Integrated Framework for Improving Large-scale Hierarchical Classification

***Abstract*—Hierarchical classification (HC) approaches leverage the hierarchical structure during training and testing of models which helps in the improvement of classification performance and prediction efficiency. However, their performance improvement in comparison to flat approaches may be poor if the hierarchy used for training contains: (i) Inconsistent hierarchical relationships such as parent-child or siblings and (ii) Less cohesive or overlapping classes. Moreover, dynamic changes in the data characteristics over time requires new classes (orphan nodes) to be identified to improve the generalization performance of learned models. In addition, we also need to deal with imbalance data in large-scale problem where large number of classes have a few examples for training, posing statistical challenges. In this paper, we propose an integrated framework to address the aforementioned issues for improving large-scale HC. Our experimental evaluations on various image and text datasets shows improved performance with our proposed framework. SOURCE CODE: http://Anonymous/IntegratedFramework**

***Keywords*-Hierarchical Classification, Inconsistency, Logistic Regression, Scalability, Orphan Node Prediction**

I. Introduction

Hierarchy (Taxonomy) provides an easy and convenient way to organize data where classes (categories) are systematically arranged from the most generic to most specific in a top-down order [1, 2]. Hierarchical Classification (HC) deals with the task of classifying unlabeled instances into the hierarchy of classes using structural relationships. Several methods that exploit the hierarchical structure for improving classification performance have been developed in the literature [3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. For example, state-of-the-art HC approaches embed the parent-child relationships from the hierarchy either, within the regularization term [13] or the loss term [14]. The intuition behind Hierarchy Regularized Logistic Regression (HR-LR) or Hierarchy Regularized Support Vector Machine (HR-SVM) approach is that data-sparse child nodes benefit during training from data-rich parent nodes and has shown to achieve the best performance on standard HC benchmarks (LSHTC datasets - DMOZ-2010 and DMOZ-2012). In case of HierCost, a cost-sensitive learning approach was adapted. This method intuitively captures the hierarchical information by treating misclassifications differently based on the commonalities (common ancestors) between the true and the predicted labels. Another approach involves using feature selection in conjunction with HC [15, 16]. It helps to improve the accuracy and efficiency of model training along with reduce memory requirement which is crucial for large-scale problem. Other developed approaches for solving HC can be found in [17, 18, 19].

The hierarchical structure has a significant impact on the classification performance of model being trained [18]. If the hierarchy used has too many inconsistent relationships such as parent-child or siblings, less cohesive or overlapping categories, than the performance can be poor. In fact for some datasets HC methods are outperformed by flat methods that ignore the hierarchy [7, 13]. Oftentimes, the hierarchy is manually designed by domain experts based on semantic relationships. This type of curated hierarchy is prone to inconsistencies and is not optimal for achieving good classification performance. Moreover, partial or incomplete domain knowledge may result in less cohesive categories that are not easily separable and vice-versa. Furthermore, constant change in data distribution over time requires new categories (orphan nodes) to be identified to improve the taxonomic representation and hence classification performance [20]. Other problem associated with large-scale classification is the highly skewed distribution between classes, where majority of the classes have few instances (examples) for training (rare categories problem). For example, more than 75% of the categories in the ODP[1] and Yahoo![2] directory have five or fewer positive instances [13, 19]. Learning a classification model for these classes suffer from statistical challenges where mis-predictions tend to favor the larger classes. As a result, overall performance on the rare categories classes are unsatisfactory.

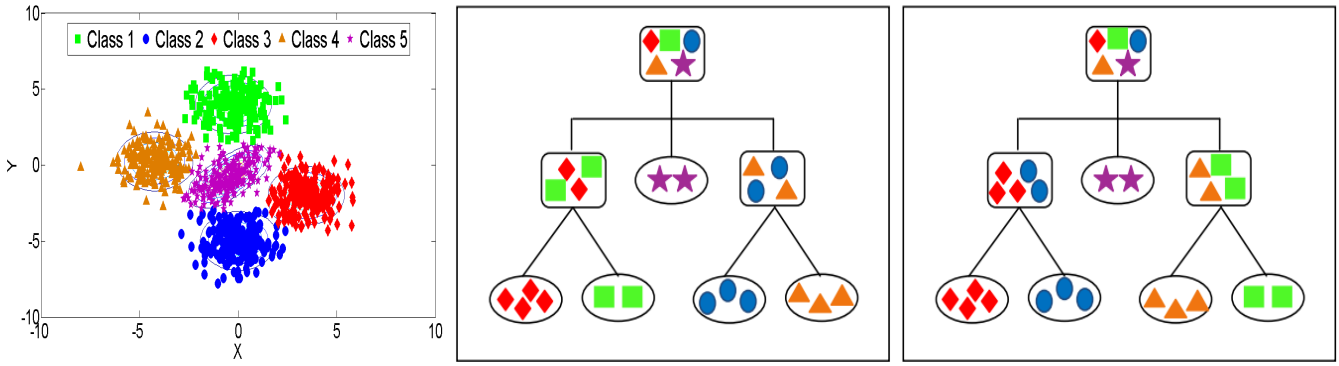
In this paper, we propose a solution to handle multiple issues that affect the HC performance. To summarize, our main contributions are as follows:

- We develop an *integrated framework* which consists of a multi-stage embarrassingly parallel pipeline to improve the HC performance.
- We propose the exploratory learning approach for orphan node identification that can be easily incorporated in our framework.
- An extensive case study was performed to analyze the strength and effectiveness of each step in the integrated framework.

