per instance, something which affects the behavior of the measures. Another important difference is that in LSHTC3, systems were allowed to classify to inner nodes, even if these nodes did not have any training instance directly belonging to them. F_H and F_{LCA} are the hierarchical measures that are most correlated with flat accuracy, although the correlation is much lower in this case where we have many more systems and inner node classification is treated as a mistake by accuracy. The correlation between GIE and MGIA is much higher than that of LSHTC2, but they are not fully correlated. A high correlation also continues to be observed between GIE and $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ for the reason explained previously in LSHTC2.

A very interesting case is that of system $X2$, which predicts many categories (labels) per instance, 10.649 labels per instance, when the average true labels per instance is 1.8550. This means that a large number of predicted labels is wrong, while the predicted labels could still be in the vicinity of the correct ones. As expected, flat accuracy penalizes this behavior giving the lowest rank to this system. GIE and MGIA also penalize this system, although MGIA less severely. On the other hand, the set-based measures do not punish X2 that much (rank 6 for F_H, 4 for $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ and 9 for F_{LCA}). A closer look at the system shows that it is in fact not that bad. However, it returns all the nodes of a path from the root to leaf as predicted labels, instead of just the leaf. Set-based measures still penalize it when the leaf and its ancestors are wrong predictions, but they do not over-penalize it, unlike pair-based measures. This is a nice example of the difference between set-based and pair-based measures. We can also observe that MGIA and F_{LCA} are the less extreme measures of their types to a point where their ranks are very close, 10 and 9 respectively. This is because these measures can be seen as hybrid measures, since the first also conducts a set operation in order tranform the error into accuracy (although it is a pair-based measure) and the second does some kind of matching between true and predicted nodes in order to create the augmented sets (although it is a set-based measure). These characteristics help them overcome the weaknesses of the measures of their respective types and in that way their behavior is more desirable.

On the third dataset (DBpedia Large) we faced some computational issues with the hierarchical measures. The problem originated from the very large scale of the dataset's hierarchy, which is a DAG (in reality it contains circles, but we removed them). To avoid the computational problems, we run the evaluation measures with a maximum path threshold of 2 and 4. This means that all nodes are forced to have a lowest common ancestor at a depth of 1 and 2 respectively (if they do not have one we create a dummy one). Although this seems restrictive, it is very similar to the idea behind the Long Distance problem of Figure 2(e) discussed in Section 2. In the Long Distance problem

System	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
H2	0.438 (1)	3.060 (1)	0.709 (1)	9.096 (1)	0.421 (2)	0.543 (1)
I2	0.429 (2)	3.155 (2)	0.689 (3)	9.310 (2)	**0.398 (4)**	**0.525 (4)**
J2	0.42 (3)	3.530 (5)	0.692 (2)	10.143 (6)	0.403 (3)	0.529 (2)
K2	0.417 (4)	**4.428 (11)**	0.677 (5)	11.385 (11)	0.378 (6)	0.509 (5)
L2	0.408 (5)	3.187 (3)	0.680 (4)	9.561 (3)	**0.443 (1)**	**0.527 (3)**
M2	0.385 (6)	3.319 (4)	0.666 (7)	10.122 (5)	0.390 (4)	0.500 (6)
N2	0.371 (7)	4.991 (13)	0.645 (8)	13.117 (13)	0.342 (8)	0.476 (8)
O2	0.357 (8)	4.302 (10)	0.643 (8)	12.185 (12)	0.323 (9)	0.462 (10)
P2	0.354 (9)	3.550 (6)	0.633 (9)	**11.146 (8)**	0.381 (5)	0.478 (7)
Q2	0.327 (10)	3.600 (8)	0.639 (8)	10.944 (7)	0.312 (11)	0.450 (12)
R2	0.32 (11)	3.552 (7)	0.603 (10)	11.365 (10)	**0.361 (7)**	0.453 (11)
S2	0.298 (12)	5.693 (14)	0.549 (13)	16.873 (15)	0.243 (13)	0.407 (13)
T2	0.25 (13)	3.741 (9)	0.592 (11)	11.304 (9)	**0.089 (14)**	0.397 (14)
U2	0.249 (14)	5.701 (15)	0.561 (12)	16.915 (16)	0.245 (12)	0.381 (15)
V2	0.245 (15)	4.780 (12)	0.537 (14)	14.351 (14)	0.234 (13)	0.374 (16)
W2	0.063 (16)	9.139 (16)	0.345 (15)	24.009 (17)	0.045 (15)	0.208 (17)
X2	**0.047 (17)**	**25.775 (17)**	**0.668 (6)**	**9.607 (4)**	**0.321 (10)**	**0.471 (9)**

Table 21 DBpedia Small results in LSHTC3. With bold, prominent differences in rankings.