II. Definitions and Notations

We use **bold** lower-case and upper-case letters to indicate vector and matrix variables, respectively. Symbol $\mathcal{N}$ denotes

[1]http://www.dmoz.org
[2]http://dir.yahoo.com



(a) Synthetic dataset with five class (b) Hierarchy 1 (c) Hierarchy 2

Figure 1. **(a) Synthetic dataset with five classes (marked with different symbol and color) and two different hierarchical structure shown in (b) and (c).**

the set of internal nodes in the hierarchy where for each node $n \in \mathcal{N}$ we learn the multi-class classifier denoted by $\mathbf{W}_n = \left[\mathbf{w}_n^c\right]_{c \in \mathcal{C}(n)}$ to discriminate between its children nodes $\left[\mathcal{C}_n^c\right]_{c \in \mathcal{C}(n)}$. $\mathcal{C}(n)$ denotes the set of children of node n and $\mathbf{w}_n^c$ represents the learned optimal model weight vectors for *c-th* child of node n. $\mathcal{L}$ denotes the set of leaf nodes (categories) to which instances are assigned. The total number of training and test instances are denoted by N_{Train} and N_{Test}. $T(n) \subseteq N_{Train}$ denotes the training instances considered at node n which correspond to all instances belonging to descendant categories at node n. $\mathcal{F}$ denotes the set of total features (dimensionality) for each instance where the *i-th* feature is denoted by f_i. $S_{\mathcal{F}} \subseteq \mathcal{F}$ denotes the subset of relevant features selected using feature selection algorithm. $\mathcal{D} = \left\{\left(\mathbf{x}(i),\, y(i)\right)\right\}_{i=1}^{N_{Train}}$ denotes the training dataset where $\mathbf{x}(i) \in \mathbb{R}^{\mathcal{F}}$ and $y(i) \in \mathcal{L}$. For training optimal model corresponding to *c-th* child at node n we use the binary label $y_n^c(i) \in \{\pm 1\}$ for *i-th* training instance where $y_n^c(i)$ = 1 iff $y(i) = c$ and $y_n^c(i)$ = -1, otherwise. Predicted label for *i-th* test instance $\widehat{\mathbf{x}}(i)$ is denoted by $\widehat{y}(i) \in \{\mathcal{L} \cup \mathcal{O}\}$ where $\mathcal{O}$ corresponds to the set of orphan nodes detected during testing phase.

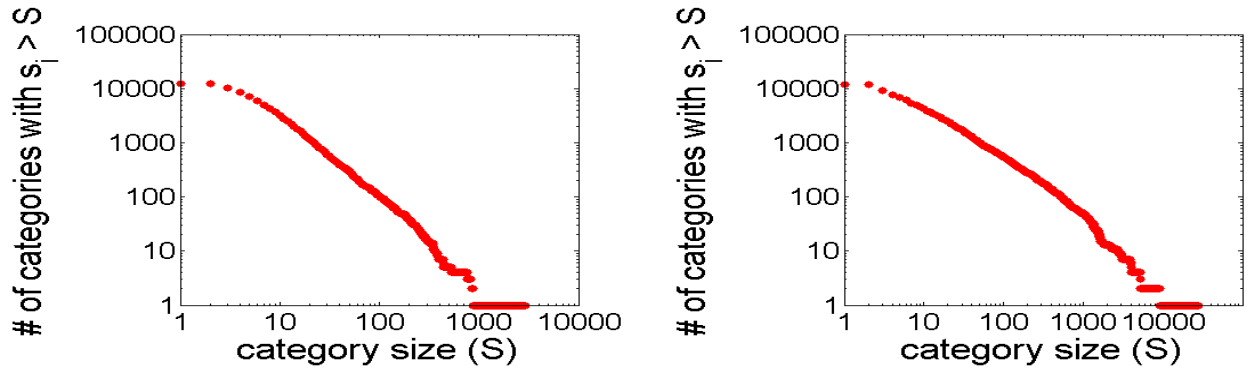
III. Motivation and Related Work

HC is a well-studied problem across different domains [17, 21]. Here we motivate the need for an integrated framework for classification by examining and highlighting the various critical issues that affect the HC performance in large-scale settings.

1) **Hierarchical structure inconsistency**: Designing a consistent hierarchy is challenging either due to insufficient domain knowledge or several confounding classes (such as *soc.religion.christian* and *talk.religion.misc* classes in Figure 4(a), both relate to religion). Moreover, hierarchy generation based on semantics is susceptible to inconsistencies [18, 22]. This problem is more common for large-scale datasets. To illustrate in detail, consider the example shown in Figure 1(a), it consist of 1000 points divided into five classes that are generated using gaussian distribution with different mean and variance. Figure 1(b) and Figure 1(c) shows two different possible hierarchical structures with these classes. *Hierarchy 1* separates examples into three categories at level 1, namely $\{$(♦, ■),(★),(▲, ●)$\}$ which are not consistent for classification since categories (♦, ■) and (▲, ●) are not easily separable (assuming linear separators) whereas *Hierarchy 2* is more consistent since it groups easily separable classes together. This explains intuitively that inconsistent hierarchical structure can deteriorate the classification performance.
2) **Rare categories**: Large-scale classification problem involves thousands of categories with millions of instances with high dimensions. In datasets of such a large-scale, skewed class distributions is observed where plenty of classes have few examples for training making it considerably difficult to learn a generalized model. This is known as the rare categories problem [7] and datasets exhibit power-law distribution for examples per class as shown in eq. (1).

$$P(s_i > S) \propto S^{-\gamma} \tag{1}$$

where s_i denotes the size of class i and γ denotes the power law exponent. Figure 2(a) and (b) shows the power-law distribution followed in large-scale DMOZ-2010 and DMOZ-2012 datasets, respectively.
3) **Less cohesive or overlapping categories**: Within real-world datasets [23], two or more categories may be so similar that learning a discriminative model between them is difficult. In such cases, it is beneficial for classification to merge these categories into one common category. For example, categories like 'softball' and 'baseball' have several common features such as 'pitch', 'ball', 'bat', 'gloves' due to which discrimination between instances of these two classes becomes



(a) DMOZ-2010 (b) DMOZ-2012

Figure 2. **Power law distribution for DMOZ-2010 and DMOZ-2012 datasets.**

difficult without having enough training examples (less likely for large-scale datasets). Alternatively, due to lack of domain knowledge instances with different characteristics, classes are put together into a common category making it less cohesive. In such cases, it is beneficial to split a category into multiple sub-categories for effective classification. Typically, classes with more generic representation are less cohesive whereas over simplified classes have overlapping instances.