	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
Acc	1					
GIE	0.662	1				
F_H	0.765	0.485	1			
$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	0.485	0.735	0.662	1		
MGIA	0.691	0.618	0.721	0.588	1	
F_{LCA}	0.794	0.574	0.853	0.603	0.838	1

Table 22 Kendall's rank correlation on the evaluation measure rankings of DBpedia Small in LSHTC3.

H2	I2	J2	K2	L2	M2	N2	O2	P2
1.506	1.482	1.909	2.208	1.415	1.490	2.427	1.889	1.414
Q2	R2	S2	T2	U2	V2	W2	X2	
1.529	1.184	2.423	2.000	2.841	1.712	4.334	10.649	

Table 23 Average predictions per instance on DBpedia Small in LSHTC3.

we used dummy nodes in order to link nodes that were further than a threshold from each other, in order to avoid overpenalization. The same dummy nodes are used here for computational reasons.

Tables 24 and 25 present the results for a distance threshold of 4 for the systems of LSHTC2. Since this dataset has the most complex hierarchy and it is the most multi-labeled one, it should be treated very differently by each measure. Interestingly the rankings of F_H, MGIA and F_{LCA}, which are not based only on FP and FN, remain highly correlated with accuracy compared to the other measures (GIE and $l_\Delta(Y_{aug}, \hat{Y}_{aug})$), although the number of systems is not high enough (only 5 systems) in order to make safe conclusions.

Another interesting observation is the disagreement of GIE and MGIA about systems C3 and E3. As shown in Table 26, E3 predicts fewer categories per instance than C3. Since most of the times the predicted categories (labels)

are fewer than the true ones and GIE over-penalizes all the unmatched true categories, it is natural for GIE to penalize system C3 more than E3. This problem is fixed by MGIA, which allows multi-pairing and this is why it instead ranks C3 as a better system than E3.

We can also see that this difficult hierarchy affects the performance of $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ and its ranks become less correlated with F_{LCA}, compared to the previous datasets. A more interesting observation is that F_H, MGIA and F_{LCA} are completely correlated with each other (Table 25) and not correlated with the average number of predictions per instance (Table 26).

System	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
A3	0.347 (1)	4.647 (4)	0.538 (1)	42.470 (3)	0.319 (1)	0.44 (1)
B3	0.337 (2)	4.392 (2)	0.511 (2)	40.811 (1)	0.315 (2)	0.437 (2)
C3	0.283 (3)	**6.178 (5)**	0.440 (4)	50.709 (5)	**0.253 (4)**	0.39 (4)
D3	0.272 (4)	4.288 (1)	0.483 (3)	42.430 (2)	0.294 (3)	0.388 (3)
E3	0.177 (5)	**4.535 (3)**	0.314 (5)	47.957 (4)	**0.212 (5)**	0.331 (5)

Table 24 DBpedia Large results with a maximum path threshold of 4 in LSHTC2. With bold, prominent differences in rankings.

	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
Acc	1					
GIE	0	1				
F_H	0.8	0.2	1			
$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	0.2	0.8	0.4	1		
MGIA	0.8	0.2	1	0.4	1	
F_{LCA}	0.8	0.2	1	0.4	1	1

Table 25 Kendall's rank correlation on the evaluation measure rankings with a maximum path threshold of 4 on DBpedia Large in LSHTC2.

A3	B3	C3	D3	E3
3.15	2.69	3.62	2.81	1.27

Table 26 Average predictions per instance on DBpedia Large systems in LSHTC2.

Tables 27 and 28 present the results for a maximum path threshold of 2. The main purpose of this experiment is to show that the measures remain largely unaffected by this parameter. Indeed most rankings do not seem to be affected compared to Tables 24 and 25. Nevertheless, general advice is to keep the maximum paths parameter as large as possible, since it is always better to compute the exact value of a measure, than an estimation of it.