4) **Orphan node identification**: Due to change in distinctive characteristics of data over time; we often need to evolve/adapt the hierarchy by creating a new set of classes whose instances have not yet been observed in the training data. This problem referred as orphan node detection (or identification) is an important factor to consider for improving the HC performance [24].

IV. Methods

In this section, we illustrate our proposed approach to handle issues discussed in Section III. Firstly, we provide solution to handle each of these issues independently and then we discuss an integrated framework to combine these solutions. It must be noted that we are also interested in solving the large-scale problem and therefore desire solutions that are easily parallelizable.

A. Modification Strategies

1) **Hierarchical structure inconsistency**: Cross-linkage of inconsistent relationships is an effective method to solve hierarchical structural inconsistency. The decision to perform cross-linkage is based on the average pairwise cosine similarity score between classes and their relative positions in the hierarchy. Intuitively, the idea is to group classes with high similarities together to a common parent. To this end, first we identify a set of class pairs with high similarity scores and than we iteratively perform check on these class pairs if they are grouped together. If not than we apply elementary operations – node creation, parent-child rewiring and node deletion to correct inconsistencies as done in Naik et al. [22]. Node creation helps to group similar classes from different hierarchical branches by creating new node with only children as the similar classes. This operation is used only when a proper subset of the classes from different branches are similar, otherwise parent-child rewiring operation is performed which simply reassigns the parent of one class to other. Finally, node deletion operation helps to remove irrelevant nodes with 0 or 1 children.

2) **Rare categories**: Feature Selection (FS) is an effective method for dealing with rare categories [16]. It helps to prevent the model over-fitting by considering only the relevant features; thereby improving the classification performance on rare categories classes. We propose to use the Gini-index measure to determine the relevance of each feature in classification as it gives comparatively better results over other measures and can be easily parallelized. Gini-index corresponding to feature f_i can be computed as shown in eq. (2). Smaller the value of Gini-index better the feature is for classification.

$$\mathbf{Gini-index}(f_i) = 1 - \sum_{k=1}^{\mathcal{L}} \Big(p(k|f_i)\Big)^2 \qquad (2)$$

where $p(k|f_i)$ is the conditional probability of class k given feature f_i. Appropriate set of features $S_{\mathcal{F}}$ at each internal node is determined using small validation dataset.

3) **Less cohesive or overlapping categories**: Overlapping categories have a high degree of similarities between them. In order to identify overlapping categories (possible only between siblings), we first compute the pairwise cosine feature similarity and consider two or more siblings as overlapping if the similarity score exceeds

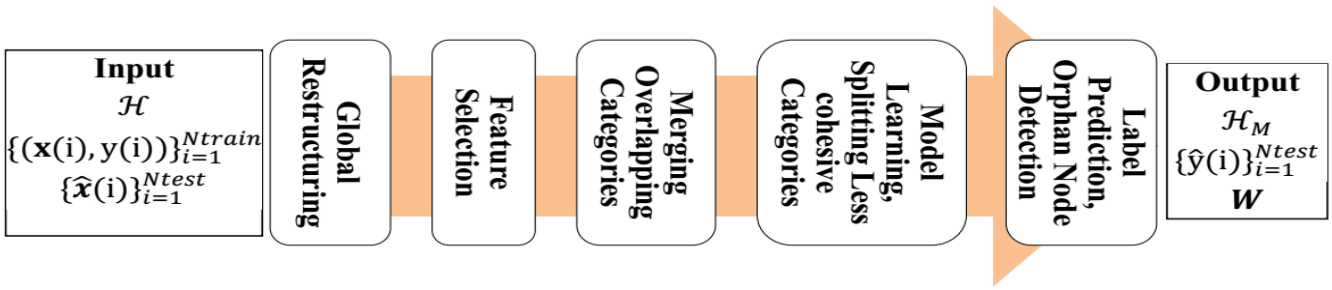


Figure 3. **Integrated framework pipeline for large-scale hierarchical classification.**

Algorithm 1 Orphan Node Identification

Data: Test instance $\widehat{\mathbf{x}}(i)$, Hierarchy $\mathcal{H}$
Result: Optimal label assignment $\widehat{y}(i)$, Modified Hierarchy $\mathcal{H}_M$
$n = root$;
DetectOrphanNode $= 0$;
while *($n \notin \mathcal{L}$)* **and** *(DetectOrphanNode $== 0$))* **do**
  Compute probability score for children of *n-th* node using eq. (3)
  /* **orphan node check** */
  if $NearlyUniform\Big(\forall_{c \in \mathcal{C}(n)}\ p(\mathcal{C}_n^c|\widehat{\mathbf{x}}(i))\Big)$ **then**
    Create new class c' at level l
    Initialize parameters for new class $\mathbf{w}_n^{c'}$ with training instance as $\widehat{\mathbf{x}}(i)$
    Parent(c') $= n$;
    $\widehat{y}(i) = c'$;
    $\mathcal{L} = \mathcal{L} \cup \{c'\}$;
    /* modified hierarchy with new orphan node */
    $H_M = OrphanNode(\mathcal{H})$;
    DetectOrphanNode $= 1$;
  else
    $n = \underset{c \in \mathcal{C}(n)}{\mathbf{argmax}} \Big(p(\mathcal{C}_n^c|\widehat{\mathbf{x}}(i))\Big)$;
    $\widehat{y}(i) = n$;
  end
end
return $\mathcal{H}_M, \widehat{y}(i)$

Algorithm 2 Criterion to determine nearly uniform probability score [24]

Data: Probability score of n-th children $p(\mathcal{C}_n^c|\widehat{\mathbf{x}}(i))$
Result: $True$ or $False$
$maxProbability = \underset{c \in \mathcal{C}(n)}{\mathbf{argmax}} \Big(p(\mathcal{C}_n^c|\widehat{\mathbf{x}}(i))\Big)$
$minProbability = \underset{c \in \mathcal{C}(n)}{\mathbf{argmin}} \Big(p(\mathcal{C}_n^c|\widehat{\mathbf{x}}(i))\Big)$
if $\Big(\frac{maxProbability}{minProbability} < 2\Big)$ *and* $\Big(maxProbability < \frac{2}{|\mathcal{C}(n)|}\Big)$ **then**
  $NearlyUniform = True$
else
  $NearlyUniform = False$
end
return $NearlyUniform$

a certain threshold. For experimental evaluations, we set the threshold value as 0.5. To determine the less cohesive categories, we use the feedback from classification results on a validation dataset [23]. Categories are identified to be less cohesive if $P_c << \overline{P}$ where P_c is the precision of a category and $\overline{P}$ is the average precision of siblings categories. Moreover, we use k-means clustering for splitting less cohesive categories.

4) **Orphan node identification**: We followed an exploratory learning approach to identify orphan nodes [24]. Algorithm 1 shows how the orphan nodes are determined during the prediction phase. Essentially, we start predicting the label of instances in the top-down order by computing the probability score given in eq. (3) and selecting the best child with highest probability score. At each level, we check if new node (*i.e.* orphan) needs to be created for better classification. The decision to create a new node is based on the probability score of children nodes; nearly uniform score (computed using MinMax partition as shown in Algorithm 2) indicates that the test instance cannot be confidently assigned to any of the children nodes and hence we create a new node. Otherwise we select the best child (with highest probability score) and proceed down the level. This process is repeated until the leaf node is reached or new orphan node is created.

$$p(\mathcal{C}_n^c|\widehat{\mathbf{x}}(i)) = \frac{exp\Big((\mathbf{w}_n^c)^T\widehat{\mathbf{x}}(i)\Big)}{\sum_{c \in \mathcal{C}(n)} exp\Big((\mathbf{w}_n^c)^T\widehat{\mathbf{x}}(i)\Big)} \quad (3)$$

B. An Integrated Framework for HC

Figure 3 presents our proposed integrated framework. The order in which the above steps are executed is important to achieve the maximum HC performance improvement. Global restructuring must be performed first because full set of features are important to identify inconsistent parent-child,

Table I
DATASET STATISTICS

Name	Domain	Total Node	Categories	Levels	Train	Test	Features
CLEF	Image	88	63	3	10000	1006	80
IPC	Text	553	451	3	46324	28926	1123497
DMOZ-SMALL	Text	2388	1139	5	6323	1858	51033
DMOZ-2010	Text	17222	12294	5	128710	34880	381580
DMOZ-2012	Text	13963	11947	5	383408	103435	348548

followed by the feature selection which selects relevant features amongst siblings categories for effective classification. Less-cohesive or overlapping categories are determined next which is based on feature similarities computed using features selected in the previous step. Finally, orphan node identification step is performed that is invoked during the prediction phase.

Scalability - As noted earlier, we are interested in solution which can be easily parallelized at different steps. Our proposed integrated framework is embarrassingly parallel which makes it favorable for large-scale problems. To summarize parallelization can be exploited at following steps: 1) Computing similarities between different classes. 2) Selecting features using Gini-index at each internal node and 3) Learning optimal model weight vectors.

C. Hierarchical Classification

Given a hierarchy $\mathcal{H}$, we train multi-class classifiers for each of the internal nodes $n \in \mathcal{N}$ in the hierarchy— to discriminate between its children nodes $\mathcal{C}(n)$. In this paper, we have used Logistic Regression (LR) as the underlying base model for training. The LR objective uses logistic loss to minimize the empirical risk and squared l_2-norm term (denoted by $||\cdot||_2^2$) to control model complexity and prevent overfitting. The objective function f_n^c for training a model corresponding to *c-th* child of node n is provided in eq. (4).

$$f_n^c = \min_{\mathbf{w}_n^c} \left[\lambda \sum_{i=1}^{T(n)} \log\left(1 + \exp\left(-y_n^c(i)\left(\mathbf{w}_n^c\right)^T \mathbf{x}(i)\right)\right) + \left|\left|\mathbf{w}_n^c\right|\right|_2^2 \right] \tag{4}$$

where $\lambda > 0$ is a mis-classification penalty parameter.

For each child c of node n within the hierarchy, we solve eq. (4) to obtain the optimal weight vector denoted by $\mathbf{w}_n^c$. The complete set of parameters for all the children nodes $\mathbf{W}_n = [\mathbf{w}_n^c]_{c \in \mathcal{C}(n)}$ constitutes the learned multi-class classifiers at node n whereas total parameters for all internal nodes $\mathbf{W} = [\mathbf{W}_n]_{n \in \mathcal{N}}$ constitutes the learned model for Top-Down (TD) classifier. Label prediction for a test instance $\widehat{\mathbf{x}}(i)$ is done in conjunction with the orphan node identification as shown in Algorithm 1.

V. Experimental Protocol

A. Datasets

We have used image and text datasets with varying characteristics for evaluation purposes. Various statistics of the datasets used in our experiments are listed in Table I. All these datasets are single-labeled and the examples are assigned to the leaf nodes in the hierarchy. For text datasets, we have preprocessed the datasets by applying the tf-idf transformation with $l2$-norm normalization on the word-frequency feature vector. Further, these datasets do not have orphan nodes therefore for evaluation we randomly remove 1% of the classes from training dataset. Descriptions of the datasets used in our experiments are as follows:

CLEF [25] - Datasset contains medical images annotated with Information Retrieval in Medical Applications (IRMA) codes. Each image is represented by the 80 features that are extracted using local distribution of edges method.

IPC[3] - Collection of patent documents organized in International Patent Classification (IPC) hierarchy. This is high-dimensional dataset.