	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
A3	0.347 (1)	4.647 (4)	0.503 (1)	22.006 (3)	0.206 (2)	0.460 (2)
B3	0.337 (2)	4.392 (2)	0.475 (2)	21.028 (1)	0.207 (1)	0.461 (1)
C3	0.283 (3)	6.178 (5)	0.412 (4)	25.465 (5)	0.163 (3)	0.416 (3)
D3	0.272 (4)	4.288 (1)	0.439 (3)	21.702 (2)	0.155 (4)	0.405 (4)
E3	0.177 (5)	4.535 (3)	0.282 (5)	24.146 (4)	0.134 (5)	0.360 (5)

Table 27 DBpedia Large results with a maximum path threshold of 2 in LSHTC2.

	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
Acc	1					
GIE	0	1				
F_H	0.8	0.2	1			
$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	0.2	0.8	0.4	1		
MGIA	0.8	0.2	0.6	0.4	1	
F_{LCA}	0.8	0.2	0.6	0.4	1	1

Table 28 Kendall's rank correlation on the evaluation measure rankings with a maximum path threshold of 2 on DBpedia Large in LSHTC2.

Tables 29 and 30 present the results on the same dataset (DBpedia Large), but with the systems of LSHTC3. The main difference is that in LSHTC3, systems were allowed to classify to inner nodes. Table 31 shows that the average number of predictions per instance is similar to that of LSHTC2. The most interesting observation is that GIE and $l_\Delta(Y_{aug}, \hat{Y}_{aug})$ rank F3 as one of the worst systems, while all the other hierarchical measures and Accuracy ranked it once again high. It is even more interesting that, according to Table 31, system F3 provides the most labels per instance. The assignement of many labels is penalized heavily by measures based only on FP and FN (GIE and $l_\Delta(Y_{aug}, \hat{Y}_{aug})$), as we mentioned before.

Another interesting observation is that MGIA and F_{LCA} are not fully correlated anymore. In fact MGIA is more correlated with F_H than with F_{LCA}. As the hierarchy becomes more complicated and the results more multi-labeled our two proposed measures behave more differently. Finally system G3, which predicts the smallest number of instances per document, is one of the best systems according to all measures except MGIA which ranks as the second worst. This is because although G3 has the highest P_H and P_{LCA}, it also has a very low R_H and R_{LCA} compared to other systems. The computation of F_1 seems more sutable in this case compared to the transformation that we proposed for MGIA, one extra reason why we propose F_{LCA} over MGIA.

Similar experiments with a maximum path threshold of 2 were also conducted with systems of LHSTC3, but the results were similar to the ones of LSHTC2 and thus we omit them here, to save of space.

Tables 32, 33 and 34 present the Kendall's rank correlations of the three batches of the 2013 BioASQ challenge. These tables contain results for two flat and two hierarchical evaluation measures. The flat measures are accuracy and

System	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
F3	**0.381 (1)**	**6.104 (6)**	0.557 (1)	**42.790 (5)**	0.374 (2)	0.465 (1)
G3	0.346 (2)	3.756 (1)	0.513 (3)	33.630 (1)	**0.309 (5)**	0.456 (2)
H3	0.340 (3)	3.763 (2)	0.508 (4)	38.137 (2)	0.35 (3)	0.45 (3)
I3	0.333 (4)	3.763 (3)	0.507 (5)	44.435 (6)	0.31 (4)	**0.43 (5)**
J3	0.332 (5)	4.216 (4)	0.517 (2)	40.560 (3)	0.381 (1)	0.449 (4)
K3	0.272 (6)	4.288 (5)	0.483 (6)	42.430 (4)	0.294 (6)	0.388 (6)

Table 29 DBpedia Large results with a maximum path threshold of 4 in LSHTC3. With bold, prominent differences in rankings.

	Acc	GIE	F_H	$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	MGIA	F_{LCA}
Acc	1					
GIE	0.276	1				
F_H	0.6	0.138	1			
$l_\Delta(Y_{aug}, \hat{Y}_{aug})$	0.2	0.552	0.067	1		
MGIA	0.2	-0.276	0.6	-0.067	1	
F_{LCA}	0.828	0.071	0.828	0.276	0.414	1

Table 30 Kendall's rank correlation on the evaluation measure rankings with a maximum path threshold of 4 on DBpedia Large in LSHTC3.