DMOZ-SMALL, DMOZ-2010 and DMOZ-2012[4] - Multiple web documents organized in the hierarchical structure. Datasets have been released as the part of the LSHTC[5] challenge in the year 2010, 2012. DMOZ datasets are characterized by high-dimensional features and imbalance class distribution were more than 70% of the classes belong to rare categories. Moreover, DMOZ-2010 and DMOZ-2012 are large-scale datasets with more than 10000 classes. Since, we don't have access to DMOZ-2010 and DMOZ-2012 datasets true labels for test instances we split the train dataset into 80:20 ratio for experimental evaluation.

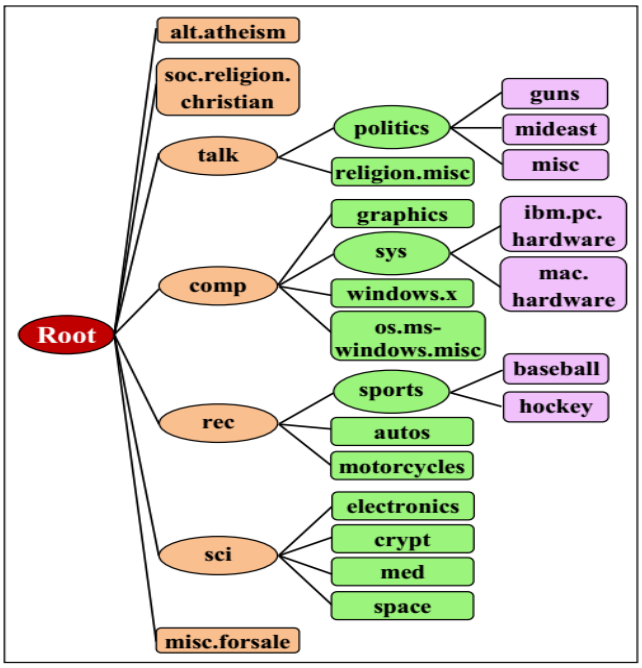
B. Evaluation Metrics

We have used the standard set based performance measures Micro-F_1 and Macro-F_1 [26] for evaluating the performance of learned models.

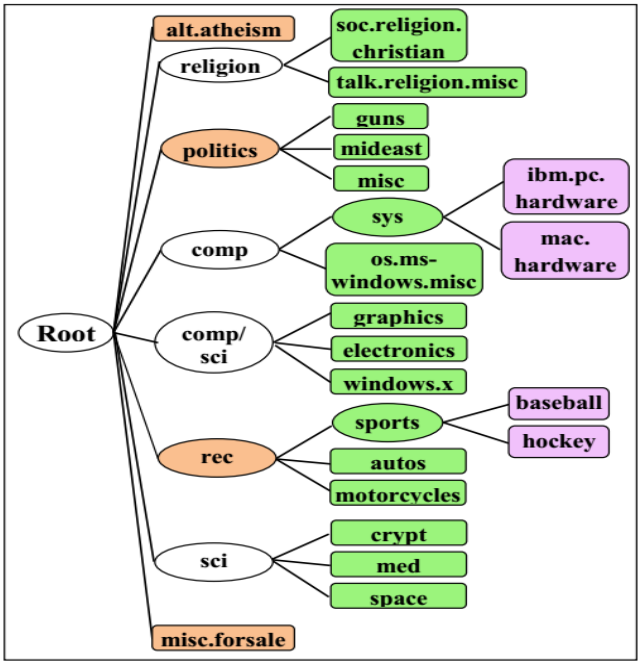
Micro-F_1 - To compute Micro-F_1, we sum up the category specific true positives (TP_l), false positives (FP_l) and false negatives (FN_l) for different leaf categories and compute the Micro-F_1 score as follows:

$$Micro - F_1 = \frac{2PR}{P + R}$$

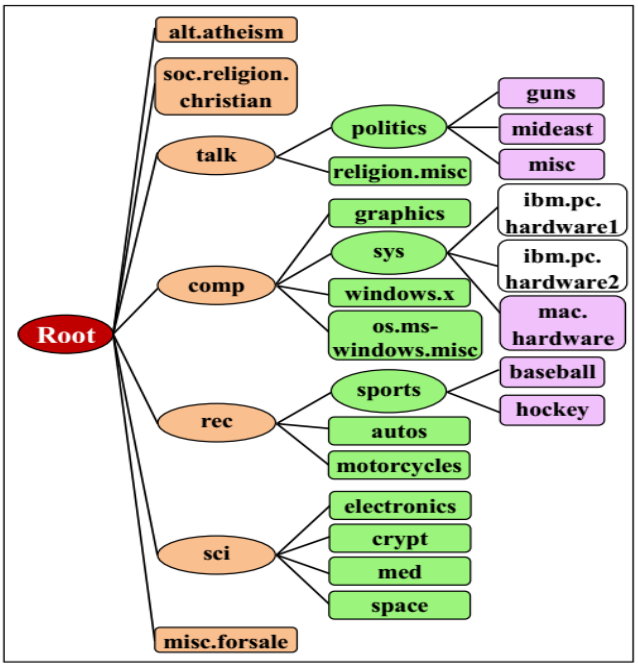
[3]http://www.wipo.int/classifications/ipc/en
[4]http://dmoz.org
[5]http://lshtc.iit.demokritos.gr



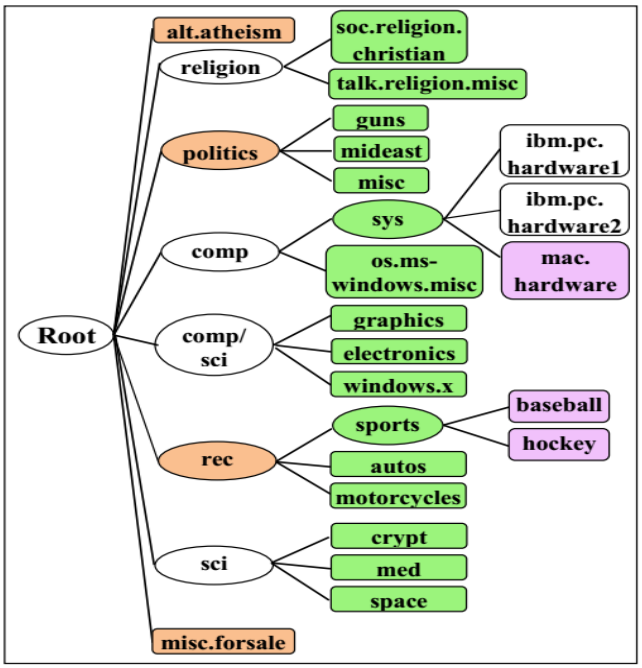
(a) Newsgroup dataset original (predefined) hierarchy ($\mathcal{H}$)



(b) Original hierarchy modified after global restructuring step



(c) Original hierarchy modified after less cohesive or ovelapping step



(d) Modified hierarchy after global restructuring and less cohesive or ovelapping step

Figure 4. **Newsgroup dataset (a) Original (predefined) hierarchy and modified hierarchy obtained after (b) Global restructuring (c) Less cohesive or overlapping categories step and (d) Global restructuring and less cohesive or ovelapping steps. Structural changes are marked in white color.**

Table II
MICRO-F_1 AND MACRO-F_1 PERFORMANCE COMPARISON USING DIFFERENT HIERARCHICAL STRUCTURES OBTAINED FROM NEWSGROUP DATASET SHOWN IN FIGURE 4

Metric	Figure 4(a)	Figure 4(b)	Figure 4(c)	Figure 4(d)
$\mu F_1(\uparrow)$	77.04 (0.1821)	78.00 (0.0231)	81.24 (0.0832)	**82.02 (0.0523)**
$MF_1(\uparrow)$	77.94 (0.0412)	78.20 (0.0132)	81.94 (0.0400)	**82.16 (0.0124)**

Table shows mean and (standard deviation) in bracket across five runs.

where P and R are the overall precision and recall values for all classes taken together and given by:

$$P = \frac{\sum_{l \in \mathcal{L}} TP_l}{\sum_{l \in \mathcal{L}} (TP_l + FP_l)}$$
$$R = \frac{\sum_{l \in \mathcal{L}} TP_l}{\sum_{l \in \mathcal{L}} (TP_l + FN_l)}$$

Macro-F_1 - Unlike Micro-F_1, Macro-F_1 gives equal weight to all the categories so that the average score is not skewed in favor of the larger categories. It is defined as follows:

$$Macro - F_1 = \frac{1}{|\mathcal{L}|} \sum_{l \in \mathcal{L}} \frac{2 P_l R_l}{P_l + R_l}$$

where P_l and R_l are the precision and recall values for leaf category $l \in \mathcal{L}$ and given by:

$$P_l = \frac{TP_l}{TP_l + FP_l}$$
$$R_l = \frac{TP_l}{TP_l + FN_l}$$

For test dataset with orphan nodes, results are reported by considering only the *seed* classes $i.e.$ having at least one training instance as reported in Dalvi et al. [24].

C. Methods for Comparison

We use various TD methods for comparison purpose.

Top-Down Logistic Regression (TD-LR): In this method, we use the original hierarchy for model learning. For predicting the labels of unlabeled test instances, we start at root node and recursively select the best child nodes until leaf node is reached. This is the baseline model used in our experiments.

HC with Incomplete Class Hierarchies [24]: Similar to TD-LR, we use the original hierarchy for model learning. However, prediction for unlabeled test instances are done using Algorithm 1. This method has advantage over baseline TD-LR since it can detect the orphan nodes during the testing (prediction) phase. We refer to this method as TD-LR + ICH.

HC with Integrated Framework: This is our proposed method discussed in Figure 3. We refer to this method as TD-LR + IF.

D. Experimental Details

For all the experiments, we divide the training dataset into train and small validation dataset in the ratio 90:10. The train dataset is used to train TD model whereas the validation dataset is used to tune the parameters. The model is trained for a range of mis-classification penalty parameter (λ) values in the set $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$ with best value selected using a validation dataset. Adapting the best parameter, we retrain the model on the entire training dataset and measure the performance on a separate held-out test dataset. For feature selection, we choose the best set of features $S_{\mathcal{F}}$ using the validation dataset. We use the liblinear solver[6] [27] for efficient training of TD model. Source code implementation of our proposed integrated framework discussed in this paper is made available at our website[7] for results reproducibility and future research.

VI. RESULTS AND DISCUSSION

A. Case Study

To understand the strength and effectiveness of each step in the integrated framework we perform case study on smaller Newsgroup[8] dataset containing 20 classes, 61188 features, 7505 test and 11269 training instances (evenly distributed across 20 classes). Figure 4(b) and (c) shows the intermediate hierarchy obtained on the original Newsgroup dataset (shown in Figure 4(a)) after global restructuring, less cohesive or overlapping categories removal, respectively. Figure 4(d) shows the combined hierarchy obtained after restucturing and less cohesive or overlapping categories removal. For evaluating each of this hierarchy we randomly choose five different sets of train and test split in the same ratio as original dataset. Table II shows the performance comparison with different hierarchical structures. We can see from the table that modified hierarchy obtained by combination of global restructuring and less cohesive or overlapping categories removal achieves the best performance with an average $Macro - F_1$ score of 82.16 (0.0124) as opposed to original hierarchy that achieves a score of 77.94 (0.0412). Further, each of the modified hierarchy shown in Figure 4(b) and Figure 4(c) has better performance in comparison to the original hierarchy. This result shows that modified hierarchy is more beneficial for HC.