F3	G3	H3	I3	J3	K3
3.949	1.482	2.315	2.903	2.902	2.810

Table 31 Average predictions per instance of DBpedia Large systems in LSHTC3.

Micro-F_1. The hierarchical measures are our proposed F_{LCA} and F_H. F_{LCA} and Micro-F_1 were used in the challenge in order to select the winning systems.

The first observation is that according to all three tables, the two flat measures are more correlated with each other, than with the rest of the measures. This is an indication that the flat and the hierarchical measures lead to different rankings, since they treat the problem differently. It is interesting that the same is not always true for the two hierarchical measures, since in the second batch F_H is more correlated with the two flat measures than with F_{LCA}. This observation shows that F_{LCA} has sufficiently different behavior from F_H. Finally, F_{LCA} is the least correlated with all the other measures.

	Acc	F_{LCA}	Micro-F_1	F_H
Acc	1			
F_{LCA}	0.91	1		
Micro-F_1	0.95	0.88	1	
F_H	0.89	0.90	0.87	1

Table 32 Kendall's rank correlation on the evaluation measure rankings with a maximum path threshold of 8 on batch 1 of the BioASQ challenge (Task 1).

	Acc	F_{LCA}	Micro-F_1	F_H
Acc	1			
F_{LCA}	0.89	1		
Micro-F_1	0.98	0.91	1	
F_H	0.94	0.92	0.94	1

Table 33 Kendall's rank correlation on the evaluation measure rankings with a maximum path threshold of 8 on batch 2 of the BioASQ challenge (Task 1).

	Acc	F_{LCA}	Micro-F_1	F_H
Acc	1			
F_{LCA}	0.90	1		
Micro-F_1	0.99	0.90	1	
F_H	0.92	0.92	0.90	1

Table 34 Kendall's rank correlation on the evaluation measure rankings with a maximum path threshold of 8 on batch 3 of the BioASQ challenge (Task 1).

The experiments presented in this section illustrated, with the use of real systems and datasets, that hierarchical measures treat the competing systems differently than flat measures. This was shown by presenting the differences in the rankings of the systems across the four datasets. Flat evaluation measures, which are commonly used, often provide a false indication of which system performs better by ignoring hierarchical dependencies of classes and treating all errors equally. As a result, their use guides research away from the methods that incorporate the hierarchy in the classification process. We also showed that different variants of hierarchical measures give different rankings under different conditions. The goal was not to choose the best measure, but to show that different hierarchical evaluation measures give different results, not only in absolute values but also in the ranking of the systems. Finally we showed that the scale of the task is also an issue which requires attention.

5 Conclusions

In this work we studied the problem of evaluating the performance of hierarchical classification methods. Specifically, this work abstracted and presented the key points of existing performance measures. We proposed a grouping of the methods into a) *pair-based* and b) *set-based*. Measures in the former group attempt to match each prediction to a true class and measure their distance. By contrast, set-based measures use the hierarchical relations in order to augment the sets of predicted and true labels, and then use set operations, like symmetric difference and intersection, on the augmented label sets.

In order to model pair-based measures, we introduced a novel generic framework based on flow networks, while for set-based measures we provided a framework based on set operations. Thus, salient features of these measures were stressed and presented under a common formalism.

Another contribution of this paper was the proposal of two new measures (one for each group) that address several deficiencies of existing measures. The proposed measures, along with existing ones, were assessed in two ways. First, we applied them to selected cases, in order to demonstrate their pros and cons. Second, we studied them empirically on four large datasets based on DMOZ, DBpedia and biomedical data (BioASQ) with different characteristics (single-label, multi-label,tree and DAG hierarchies). The analysis of the results showed that the hierarchical measures behave differently, especially in cases of multi-label data and DAG hierarchies. Also, the two proposed measures have shown a more robust behavior compared to their counterparts. Finally, the results supported our initial premise that flat measures are not adequate for evaluating hierarchical categorization systems.

Our analysis showed that although in certain rare cases pair-based measures may behave more desirably, in most cases the set-based method proposed in this paper (F_{LCA}) exhibits more desirable behavior than that of our proposed pair-based measure (MGIA), since it is actually a hybrid measure, because of the pairings it uses to select LCAs. This why we propose the use of F_{LCA} instead of all other hierarchical measures, although it is still an open-issue to devise a measure that combines all the pros of our proposed MGIA and F_{LCA} measures.