[6]http://www.csie.ntu.edu.tw/∼cjlin/liblinear
[7]http://Anonymous/IntegratedFramework
[8]http://qwone.com/∼jason/20Newsgroups

Table III
PERFORMANCE COMPARISON WITH AND WITHOUT ORPHAN NODE IDENTIFICATION

Removed Nodes for Experiment	Identified Nodes	Metric	Accuracy	
			with Orphan Node	without Orphan Node
soc.religion.christian, guns, windows.x, electronics	*guns, electronics*	$Micro - F_1$	**76.34** (0.0123)	75.23 (0.0182)
		$Macro - F_1$	**76.00** (0.0224)	74.68 (0.0682)
windows.x, mideast, crypt, misc.forsale	*mideast, crypt, misc.forsale*	$Micro - F_1$	**76.48** (0.0624)	75.22 (0.0213)
		$Macro - F_1$	**76.26** (0.0817)	74.64 (0.0518)

Table shows mean and (standard deviation) in bracket across five runs.

Table IV
$Micro - F_1$ AND $Macro - F_1$ PERFORMANCE COMPARISON

Name	TD-LR (Baseline)		TD-LR + ICH (*Dalvi et al.* [24])		TD-LR + IF (Proposed)	
	Micro-F_1	Macro-F_1	Micro-F_1	Macro-F_1	Micro-F_1	Macro-F_1
CLEF	73.06 (0.0231)	33.86 (0.0821)	73.56 (0.0526)	34.13 (0.0642)	**74.63** (0.0521)	**34.98** (0.0916)
IPC	47.24 (0.2645)	41.25 (0.1737)	48.92 (0.2536)	43.00 (0.6150)	**49.26**△ (0.1427)	**44.12**▲ (0.3638)
DMOZ-SMALL	40.12 (0.2913)	23.63 (0.1245)	40.29 (0.1734)	24.17 (0.2234)	**42.34**▲ (0.2236)	**26.23**▲ (0.1194)
DMOZ-2010	36.18 (0.2437)	21.42 (0.1470)	36.20 (0.2522)	21.44 (0.0167)	**38.24**▲ (0.1243)	**23.11**▲ (0.1850)
DMOZ-2012	38.42 (0.1029)	23.88 (0.0324)	38.92 (0.0124)	24.24 (0.1142)	**40.87**▲ (0.0243)	**26.34**▲ (0.185)

Table shows mean and (standard deviation) in bracket across five runs. ▲ (△) indicates that improvements are statistically significant with 0.01 (0.05) significance level. Comparison is between our proposed TD-LR + IF model and best baseline model $i.e.$, TD-LR + ICH.

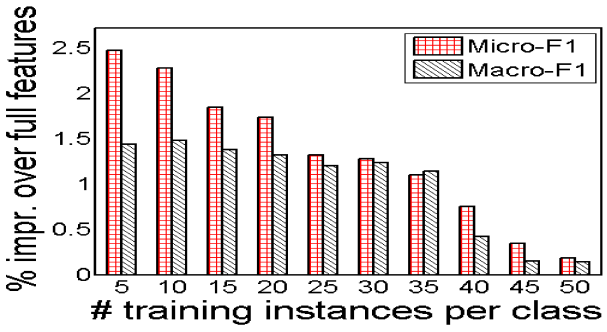


Figure 5. **Performance comparison of feature selection with full features with varying distribution of training instances.**

To understand the benefits of feature selection step we perform experiments by varying the number of instances in each class between 5 and 50. Figure 5 shows the comparison of feature selection over using full set of features. Rare categories with 5-10 instances per class benefits significantly with feature selection. Moreover, the improvement in performance decreases as the number of instances per class increases because more training instances leads to better generalized models with full set of features. It should be noted that feature selection is also helpful in terms of training, prediction time and memory requirements to store model weight vectors.

Finally, to access the orphan node identification step, we performed experiments by randomly removing 4 classes from training dataset. Results obtained with different sets of orphan nodes is shown in Table III. Our approach effectively identifies a subset of orphan nodes which is helpful in improving the HC performance of seed classes.

B. Accuracy Comparison

Table IV shows the Micro-F_1 and Macro-F_1 performance comparison of various models. Each experiment is repeated five times by randomly selecting the train and test dataset. Our proposed integrated framework consistently outperforms the other approaches across different datasets. To access the performance improvement we conducted pairwise statistical significance tests between the best baseline model TD-LR + ICH and our proposed model TD-LR +IF. Specifically, we perform micro-sign test for Micro-F_1[28] and non-parametric wilcoxon rank test for Macro-F_1 scores (it should be noted that significance tests are between two models for single run). Statistical significant results are denoted by ▲(△) for significance with p-value 0.01 (0.05). We can see from the table that most of the results obtained are statistically significant with our proposed framework.

Baseline TD-LR model has the worse performance due to the presence of inconsistencies within the hierarchy along with rare categories, less cohesive or overlapping categories and orphan nodes. On comparison, TD-LR + ICH model is slightly better because this method can deal with orphan nodes. Our approach TD-LR + IF is best performing method because it can effectively deal with issues that are faced by other two models.

Table V
RUNTIME (IN MINUTES) AND MEMORY REQUIREMENTS COMPARISON

Name	TD-LR (Baseline)			TD-LR + ICH (*Dalvi et al.* [24])			TD-LR + IF (Proposed)		
	train	prediction	memory	train	prediction	memory	train	prediction	memory
CLEF	0.28	< 1	27.52 KB	0.28	< 1	28.16 KB	3.12	< 1	19.00 KB
IPC	68.58	1.91	02.46 GB	68.58	2.31	02.50 GB	102.54	2.00	01.48 GB
DMOZ-SMALL	3.17	< 1	00.48 GB	3.17	1.24	00.50 GB	18.29	1.06	00.31 GB
DMOZ-2010	6418	20.60	26.10 GB	6418	28.24	26.56 GB	10826	24.08	15.95 GB
DMOZ-2012	19193	49.9	19.30 GB	19193	62.49	19.90 GB	28107	53.46	14.02 GB

C. Runtime and Memory Comparison

Table V shows the average training, prediction time and memory comparison of various models. Our model (TD-LR + IF) has expensive training time due to overhead associated with feature selection and pairwise similarity computation steps. However, we would like to emphasize that both these steps are embarrassingly parallel. On comparing prediction time, our model and TD-LR + ICH model are expensive due to extra time required for orphan node identification. Comparatively, TD-LR + ICH is more expensive due to mixed integer programming being solved [24] for making the final predictions. Baseline TD-LR model takes least time because predicting label involves recursive selection of best child node along tree path only.