References

Alfred Aho, John Hopcroft, and Jeffrey Ullman. On finding lowest common ancestors in trees. In *Proc. 5th ACM Symp. Theory of Computing (STOC)*, pages 253–265, 1973.

Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

Hendrik Blockeel, Maurice Bruynooghe, Saso Dzeroski, Jan Ramon, and Jan Struyf. Hierarchical multi-classification. In *ACM SIGKDD 2002 Workshop on Multi-Relational Data Mining*, pages 21–35, 2002.

Florian Brucker, Fernando Benites, and Elena Sapozhnikova. An empirical comparison of flat and hierarchical performance measures for multi-label classification with hierarchy extraction. In *Proceedings of the 15th international conference on Knowledge-based and intelligent information and engineering systems - Volume Part I*, pages 579–589, 2011.

Lijuan Cai and Thomas Hofmann. Exploiting known taxonomies in learning overlapping concepts. In *International Joint Conferences on Artificial Intelligence*, pages 714–719, 2007.

Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Incremental algorithms for hierarchical classification. *Journal of Machine Learning Research*, 7:31–54, 2006.

E. P. Costa, A. C. Lorena, Carvalho, and A. A. Freitas. A review of performance evaluation measures for hierarchical classifiers. In *2007 AAAI Workshop, Vancouver*, 2007.

Ofer Dekel, Joseph Keshet, and Yoram Singer. Large margin hierarchical classification. In *Proceedings of the twenty-first international conference on Machine learning*, pages 209–216, 2004.

N. Holden and A.A. Freitas. Hierarchical classification of g-protein-coupled receptors with a pso/aco algorithm. In *IEEE Swarm Intelligence Symposium (SIS-06)*, pages 77–84, 2006.

Panagiotis G. Ipeirotis, Luis Gravano, and Mehran Sahami. Probe, count, and classify: categorizing hidden web databases. In *ACM SIGMOD international conference on Management of data*, SIGMOD '01, pages 67–78, 2001.

M. G. Kendall. A new measure of rank correlation. In *Biometrica*, volume 30, pages 81–93, 1938.

Svetlana Kiritchenko, Stan Matwin, and A. Fazel Famili. Functional annotation of genes using hierarchical text categorization. In *ACL workshop on linking biological literature, ontologies and databases: mining biological semantics*, 2005.

Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. 1997.

Aris Kosmopoulos, Eric Gaussier, George Paliouras, and Sujeevan Aseervatham. The ECIR 2010 large scale hierarchical classification workshop. *SIGIR Forum*, 44:23–32, 2010.

Andrew McCallum, Ronald Rosenfeld, Tom M Mitchell, and Andrew Y Ng. Improving text classification by shrinkage in a hierarchy of classes. In *ICML*,

volume 98, pages 359–367, 1998.

Stefanie Nowak, Hanna Lukashevich, Peter Dunker, and Stefan Rüger. Performance measures for multilabel evaluation: a case study in the area of image classification. In *Proceedings of the international conference on Multimedia information retrieval*, pages 35–44, 2010.

Carlos N. Silla, Jr. and Alex A. Freitas. A survey of hierarchical classification across different application domains. *Data Mining Knowledge Discovery*, 22:31–72, 2011.

Marina Sokolova and Lapal Guy. A systematic analysis of performance measures for classification tasks. *Information Processing Management*, 45:427–437, 2009.

Jan Struyf, Saso Dzeroski, Hendrik Blockeel, and Amanda Clare. Hierarchical multi-classification with predictive clustering trees in functional genomics. In Carlos Bento, Amlcar Cardoso, and Gal Dias, editors, *Progress in Artificial Intelligence*, volume 3808 of *Lecture Notes in Computer Science*, pages 272–283. 2005.

Aixin Sun and Ee-Peng Lim. Hierarchical text classification and evaluation. In *IEEE International Conference on Data Mining*, pages 521–528, 2001.

Aixin Sun, Ee-Peng Lim, and Wee-Keong Ng. Performance measurement framework for hierarchical text classification. *Journal of the American Society for Information Science and Technology*, 54:1014–1028, 2003.

F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

Lin Xiao, Dengyong Zhou, and Mingrui Wu. Hierarchical classification via orthogonal transfer. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 801–808, 2011.

Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49. ACM, 1999.