On comparing memory required to store model parameters for different approaches, our approach requires the least space due to feature selection at each internal node which reduces the dimensions of learned model weight vectors and hence memory. In fact, for large-scale DMOZ-2010 and DMOZ-2012 datasets improvement is much more significant with $\sim$40% reduction in memory space in comparison to baseline TD-LR model.

VII. CONCLUSION AND FUTURE WORK

In this paper, we propose an integrated framework to solve the major issues faced by large-scale HC. Unlike previous HC approaches, our framework can identify the orphan nodes during the testing phase. We also performed an extensive analysis with case study to determine the importance of each step in HC. Our experimental evaluation on various datasets shows that the proposed framework can achieve significant improvements in terms of accuracy and memory requirements. Our approach is scalable due to high degree of parallelization at each step which can be exploited to improve the training time of model learning.

In future, we plan to extend our work by learning more complex model at each node such as multi-task learning method in conjunction with our proposed framework. We also plan to explore other hierarchy pruning approaches [7, 22] with our framework.

REFERENCES

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–255.

[2] I. Partalas, A. Kosmopoulos, N. Baskiotis, T. Artieres, G. Paliouras, E. Gaussier, I. Androutsopoulos, M.-R. Amini, and P. Galinari, "Lshtc: A benchmark for large-scale text classification," *arXiv preprint arXiv:1503.08581*, 2015.

[3] D. Koller and M. Sahami, "Hierarchically classifying documents using very few words," in *Proceedings of the fourteenth International Conference on Machine Learning*, 1997, pp. 170–178.

[4] S. Dumais and H. Chen, "Hierarchical classification of web content," in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, 2000, pp. 256–263.

[5] L. Cai and T. Hofmann, "Hierarchical document categorization with support vector machines," in *CIKM*, 2004, pp. 78–87.

[6] O. Dekel, J. Keshet, and Y. Singer, "Large margin hierarchical classification," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 27.

[7] R. Babbar, I. Partalas, E. Gaussier, and M.-R. Amini, "On flat versus hierarchical classification in large-scale taxonomies," in *Advances in Neural Information Processing Systems*, 2013, pp. 1824–1832.

[8] P. N. Bennett and N. Nguyen, "Refined experts: improving classification in large taxonomies," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, 2009, pp. 11–18.

[9] A. Naik and H. Rangwala, "A ranking-based approach for hierarchical classification," in *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2015, pp. 1–10.

[10] A. Sun and E.-P. Lim, "Hierarchical text classification and evaluation," in *Proceedings of the IEEE International Conference on Data Mining(ICDM)*. IEEE, 2001, pp. 521–528.

[11] G.-R. Xue, D. Xing, Q. Yang, and Y. Yu, "Deep classification in large-scale text hierarchies," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 619–626.

[12] S. Gopal, Y. Yang, B. Bai, and A. Niculescu-Mizil, "Bayesian models for large-scale hierarchical classification," in *Advances in Neural Information Processing Systems*, 2012, pp. 2411–2419.

[13] S. Gopal and Y. Yang, "Recursive regularization for large-scale classification with hierarchical and graphical dependencies," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2013, pp. 257–265.

[14] A. Charuvaka and H. Rangwala, "Hiercost: Improving large scale hierarchical classification with cost sensitive learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2015, pp. 675–690.

[15] L. Xiao, D. Zhou, and M. Wu, "Hierarchical classification via orthogonal transfer," in *ICML*, 2011, pp. 801–808.

[16] A. Naik and H. Rangwala, "Embedding feature selection for large-scale hierarchical classification," in *IEEE International Conference on Big Data*, 2016, pp. 1212–1221.

[17] C. N. Silla Jr and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 31–72, 2011.

[18] A. Naik and H. Rangwala, "Inconsistent node flattening for improving top-down hierarchical classification," in *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2016, pp. 379–388.

[19] T.-Y. Liu, Y. Yang, H. Wan, H.-J. Zeng, Z. Chen, and W.-Y. Ma, "Support vector machines classification with a very large-scale taxonomy," *ACM SIGKDD Explorations Newsletter*, vol. 7, no. 1, pp. 36–43, 2005.

[20] B. Dalvi, W. W. Cohen, and J. Callan, "Exploratory learning," in *ECML PKDD*, 2013, pp. 128–143.

[21] C. Lin, Y. Zou, J. Qin, X. Liu, Y. Jiang, C. Ke, and Q. Zou, "Hierarchical classification of protein folds using a novel ensemble classifier," *PloS one*, vol. 8, no. 2, p. e56499, 2013.

[22] A. Naik and H. Rangwala, "Filter based taxonomy modification for improving hierarchical classification," *http://arxiv.org/abs/1603.00772*, 2016.

[23] L. He and X. Sun, "Automatic maintenance of the category hierarchy," in *Proceedings of the International Conf, on Semantics, Knowledge and Grids*, 2013, pp. 218–221.

[24] B. Dalvi, A. Mishra, and W. W. Cohen, "Hierarchical semi-supervised classification with incomplete class hierarchies," in *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, 2016, pp. 193–202.

[25] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Džeroski, "Hierarchical annotation of medical images," *Pattern Recognition*, vol. 44, no. 10, pp. 2436–2449, 2011.

[26] Y. Yang, "An evaluation of statistical approaches to text categorization," *Information retrieval*, vol. 1, no. 1, pp. 69–90, 1999.

[27] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of machine learning research*, vol. 9, no. Aug, pp. 1871–1874, 2008.

[28] 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*, 1999, pp. 42–49.